TRANSPORTATION SCIENCE
Vol. 47, No. 2, May 2013, pp. 181–197
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.1120.0422

Expected Future Value Decomposition Based Bid Price Generation for Large-Scale Network Revenue Management

L. F. Escudero
Departamento de Estadística e Investigación Operativa, Universidad Rey Juan Carlos, Madrid, Spain,
laureano.escudero@urjc.es

J. F. Monge
Centro de Investigación Operativa, Universidad Miguel Hernández, Elche (Alicante), Spain,
monge@umh.es

D. Romero Morales, J. Wang
Saïd Business School, University of Oxford, Oxford, United Kingdom
{dolores.romero-morales@sbs.ox.ac.uk, jingbo.wang@oba.co.uk}

This paper studies a multistage stochastic programming (SP) model for large-scale network revenue management. We solve the model by means of the so-called expected future value (EFV) decomposition via scenario analysis, estimating the impact of the decisions made at a given stage on the objective function value related to the future stages. The EFV curves are used to define bid prices on bundles of resources directly, as opposed to the traditional additive approach. We compare our revenues to those obtained by additive bid prices, such as the bid prices derived from the deterministic equivalent model (DEM) of the compact representation of the SP model. Our computational experience shows that the revenues obtained by our approach are better for middle-range values of the load factor of demand, whereas the differences among all the approaches we have tested are insignificant for extreme values. Moreover, our approach requires significantly less computation time than does the optimization of DEM by plain use of optimization engines. Problem instances with 72 pairs of bundle-fare classes have been solved in less than one minute, with 800 pairs in less than five minutes, and with 4,000 pairs in less than one hour. The time taken by DEM was, in general, of one order of magnitude higher. Finally, for the three largest problem instances, and after two hours, the expected revenue returned by DEM was below that obtained by EFV by 13.47%, 17.14%, and 38.94%, respectively.

Key words: network revenue management; scenario trees; stochastic dynamic programming; expected future value curves; nonadditive bid prices; load factor of demand
History: Received: September 2010; revision received: July 2011; accepted: March 2012. Published online in *Articles in Advance* June 22, 2012.

1. Introduction

Revenue management aims to maximize the revenue of selling limited quantities of a set of resources by means of demand management decisions. A resource in revenue management is usually a perishable product or service, such as seats on a single flight leg or hotel rooms for a given date. It is common in revenue management that multiple resources are sold in "bundles." For instance, connecting flight legs are sold on a single ticket and hotel customers may stay multiple nights. In this case, the lack of availability of any resource will prevent sales of the bundle, which creates interdependence among these resources. Consequently, the demand management decisions of these resources must be coordinated, which is generally referred to as the network revenue management problem (Talluri and van Ryzin 2004).

The state-of-the-art approach to this problem is to use bid-price policies (Talluri and van Ryzin 1998), in which a bid price is generated for each bundle and a request to purchase the bundle is accepted if and only if the associated revenue exceeds the bid price. Bid-price policies are not optimal, in general, but they are very popular because they are intuitive and easy to implement; see Talluri and van Ryzin (1998, 2004). Bid prices can be generated using either an "additive" or a "nonadditive" approach (Bertsimas and Popescu 2003). In the additive approach, a bid price is generated for each resource, and the bid price of a bundle is defined as the sum of the bid prices of all resources used by the bundle. On the other hand, in the nonadditive approach a bid price is generated directly for each bundle.

A number of optimization models have been proposed for generating additive bid prices, mainly

differing in the way uncertainty in demand is incorporated. The simplest and most popular model is the so-called deterministic linear programming (DLP) due to D'Sylva (1982), Glover et al. (1982), and Wong, Koppelman, and Daskin (1993). The demand for each bundle, which is stochastic by nature, is replaced by the mean value. The objective of the optimization model is to maximize the revenue such that sales are bounded by the mean demand and the capacities on the resources are not violated. The bid prices of the resources are calculated as the dual variables of the capacity constraints (Glover et al. 1982; Wong, Koppelman, and Daskin 1993). Talluri and van Ryzin (1998) showed that DLP is asymptotically optimal as capacity and demand increase linearly and with the same rate. The main drawback of DLP is that it considers only the mean demand and ignores all the uncertainty. As a result, it could suffer when the variance of demand is high.

A natural extension of DLP is to replace the mean demand by the demand distribution, which is called the probabilistic nonlinear programming (PNLP) model. A linear programming (LP) version of PNLP was proposed by Wollmer (1992) by using discrete scenarios to represent the demand distribution. In the same manner as in DLP, the bid prices of the resources are calculated as the dual variables of the capacity constraints.

The randomized linear programming (RLP) method was first proposed by Smith and Penn (1988) and was then further investigated by Talluri and van Ryzin (1998). Basically, RLP considers a finite set of demand scenarios with their corresponding weights and then solves a different DLP for each scenario. The bid prices of the resources are calculated as the weighted average of the dual variables of the capacity constraints corresponding to the DLPs using scenario demand rather than mean demand. Topaloglu (2009a) has recently showed that RLP is asymptotically optimal under the abovementioned conditions for DLP.

PNLP and RLP consider different scenarios for the total demand over the whole booking horizon and ignore the potential intertemporal uncertainty of demand along the booking horizon. In order to account for these dynamics, Higle and Sen (2005) converted the PNLP into a two-period stochastic programming (SP) simple recourse model, in which the initial allocation of resources is revised based on the demand observed in the first period. Numerical results were provided for small and medium networks. Chen and Homem-de-Mello (2010) also studied a two-period SP model and proposed to handle the multiperiod problem by solving a sequence of two-period models.

As far as the authors are aware, Möller, Römisch, and Weber (2004, 2008) were the first to study a multiperiod SP model. In Möller, Römisch, and Weber (2004), their approach was tested on a single-resource problem instance, whereas a small network with a hub-and-spoke structure was used in Möller, Römisch, and Weber (2008). In a further work, Emich, Möller, and Römisch (2010) proposed the Lagrangian relaxation of the capacity constraints. DeMiguel and Mishra (2008) studied another multiperiod SP model, where protection levels are not modeled. They focus on examining different methods for generating the scenarios trees, and numerical results are provided for small and medium sized networks. Recently, Haensel, Mederer, and Schmidt (2011) have proposed a stochastic mixed integer programming model for the rental car industry.

Additive bid prices are easy to implement and popular in practice; see Adelman (2007), Akan and Ata (2009), Ball and Queyranne (2009), Kunnumkal and Topaloglu (2010), Topaloglu (2009b). However, nonadditive bid prices could provide a more accurate reflection of the opportunity cost of bundles. As argued in Talluri and van Ryzin (1999), the opportunity cost of a bundle is basically determined by the most constraining resource used by the bundle, and therefore additive bid prices could be restrictive compared to nonadditive ones. Bertsimas and Popescu (2003) propose a framework for generating nonadditive bid prices. Given a model for network revenue management, the bid price of a bundle is calculated as the change on the expected revenue when the capacity on each resource used by the bundle is reduced by one unit. However this approach can be computationally expensive because a different optimization problem needs to be solved for each bundle.

In this study, we propose an SP based method for generating nonadditive bid prices. Noticing that the high computational complexity of multiperiod SP models has prevented their application in large networks, we use the expected future value (EFV) decomposition algorithm proposed in Cristóbal, Escudero, and Monge (2009). Adopting a stochastic dynamic programming (SDP) approach (Powell 2007; Ross 1995), the EFV decomposition algorithm first combines time periods into stages, then divides the SP problem into intra-stage subproblems that are linked to each other, and finally solves them iteratively. The crucial ingredient of the algorithm is the family of so-called EFV curves, estimating the impact of the decisions to be made at a given stage on the objective function value related to the future stages. The EFV curves are used to define nonadditive bid prices on bundles directly.

We apply the EFV decomposition algorithm to the two multiperiod SP models proposed in Möller, Römisch, and Weber (2008) and DeMiguel and Mishra (2008) to derive nonadditive bid prices. The deterministic equivalent model (DEM) of the compact

representation of these two multiperiod SP models is used to derive additive bid prices. We compare the EFV revenues with the ones obtained by other five additive approaches including DLP, RLP, DEM, and the Perfect Hindsight. The revenue performance of the nonadditive bid prices from EFV is tested against the benchmarking ones in a rolling horizon simulation in three test networks, for a range of values of the load factor of demand, which indicates how the resource capacity relates to the demand. For extreme values of the load factor, all models yield similar revenues. For middle-range values, our nonadditive bid prices are consistently the best ones. RLP ranks third and is followed by DEM. Moreover, our approach requires significantly less computation time than does the optimization of DEM by plain use of optimization engines. Problem instances with 72 pairs of bundle-fare classes have been solved in less than one minute, with 800 pairs in less than five minutes, and with 4,000 pairs in less than one hour. The time taken by DEM was, in general, of one order of magnitude higher. Finally, for the three largest problem instances, and after two hours, the expected revenue returned by DEM was below that obtained by EFV by 13.47%, 17.14%, and 38.94%, respectively.

The remainder of the paper is organized as follows. Section 2 introduces the general SP formulation and reviews the EFV decomposition algorithm. The network revenue management problem and the two multiperiod SP models are introduced in §3. In §4, we explain how the EFV decomposition algorithm can be used to generate nonadditive bid prices. Section 5 is devoted to the computational experience, in which we examine the solution accuracy, computation time, and revenue performance of our approach. Finally, we conclude the paper and discuss future research directions in §6.

2. The EFV Decomposition Algorithm

In this section, we familiarize the reader with the EFV decomposition algorithm for solving stochastic programs proposed in Cristóbal, Escudero, and Monge (2009). (See, e.g., Birge 1985; Donohue and Birge 2006; Escudero et al. 2007, 2009, 2010a, b, for some Benders and Lagrangian decomposition methods; see also van Slyke and Wets 1969 for the *L*-shaped decomposition method as an application of the Benders method for the particular case of two-stage SP problems.) Section 2.1 presents the general SP formulation. Section 2.2 decomposes the time horizon into stages. The algorithmic framework of the EFV decomposition algorithm is described in §2.3. The crucial ingredient of the algorithm, the EFV curves, is explained in detail in §2.4.

2.1. The General SP Formulation

Consider the following dynamic multilinking constraint deterministic program, in which decisions are taken over a time horizon $\mathcal{T}$ with T periods:

$$\text{maximize} \quad \sum_{t\in\mathcal{T}} c_t x_t \tag{1}$$

$$\text{s.t.} \quad \sum_{t\in\{\tau-1,\,\tau\}} A_\tau^t x_t = b_\tau \quad \forall\, \tau\in\mathcal{T}, \tag{2}$$

$$x_t \in \mathcal{X}_t \quad \forall\, t\in\mathcal{T}, \tag{3}$$

where x_t is the vector of decision variables related to time period t, c_t is the row vector of the objective function coefficients associated with x_t, A_τ^t is the coefficient matrix in the constraints related to time period τ for x_t, and b_τ is the right-hand-side (rhs) vector for the constraint related to time period τ for $\tau\in\mathcal{T}$. Additional constraints on x_t, such as nonnegativity and 0–1 constraints, are modeled in $\mathcal{X}_t$. All vectors and matrices have the appropriate dimensions. Hereafter, components of x_t that have nonzero coefficients in A_{t+1}^t will be referred as "linking" variables because they will affect the decisions in period $t+1$. In this formulation decision variables in a given period will affect the ones in the next period but not further periods into the future. As we will see in §4, this assumption is satisfied by the network revenue management problem. (Note that this assumption is only made for the sake of clarity; see, e.g., Cristóbal, Escudero, and Monge 2009 for the general case.)

Inspired by our revenue management application, we introduce stochasticity in the objective function and the rhs vectors by means of a scenario tree, such as the one illustrated in Figure 1. Each node in the tree represents a point in time where a decision will be made. Once a decision is made, some contingencies arise and information related to these contingencies will be available before the next point in time. If g is a leaf node, g represents a scenario defined by the realization of the whole set of uncertain parameters given by the root-to-leaf path in the tree. In general, any node g in the tree defines a subtree rooted at g, which constitutes a group of scenarios that are identical up to node g. Accordingly with the nonanticipativity principle (stated in Wets 1974; see also Birge and Louveaux 1997 among many others), scenarios belonging to the same group at a given time period t should have the same value for decision variables x_τ with $\tau\le t$, for all $t\in\mathcal{T}$.

In the following we introduce the notation for describing the elements of the scenario tree:

Ω, set of scenarios.
$\mathcal{G}$, set of scenario groups.
$t(g)$, time period for scenario group g, for $g\in\mathcal{G}$.
Ω_g, set of scenarios in group g, such that the scenarios that belong to the same group are

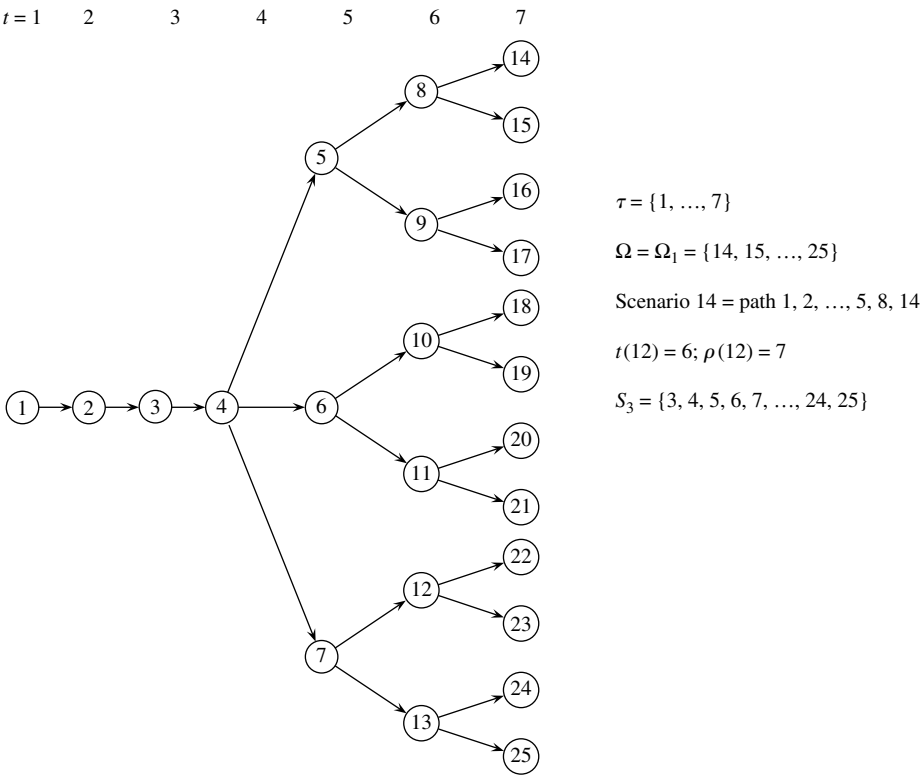

Figure 1 An Example of a Scenario Tree

identical in all realizations of the uncertain parameters up to period $t(g)$, for $g \in \mathcal{G}$. Notice that $\Omega_g \subseteq \Omega$. From now on we will without distinction use nodes in the scenario tree and scenario groups.

w^g, likelihood associated with scenario group g, for $g \in \mathcal{G}$, such that $w^g = \sum_{\omega \in \Omega_g} w^\omega$, where w^ω is the likelihood associated with scenario $\omega \in \Omega$.

$\rho(g)$, immediate ancestor node of node g, for $g \in \mathcal{G}$. (The root node has no ancestor.)

$\mathcal{S}_g$, set of nodes in the subtree whose root is node g, for $g \in \mathcal{G}$.

In order to present the stochastic version of Model (1)–(3), let the following notation be used for the decision variables and the uncertain parameters.

x^g, vector of decision variables under scenario group g, for $g \in \mathcal{G}$. It replaces the vector $x_{t(g)}$ in the deterministic model.

Uncertain Parameters:

c^g, vector of the objective function coefficients for the decision variables x^g under scenario group g, for $g \in \mathcal{G}$. It replaces the vector $c_{t(g)}$ in the deterministic model.

b^g, rhs of the constraints under scenario group g, for $g \in \mathcal{G}$. It replaces the vector $b_{t(g)}$ in the deterministic model.

The DEM of the stochastic program with complete recourse for maximizing the expected objective function value over the set of scenarios has the following so-called *compact* representation as an alternative to Model (1)–(3),

$$\text{maximize} \quad \sum_{g \in \mathcal{G}} w^g c^g x^g \tag{4}$$

$$\text{s.t.} \quad \sum_{\delta \in \{\rho(g),\, g\}} A^{t(\delta)}_{t(g)} x^\delta = b^g \quad \forall\, g \in \mathcal{G}, \tag{5}$$

$$x^g \in \mathcal{X}_{t(g)} \quad \forall\, g \in \mathcal{G}. \tag{6}$$

2.2. Breaking the Time Horizon into Stages

Our algorithm combines consecutive time periods into stages, as shown in Figure 2. The following notation describing the stages will be used throughout the paper:

$\mathcal{E}$, set of stages in the time horizon.

$\mathcal{G}^e$, set of scenario groups from stage e, for $e \in \mathcal{E}$.

$\mathcal{A}^e$, set of scenario groups associated with the root nodes from stage e, for $e \in \mathcal{E}$. Notice that $\mathcal{A}^e \subseteq \mathcal{G}^e$.

$\mathcal{C}_a$, set of nodes in the subtree rooted at node a with nodes in $\mathcal{G}^e$, for $a \in \mathcal{A}^e$, $e \in \mathcal{E}$.

$\mathcal{P}_a$, set of leaf nodes in $\mathcal{C}_a$, for $a \in \mathcal{A}^e$, for $e \in \mathcal{E}$.

Once the time horizon has been split into stages, the DEM can be divided into subproblems, which are connected by the linking variables. For each $e \in \mathcal{E}$ and $a \in \mathcal{A}^e$, we associate a subproblem with the subtree defined by node set $\mathcal{C}_a$. In Figure 2, $\mathcal{C}_5 = \{5, 8, 9, 14, \ldots, 17\}$ defines a subtree/subproblem in stage 2 with node 5 as the root node. In this example, there

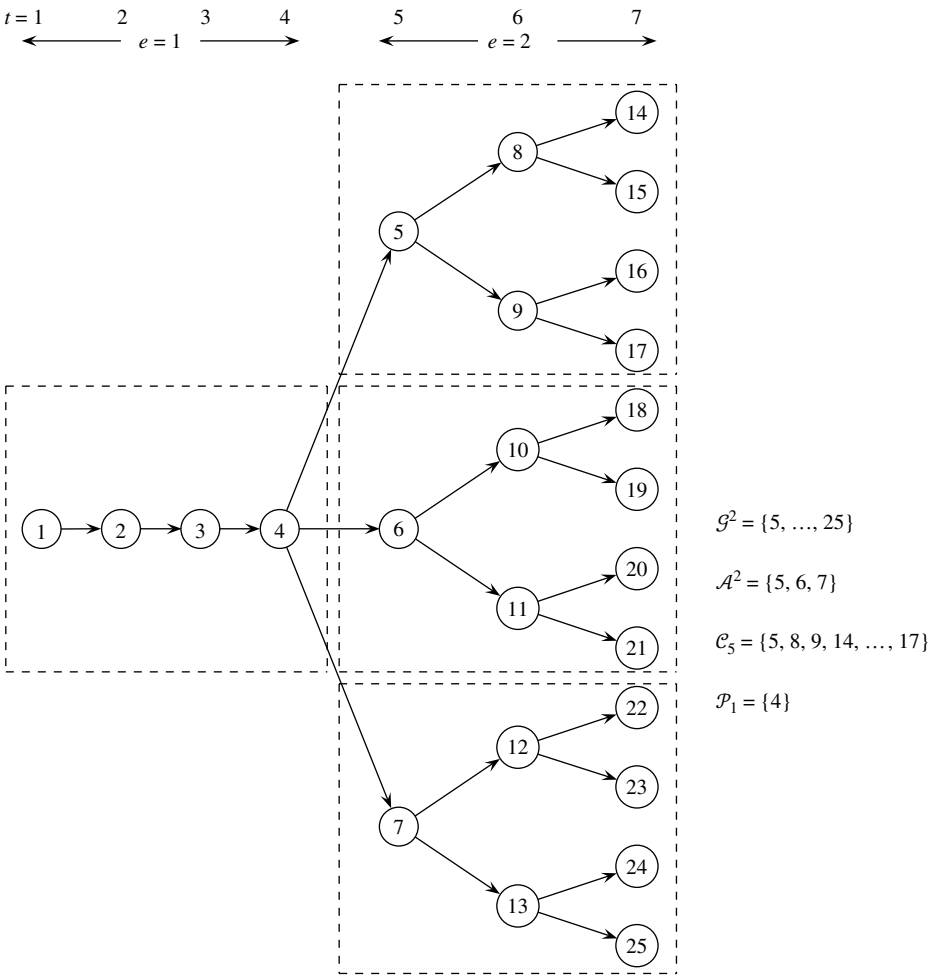

Figure 2 Breaking the Time Horizon Into Stages

are in total four subtrees/subproblems marked by dashed boxes.

The subproblem defined by node set $\mathscr{C}_a$ can be written as

$$\sigma^a(\bar{x}^{\rho(a)})$$

$$:= \text{maximize} \quad \frac{1}{w^a}\left(\sum_{g\in\mathscr{C}_a} w^g c^g x^g + \sum_{g\in\mathscr{P}_a} \lambda^g(x^g)\right), \qquad (7)$$

$$\text{s.t.} \sum_{\delta\in\{\rho(g),\, g\}} A^{t(\delta)}_{t(g)} x^\delta = b^g \quad \forall g \in \mathscr{C}_a, \qquad (8)$$

$$x^g \in \mathscr{X}_{t(g)} \quad \forall g \in \mathscr{C}_a, \qquad (9)$$

$$x^{\rho(a)} = \bar{x}^{\rho(a)}, \qquad (10)$$

where $\bar{x}^{\rho(a)}$ is the vector of decision variables the subproblem receives from its ancestor node $\rho(a)$, from which only the linking components will be actually used in the subproblem, and $\lambda^g(x^g)$ represents the expected future objective function value under the set of scenarios Ω_g, for each leaf node $g \in \mathscr{P}_a$. The function $\lambda^g(x^g)$ is therefore called the expected future value (EFV) curve of node g; see Figure 3 for an example where we have dropped the superindex g. Three straightforward observations need to be made here. First, this subproblem is only concerned with the subtree rooted at node a and this explains why the objective function must be adjusted by $1/w^a$. Second, no decisions are taken prior to the root node $a = 1$, and therefore any reference to $\bar{x}^{\rho(a)}$ will be dropped out from σ^1, including constraint (10). Third, no EFV curves are present in the subproblems in the last stage because the time horizon ends there.

The functions $\lambda^g(\cdot)$ and $\sigma^{a'}(\cdot)$ functions are closely related. Consider a leaf node g in stage e, where $g \in \mathscr{P}_a$, $a \in \mathscr{A}^e$, $e \in \mathscr{E}$. The descendent subproblems of node g in stage $e+1$ are given by node sets $\mathscr{C}_{a'}$, $\forall a' \in \mathscr{S}_g \cap \mathscr{A}^{e+1}$. We can express $\lambda^g(\cdot)$ as the weighted sum of the optimal objective function value of these subproblems:

$$\lambda^g(\cdot) = \sum_{a'\in\mathscr{S}_g\cap\mathscr{A}^{e+1}} w^{a'}\sigma^{a'}(\cdot). \qquad (11)$$

2.3. The EFV Decomposition Algorithm

Because the EFV curves are generally difficult to compute, the EFV decomposition algorithm proposes to

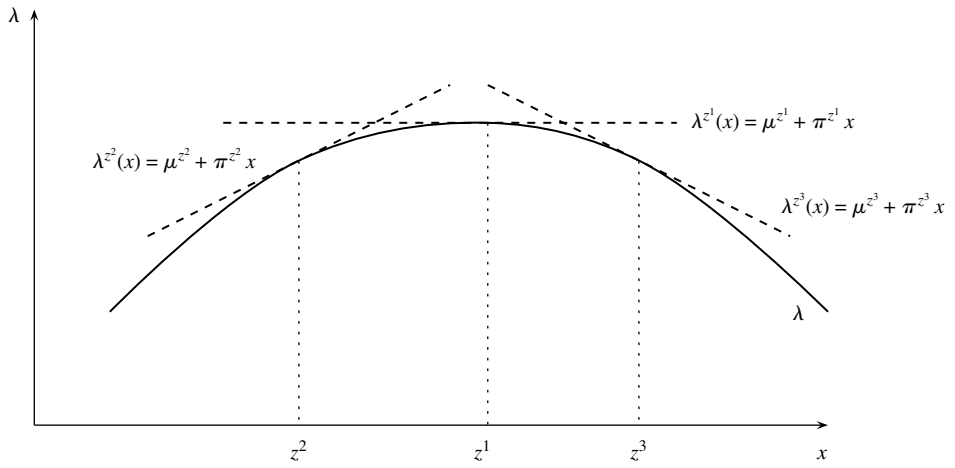

Figure 3 An Example of an EFV Curve and Its Approximation

approximate them by piecewise linear and concave functions; see Figure 3 for an example where we have dropped the superindex g.

The approximation of the EFV curves in a given stage yields an approximation of the subproblems (7)–(10) in the previous stage. Indeed, let $\mathcal{Z}^g$ denote the set of *reference levels* for x^g, where a reference level is a vector at which the function $\lambda^g(\cdot)$ is approximated by a linear function. Let μ^{gz}_{e+1} and π^{gz}_{e+1} be the parameters defining the linear function associated with reference level z. We have that subproblem (7)–(10) can be approximated by

$$\text{maximize} \quad \frac{1}{w^a}\left(\sum_{g\in\mathcal{C}_a} w^g c^g x^g + \sum_{g\in\mathcal{P}_a} \lambda^g\right) \tag{12}$$

$$\text{s.t.} \sum_{\delta\in\{\rho(g),\, g\}} A^{t(\delta)}_{t(g)} x^\delta = b^g \quad \forall g\in\mathcal{C}_a, \tag{13}$$

$$x^g \in \mathcal{X}_{t(g)} \quad \forall g\in\mathcal{C}_a, \tag{14}$$

$$x^{\rho(a)} = \bar{x}^{\rho(a)}, \tag{15}$$

$$\lambda^g \le \mu^{gz}_{e+1} + \pi^{gz}_{e+1} x^g \quad \forall g\in\mathcal{P}_a,\, z\in\mathcal{Z}^g. \tag{16}$$

Hereafter, we will refer to constraints (16) as the EFV curve constraints. For the sake of simplicity, we will still denote the optimal objective value of this subproblem by $\sigma^a(\bar{x}^{\rho(a)})$. It is trivial to see that the value $\sigma^a(\bar{x}^{\rho(a)})$ does not increase when we enlarge the set $\mathcal{Z}^g$.

The algorithm adopts an SDP approach (Powell 2007; Ross 1995), where the approximation of the EFV curves is refined iteratively using recursion (11). Each iteration of the EFV decomposition algorithm consists of a *front-to-back* scheme followed by a *back-to-front* scheme. The front-to-back scheme is aimed at building a feasible solution $\bar{x}$ for the problem and checking whether it improves the incumbent solution x^*. Subproblems from stage 1 to stage $|\mathcal{E}|$ are solved, passing the obtained values of linking variables to the subproblems in the next stage. The back-to-front scheme is aimed at refining the EFV curves around the feasible solution $\bar{x}$ built in that iteration. Subproblems from stage $|\mathcal{E}|$ to stage 1 are solved, passing the refinement of the EFV curves to the subproblems in the previous stage. The algorithm will stop if the relative change in σ^1 between two consecutive iterations is below a tolerance parameter, $\epsilon > 0$, or an upper bound on the number of iterations, $N^{\max}_{\text{iter}}$, is reached. In each iteration, the sets $\mathcal{Z}^g$ are enlarged, and therefore the value σ^1 is nonincreasing. Therefore, with this stopping criteria, the EFV decomposition algorithm will eventually stop.

A flowchart of the entire EFV decomposition algorithm can be found in Figure 4, including the initialization and both the front-to-back and the back-to-front schemes in which each iteration is split. In the next section we discuss in detail the refinement of the EFV curves, i.e., how to obtain the set of new reference levels and the parameters defining the linear approximations associated with the new reference levels.

2.4. Approximating the EFV Curves

In this section we will explain how the approximation of the EFV curve $\lambda^g(\cdot)$ in stage e, $g\in\mathcal{P}_a$, $a\in\mathcal{A}^e$, is refined in a back-to-front scheme. We first describe how we obtain the set of new reference levels, $\mathcal{Z}^g_{\text{aux}}$, to be added to $\mathcal{Z}^g$. We then derive the parameters μ^{gz}_{e+1} and π^{gz}_{e+1} defining the EFV curve constraints (16) for each $z\in\mathcal{Z}^g_{\text{aux}}$.

Recall that $\bar{x}^g$ denotes the values of x^g obtained in the front-to-back scheme. The set $\mathcal{Z}^g_{\text{aux}}$ will contain $\bar{x}^g$ as well as small perturbations of $\bar{x}^g$.

To derive the parameters μ^{gz}_{e+1} and π^{gz}_{e+1} in (16) for $z\in\mathcal{Z}^g_{\text{aux}}$, we use an ad hoc sensitivity analysis of $\sigma^{a'}(\cdot)$, $a'\in\mathcal{S}_g\cap\mathcal{A}^{e+1}$. For each $a'\in\mathcal{S}_g\cap\mathcal{A}^{e+1}$, we solve $\sigma^{a'}(z)$. Let $\pi^{a'z}$ be the dual vector of constraint (15). Because of the concavity of this subproblem, we have

$$\sigma^{a'}(\cdot) \le \sigma^{a'}(z) + \pi^{a'z}(\cdot - z).$$

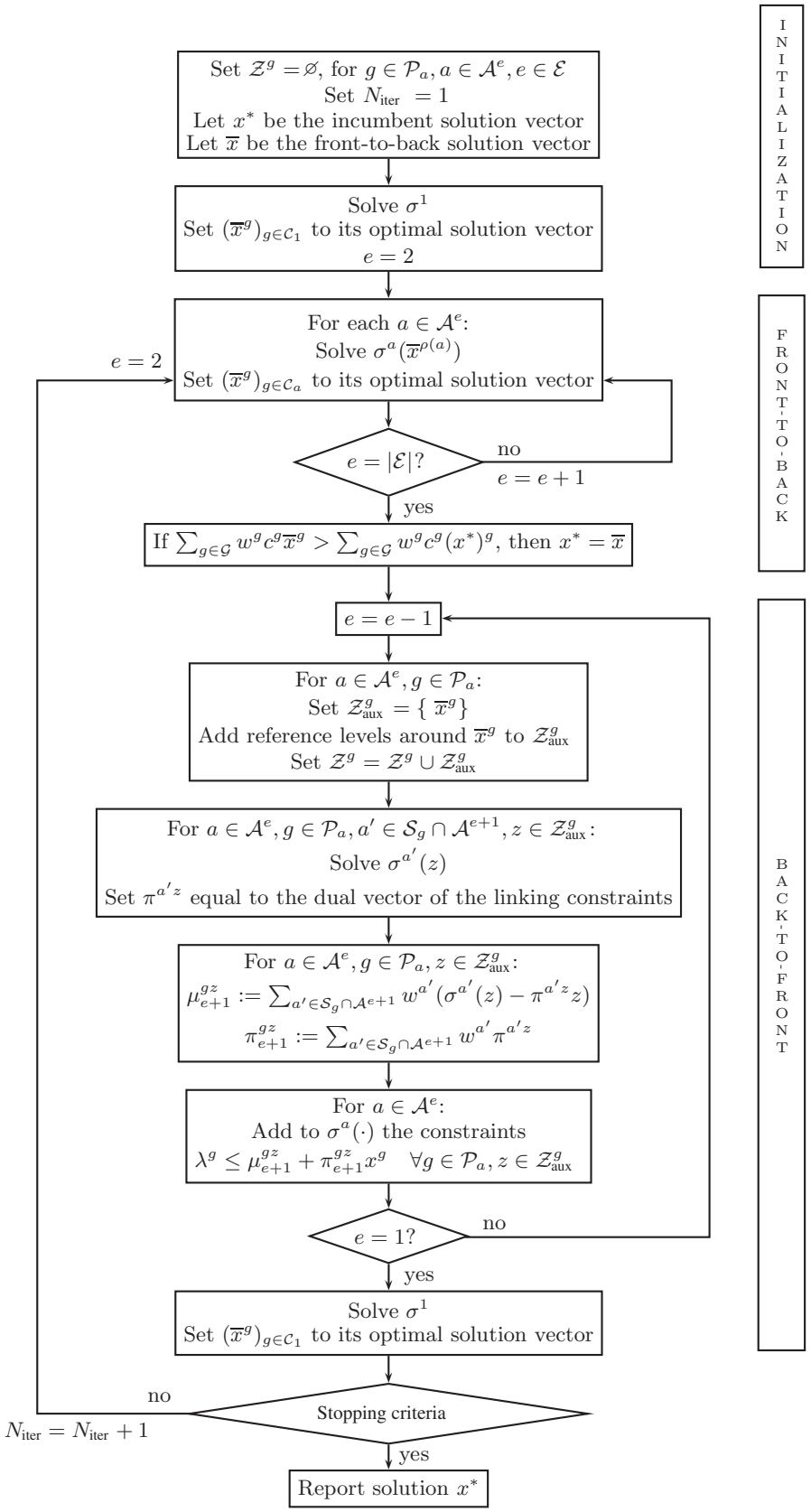

Figure 4 Flowchart of the EFV Decomposition Algorithm

Using this together with Formula (11), we set

$$\mu_{e+1}^{gz} := \sum_{a' \in \mathcal{S}_g \cap \mathcal{A}^{e+1}} w^{a'}(\sigma^{a'}(z) - \pi^{a'z}z) \quad \text{and}$$

$$\pi_{e+1}^{gz} := \sum_{a' \in \mathcal{S}_g \cap \mathcal{A}^{e+1}} w^{a'}\pi^{a'z};$$

see Cristóbal, Escudero, and Monge (2009) for more details. (Note that because the nonlinking components of x^g do not affect the objective function value of future subproblems, they have zero coefficient in $\pi^{a'z}$ and subsequently zero coefficient in π_{e+1}^{gz}.)

Once all the EFV curves in stage e are refined, we move to the previous stage, with the aim of refining the ones in that stage.

3. The Network Revenue Management Problem

As discussed in the introduction, two multiperiod SP models have been proposed for network revenue management in Möller, Römisch, and Weber (2008) and DeMiguel and Mishra (2008), respectively. In the following we give the formulations of both models.

The following notation is used to describe the network revenue management problem:

Sets:
$\mathcal{L}$, set of resources (with size L).
$\mathcal{I}$, set of bundles (with size I).
$\mathcal{J}$, set of fare classes (with size J).
$\mathcal{I}_l$, set of bundles using resource l, for $l \in \mathcal{L}$.

Deterministic Parameters:
f_{ij}, fare of bundle-class ij, for $i \in \mathcal{I}$, $j \in \mathcal{J}$.
C_l, capacity on resource l, for $l \in \mathcal{L}$.

Uncertain Parameters:
d_{ij}^g, demand for bundle-class ij in period $t(g)$ at node g, for $i \in \mathcal{I}$, $j \in \mathcal{J}$, $g \in \mathcal{G}$.

Decision Variables:
b_{ij}^g, number of accepted bookings for bundle-class ij in period $t(g)$ at node g, for $i \in \mathcal{I}$, $j \in \mathcal{J}$, $g \in \mathcal{G}$.
B_{ij}^g, cumulative number of accepted bookings of bundle-class ij along the path from the root to node g, for $i \in \mathcal{I}$, $j \in \mathcal{J}$, $g \in \mathcal{G}$.
$P_{ij}^{\rho(g)}$, protection level of bundle-class ij set at node $\rho(g)$ for cumulative accepted bookings along the path from the root to node g, for $i \in \mathcal{I}$, $j \in \mathcal{J}$, $g \in \mathcal{G}$. (Notice that all the nodes with the same immediate ancestor share the same protection level.)

Note that Möller, Römisch, and Weber (2008) model cancelations, whereas DeMiguel and Mishra (2008) do not. Because we will be comparing their models, we have chosen not to incorporate cancelations either.

3.1. The Model with Protection Levels

A stochastic programming model with protection levels and satisfying the nonanticipativity constraints was proposed in Möller, Römisch, and Weber (2004, 2008). The DEM formulation of the model is the following:

$$\text{maximize} \quad \sum_{g \in \mathcal{G}} w^g \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} f_{ij} b_{ij}^g \tag{17}$$

$$\text{s.t.} \quad B_{ij}^g = B_{ij}^{\rho(g)} + b_{ij}^g \quad \forall g \in \mathcal{G}, i \in \mathcal{I}, j \in \mathcal{J}, \tag{18}$$

$$B_{ij}^g \le P_{ij}^{\rho(g)} \quad \forall g \in \mathcal{G}, i \in \mathcal{I}, j \in \mathcal{J}, \tag{19}$$

$$\sum_{i \in \mathcal{I}_l} \sum_{j \in \mathcal{J}} P_{ij}^{\rho(g)} \le C_l \quad \forall g \in \mathcal{G}_T, l \in \mathcal{L}, \tag{20}$$

$$0 \le b_{ij}^g \le d_{ij}^g \quad \forall g \in \mathcal{G}, i \in \mathcal{I}, j \in \mathcal{J}, \tag{21}$$

where $\mathcal{G}_T$ is the set of nodes in the last period of the time horizon, T. The objective function (17) maximizes the expected revenue across the time horizon. Constraints (18) define the booking balance equations and constraints (19) ensure that the cumulative number of accepted bookings along the path from the root to node g cannot exceed the protection level set at the ancestor node $\rho(g)$. The protection levels across bundles and class fares are then bounded by the capacity on the resources in constraints (20). Constraints (21) reflect that the number of accepted bookings should be not greater than the demand. Notice that the nonanticipativity constraints are satisfied by construction. We will refer to this model as DEM^P.

Note that in DEM^P, the only linking variables are B_{ij}^g because they determine the remaining capacity on the resources. It is easy to see that B_{ij}^g is only present in constraints associated with its own period, $t(g)$, and the next one, as it happens in many planning problems.

3.2. The Model Without Protection Levels

A model partially violating the nonanticipativity principle was proposed in DeMiguel and Mishra (2008). The DEM formulation of the model is the following:

$$\text{maximize} \quad \sum_{g \in \mathcal{G}} w^g \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} f_{ij} b_{ij}^g \tag{22}$$

$$\text{s.t.} \quad B_{ij}^g = B_{ij}^{\rho(g)} + b_{ij}^g \quad \forall g \in \mathcal{G}, i \in \mathcal{I}, j \in \mathcal{J}, \tag{23}$$

$$\sum_{i \in \mathcal{I}_l} \sum_{j \in \mathcal{J}} B_{ij}^g \le C_l \quad \forall g \in \mathcal{G}_T, l \in \mathcal{L}, \tag{24}$$

$$0 \le b_{ij}^g \le d_{ij}^g \quad \forall g \in \mathcal{G}, i \in \mathcal{I}, j \in \mathcal{J}. \tag{25}$$

We will refer to this model as DEM^N. Notice that here constraints (24), imposing that the total number of accepted bookings over the whole booking horizon

is restricted by the capacity on the resources, replace constraints (19) and (20) in DEM^P. As a consequence, DEM^N has fewer decision variables and constraints than DEM^P does.

Contrary to DEM^P, in which the same protection level is chosen for all successor nodes, in DEM^N different allocations can be chosen for different successors. In other words, DEM^N assumes perfect information on which realization of demand will happen in the next period.

4. EFV Based Nonadditive Bid Prices

Bid-price policies (Talluri and van Ryzin 1998) are a state-of-the-art approach to the network revenue management problem, in which a price is generated for each bundle and a request to purchase the bundle is accepted if and only if the associated revenue exceeds the bid price. Most of the literature focus on additive bid prices, in which a bid price is generated for each resource and the bid price of a bundle is defined as the sum of the bid prices of all resources used by the bundle. Despite the benefits of nonadditive bid prices (Bertsimas and Popescu 2003), the work in this area is scarce. In this section, we propose to use the EFV decomposition algorithm introduced in §2 to derive nonadditive bid prices for network revenue management.

After the EFV decomposition algorithm stops, the unique subproblem in stage 1, σ^1, approximates the network revenue management problem. Recall that in this subproblem the EFV curves are approximated by piecewise linear and concave functions, which are modeled by the EFV curve constraints. We derive nonadditive bid prices from the dual information of the EFV curve constraints. This will be done for both the formulation with and without protection levels.

We first write down the unique subproblem in stage 1 for both DEM^P and DEM^N. Throughout the rest of the paper, we will refer to them as EFV^P and EFV^N. Let us first look at the subproblem in stage 1 for the protection levels formulation. Because the weight of the root node is equal to 1, this problem can be written as

$$\text{maximize} \quad \left\{ \sum_{g\in\mathcal{G}^1} w^g \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} f_{ij} b_{ij}^g + \sum_{g\in\mathcal{P}_1} \lambda^g \right\} \tag{26}$$

$$\text{s.t.} \quad B_{ij}^g = B_{ij}^{\rho(g)} + b_{ij}^g \quad \forall\, g\in\mathcal{G}^1,\ i\in\mathcal{I},\ j\in\mathcal{J}, \tag{27}$$

$$B_{ij}^g \le P_{ij}^{\rho(g)} \quad \forall\, g\in\mathcal{G}^1,\ i\in\mathcal{I},\ j\in\mathcal{J}, \tag{28}$$

$$\sum_{i\in\mathcal{I}_l}\sum_{j\in\mathcal{J}} P_{ij}^{\rho(g)} \le C_l \quad \forall\, g\in\mathcal{P}_1,\ l\in\mathcal{L}, \tag{29}$$

$$0 \le b_{ij}^g \le d_{ij}^g \quad \forall\, g\in\mathcal{G}^1,\ i\in\mathcal{I},\ j\in\mathcal{J}, \tag{30}$$

$$\lambda^g \le \mu_2^{gz} + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \pi_{2,ij}^{gz} B_{ij}^g, \quad \forall\, g\in\mathcal{P}_1,\ z\in\mathcal{Z}^g. \tag{31}$$

Similarly, for the formulation without protection levels, the subproblem in stage 1 reads as follows

$$\text{maximize} \quad \left\{ \sum_{g\in\mathcal{G}^1} w^g \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} f_{ij} b_{ij}^g + \sum_{g\in\mathcal{P}_1} \lambda^g \right\} \tag{32}$$

$$\text{s.t.} \quad B_{ij}^g = B_{ij}^{\rho(g)} + b_{ij}^g \quad \forall\, g\in\mathcal{G}^1,\ i\in\mathcal{I},\ j\in\mathcal{J}, \tag{33}$$

$$\sum_{i\in\mathcal{I}_l}\sum_{j\in\mathcal{J}} B_{ij}^g \le C_l \quad \forall\, g\in\mathcal{P}_1,\ l\in\mathcal{L}, \tag{34}$$

$$0 \le b_{ij}^g \le d_{ij}^g \quad \forall\, g\in\mathcal{G}^1,\ i\in\mathcal{I},\ j\in\mathcal{J}, \tag{35}$$

$$\lambda^g \le \mu_2^{gz} + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \pi_{2,ij}^{gz} B_{ij}^g, \quad \forall\, g\in\mathcal{P}_1,\ z\in\mathcal{Z}^g. \tag{36}$$

Recall that (31) and (36) are the so-called EFV curve constraints for the formulation with protection levels and without them, respectively.

Before the bid prices are defined, there are two useful observations to be made for either formulation. For a given EFV curve constraint, the coefficient $\pi_{2,ij}^{gz}$ of decision variable B_{ij}^g can be seen as an estimation of the future effect in revenue of a unit increase of the cumulated bookings B_{ij}^g. Let $\Gamma^{gz} \ge 0$ be the dual variable of the EFV curve constraint of $g\in\mathcal{P}_1$, $z\in\mathcal{Z}^g$. Using duality theory, we know that $\sum_{z\in\mathcal{Z}^g}\Gamma^{gz} = 1$, whereas Γ^{gz} will be equal to 0 when the EFV curve constraint is not binding. Therefore, for a given g, Γ^{gz} can be seen as a measure of the importance of the binding EFV curve constraints.

In both formulations, once the corresponding stage 1 subproblem is solved, the bid price for bundle i is defined as

$$\max\left\{ \sum_{g\in\mathcal{P}_1}\sum_{z\in\mathcal{Z}^g} \Gamma^{gz}\pi_{2,ij}^{gz} \colon j\in\mathcal{J} \right\}, \tag{37}$$

where we combine the slopes $\pi_{2,ij}^{gz}$ given by all the EFV curve constraints of leaf nodes in stage 1, using the dual variables Γ^{gz}. Clearly, the bid prices in (37) are calculated directly for each bundle and thus are not additive, in general.

In the following, we investigate the relationship between our bid prices and the slopes of the piecewise linear and concave approximation of the EFV curves of the leaf nodes in stage 1. From the definition of $\pi_{2,ij}^{gz}$ given in §2.4, $\pi_{2,ij}^{gz} := \sum_{a\in\mathcal{S}_g\cap\mathcal{M}^{e+1}} w^a \pi_{ij}^{az}$, the term being maximized in (37) can be rewritten as

$$\begin{aligned}\sum_{g\in\mathcal{P}_1}\sum_{z\in\mathcal{Z}^g}\Gamma^{gz}\pi_{2,ij}^{gz} &= \sum_{g\in\mathcal{P}_1}\sum_{z\in\mathcal{Z}^g}\Gamma^{gz}\sum_{a\in\mathcal{S}_g\cap\mathcal{M}^{e+1}} w^a\pi_{ij}^{az}\\ &= \sum_{g\in\mathcal{P}_1}\sum_{a\in\mathcal{S}_g\cap\mathcal{M}^{e+1}} w^a \sum_{z\in\mathcal{Z}^g}\Gamma^{gz}\pi_{ij}^{az}\\ &= \sum_{g\in\mathcal{P}_1}\sum_{a\in\mathcal{S}_g\cap\mathcal{M}^{e+1}}\sum_{z\in\mathcal{Z}^g} w^a\Gamma^{gz}\pi_{ij}^{az},\end{aligned}$$

which is a weighted average of the slopes π_{ij}^{az} because $\sum_{g\in\mathcal{P}_1}\sum_{a\in\mathcal{S}_g\cap\mathcal{M}^{e+1}}\sum_{z\in\mathcal{Z}^g} w^a\,\Gamma^{gz} = 1$.

5. Computational Experience

5.1. Overview

Our computational experience consists of two parts. The first part focuses on the performance of EFV as a decomposition algorithm, which includes testing its solution accuracy and computation time compared to DEM. The second part focuses on the revenue performance of the nonadditive bid prices generated by EFV, which consists of running a rolling horizon simulation (DeMiguel and Mishra 2008; Talluri and van Ryzin 1999) and comparing the revenues generated by EFV with state-of-the-art additive approaches, including additive bid prices derived from DEM.

There are two sets of parameters associated with the EFV decomposition algorithm, relating to the stopping criteria and the generation of reference levels in each iteration. Recall that the EFV decomposition algorithm will stop if the relative change in σ^1 between two consecutive iterations is below a tolerance parameter $\epsilon > 0$ or an upper bound on the number of iterations, $N_{\text{iter}}^{\max}$, is reached. Throughout our numerical experiments, we set $\epsilon = 0.001$ and $N_{\text{iter}}^{\max} = 15$. As the results in §5.3 will show, the limit in the number of iterations is not binding. In the back-to-front scheme, new reference levels are generated by perturbing the solution vector from the front-to-back scheme with $k\xi$, where ξ is the unit vector. (Note that if a component of the new vector becomes negative, that component is not perturbed.) In our computational experiments we have chosen $k = -2, -1, 1, 2, 3$.

We use the optimization engine CPLEX v11.0 (IBM ILOG 2007) for solving the LP problems arising. Our experiments were conducted on a PC with a 2.33 GHz Intel Xeon dual core processor, 8.5 GB of RAM, and operating system LINUX Debian 4.0.

The remainder of the section is organized as follows. We will describe the test networks and the demand model in §5.2. The results on solution accuracy and computation time will be given in §5.3. Finally, we present the results on revenue performance in §5.4.

5.2. The Test Networks and Demand Model

In this numerical study, we consider a small network, a medium, and a large one. Next, we will introduce first the dimensions and then the structure of our test networks. Finally, we will discuss the fares and the demand model.

Recall that the dimensions of a network are given by its number of bundles I, number of fare classes J, and number of resources L. We also include here the capacity on the resources C_l. The dimensions of our test networks can be found as follows:

Small network. $I = 18,\ J = 4,\ L = 10,\ C_l = 200\ \forall\, l \in \mathcal{L}$.

Medium network. $I = 200,\ J = 4,\ L = 100,\ C_l = 200\ \forall\, l \in \mathcal{L}$.

Large network. $I = 1{,}000,\ J = 4,\ L = 500,\ C_l = 200\ \forall\, l \in \mathcal{L}$.

In terms of network structure, we follow the standard practice in the literature to use randomly generated networks with a hub-and-spoke structure (Bertsimas and Popescu 2003; DeMiguel and Mishra 2008; Talluri and van Ryzin 1999), resembling networks seen in hub-based airlines. Among the resources, the first half are spoke-to-hub flight legs and the second half are hub-to-spoke flight legs. The bundles are generated as follows. In the Small network, the first 10 bundles each use one of the 10 legs, and the remaining 8 each use two random legs, **spoke1-to-hub** and **hub-to-spoke2**. In the Medium network, the first 100 bundles each use one of the 100 legs; the next 75 each use two random legs **spoke1-to-hub** and **hub-to-spoke2**; and finally the last 25 each use four random legs **spoke1-to-hub**, **hub-to-spoke2**, **spoke2-to-hub**, and **hub-to-spoke1**. Similarly, in the Large network, the first 500 bundles each use one of the 500 legs, the next 375 each use two random legs, and finally the last 125 each use four random legs.

We now present the way the fares and the demands have been generated. The class 1 fare of a single-resource bundle is generated from a normal distribution with mean 100 and standard deviation 40, truncated within the interval [20, 180]. The class 1 fare of a multi-resource bundle is the summation of the class 1 fares of the single-resource bundles associated with its resources. For all bundles, the fare of class 2 is 1.4 times that of class 1, the fare of class 3 is 4 times, and the fare of class 4 is 4.4 times. As usual in the literature, demands are generated using a Poisson distribution. (We have also used a truncated normal distribution for the demands, as in DeMiguel and Mishra 2008, and the results obtained are very similar.) The demand for bundle-class ij in period t is generated from a Poisson distribution with mean $\mu_{ijt} := \beta_{ijt}\mu$, where $\mu = \eta((\sum_{l\in\mathcal{L}} C_l)/(I \times J \times T))$. Because $(\sum_{l\in\mathcal{L}} C_l)/(I \times J \times T)$ is a constant, the higher η, the higher the demand. Therefore, we refer to η as the "load factor of demand." With respect to β_{ijt}, we have chosen them higher for single-resource bundles than for multi-resource ones. These parameters are also dynamic with respect to the time-to-service. For $j = 1, 2$, β_{ijt} decreases when getting closer to the departure, eventually becoming zero. For $j = 3, 4$, this pattern is reversed.

5.3. Results on Solution Accuracy and Computation Time

In this section, we will compare the EFV and the DEM approaches in terms of solution accuracy and computation time. We have generated six problem

Table 1 DEM Dimensions of Problem Instances on the Three Test Networks

					DEM^P				DEM^N			
ID	Tree	T	$\lvert\mathcal{G}\rvert$	$\lvert\Omega\rvert$	*n*	*m*	*nel*	*dens*	*n*	*m*	*nel*	*dens*
							SMALL					
1	$2^2 2^2 2^2$	6	64	32	11,376	9,392	24,272	0.02272	9,072	4,856	15,200	0.03450
2	$3^2 3^2 3^2$	6	365	243	61,200	54,846	143,129	0.00426	52,416	28,638	90,713	0.00604
3	$2^3 2^2 2^2$	7	128	64	22,896	18,928	48,904	0.01128	18,288	9,784	30,616	0.01711
4	$3^3 3^2 3^2$	7	1,094	729	183,672	1,64,682	429,748	0.00142	157,392	85,986	272,356	0.00201
5	$2^2 2^2 2^2 2^2$	8	256	128	45,936	38,000	98,168	0.00562	36,720	19,640	61,448	0.00852
6	$3^2 3^2 3^2 3^2$	8	3,281	2,187	551,088	494,190	1,289,603	0.00047	472,320	258,030	817,283	0.00067
							MEDIUM					
7	$2^2 2^2 2^2$	6	64	32	126,400	104,000	270,400	0.00206	100,800	53,600	169,600	0.00314
8	$3^2 3^2 3^2$	6	365	243	680,000	606,700	1,595,725	0.00039	582,400	315,500	1,013,325	0.00055
9	$2^3 2^2 2^2$	7	128	64	254,400	209,600	544,800	0.00102	203,200	108,000	341,600	0.00156
10	$3^3 3^2 3^2$	7	1,094	729	2,040,800	1,821,700	4,791,175	0.00013	1,748,800	947,300	3,042,375	0.00018
11	$2^2 2^2 2^2 2^2$	8	256	128	510,400	420,800	1,093,600	0.00051	408,000	216,800	685,600	0.00078
12	$3^2 3^2 3^2 3^2$	8	3,281	2,187	6,123,200	5,466,700	14,377,525	0.00004	5,248,000	2,842,700	9,129,525	0.00006
							LARGE					
13	$2^2 2^2 2^2$	6	64	32	632,000	520,000	1,352,000	0.00041	504,000	268,000	848,000	0.00063
14	$3^2 3^2 3^2$	6	365	243	3,400,000	3,033,500	7,978,625	0.00008	2,912,000	1,577,500	5,066,625	0.00011
15	$2^3 2^2 2^2$	7	128	64	1,272,000	1,048,000	2,724,000	0.00020	1,016,000	540,000	1,708,000	0.00031
16	$3^3 3^2 3^2$	7	1,094	729	10,204,000	9,108,500	23,955,875	0.00003	8,744,000	4,736,500	15,211,875	0.00004
17	$2^2 2^2 2^2 2^2$	8	256	128	2,552,000	2,104,000	5,468,000	0.00010	2,040,000	1,084,000	3,428,000	0.00016
18	$3^2 3^2 3^2 3^2$	8	3,281	2,187	30,616,000	27,333,500	71,887,625	0.00001	26,240,000	14,213,500	45,647,625	0.00001

instances for each test network with a load factor of demand $\eta = 2$. The scenario tree is generated through random period-by-period demand sampling based on our demand model.

Table 1 describes the scenario tree and gives the dimensions of both DEM formulations for each problem instance. The first column of the table assigns an identifier to each problem instance. The following four columns focus on the scenario tree, reporting the predefined structure Tree, the number of periods T, the number of scenarios $|\Omega|$, and the number of nodes $|\mathcal{G}|$. Column Tree displays the structure of the tree in the form $A_1^{B_1} A_2^{B_2} \cdots$, where A_i denotes the number of children each node in stage i has and B_i denotes the number of periods in stage i. For instance, the structure $3^2 3^2 3^2 3^2$ means the tree has four stages, two periods in each stage, and each node has three children. The rest of the columns show the dimensions of DEM^P and DEM^N, including the number of decision variables n, number of constraints m, number of nonzero elements in the constraint matrix *nel*, and constraint matrix density $dens := (nel/(n \times m))$ (in %).

Table 2 shows the objective value and computation time of each of the four approaches, DEM^P, DEM^N, EFV^P, and EFV^N, as well as information on the EFV decomposition algorithm. The headings are as follows. For each of the four formulations, *objval* denotes the objective value, and t_{LP} the elapsed time (in seconds) to solve the problem (where a maximum of 7,200 seconds has been imposed). For the EFV decomposition algorithm, we have the following notation: N_{iter} is the number of iterations; N_z is the total number of reference levels generated; N_{LP} is the total number of LP subproblems solved; and $error = (objval_{\mathrm{DEM}} - objval_{\mathrm{EFV}})/objval_{\mathrm{DEM}}$, where $objval_{\mathrm{DEM}}$ and $objval_{\mathrm{EFV}}$ are the objective values of the DEM and the EFV formulations, respectively, is the relative error of the solution found by the EFV decomposition algorithm (in %). Note that because we have imposed a limit on the elapsed time, this error may be negative, indicating that the EFV decomposition algorithm obtains a better solution than DEM within the time limit. For each test network, the last row shows average figures across all problem instances. All figures in Table 2 have been rounded to the closest integer, except for the *error* figures.

The problem instances generated in this paper are challenging because of their large scale, especially in the protection levels formulation for the MEDIUM and the LARGE networks. In the largest problem instance in the MEDIUM network, problem instance 12, DEM^P contains roughly 6.12 million decision variables and 5.47 million constraints, whereas DEM^N contains roughly 5.25 million decision variables and 2.84 million constraints. In the largest problem instance in the LARGE network, problem instance 18, DEM^P contains roughly 30.62 million decision variables and 27.33 million constraints, and DEM^N contains roughly 26.24 million decision variables and 14.21 million constraints.

Nevertheless, our algorithm can handle these sizes appropriately because of its decomposition nature. In order to illustrate this, and for each test network,

Table 2 **Solution Accuracy and Computation Time of DEM^P, EFV^P, DEM^N, and EFV^N on the Three Test Networks**

	DEM^P		EFV^P						DEM^N		EFV^N					
ID	*objval*	t_{LP}	*objval*	t_{LP}	N_{iter}	N_z	N_{LP}	*error*	*objval*	t_{LP}	*objval*	t_{LP}	N_{iter}	N_z	N_{LP}	*error*
								SMALL								
1	553,948	0	549,316	1	5	600	715	0.84	563,773	0	557,498	1	7	840	1,001	1.11
2	546,082	4	544,305	9	9	4,860	5,697	0.33	558,755	1	553,423	5	6	3,240	3,798	0.95
3	554,036	1	547,113	2	7	1,680	1,981	1.25	561,720	0	556,674	2	6	1,440	1,698	0.90
4	549,614	42	545,502	17	7	11,340	13,251	0.75	560,134	11	555,307	14	6	9,720	11,358	0.86
5	555,021	3	545,967	4	6	3,024	3,546	1.63	562,240	1	552,383	4	10	5,040	5,910	1.75
6	551,905	362	546,759	55	7	34,398	40,152	0.93	561,170	69	553,244	59	8	39,312	45,888	1.41
Average		69		15	7	9,317	10,890	0.95		14		14	7	9,932	11,609	1.17
								MEDIUM								
7	6,230,716	7	6,183,753	12	4	480	572	0.75	6,328,489	3	6,274,919	9	4	480	572	0.85
8	6,136,546	191	6,069,588	31	2	1,080	1,266	1.09	6,265,062	47	6,222,750	61	4	2,160	2,532	0.68
9	6,230,361	28	6,182,603	23	4	960	1,132	0.77	6,314,243	9	6,254,620	24	5	1,200	1,415	0.94
10	6,168,710	2,205	6,065,774	88	2	3,240	3,786	1.67	6,277,748	456	6,222,254	78	2	3,240	3,786	0.88
11	6,255,828	108	6,204,048	52	4	2,016	2,364	0.83	6,328,098	28	6,280,298	67	6	3,024	3,546	0.76
12	5,428,674	7,200	6,160,114	257	2	9,828	11,472	−13.47	6,289,127	5,733	6,252,684	480	4	19,656	22,944	0.58
Average		1,623		77	3	2,934	3,432	−1.39		1,046		119	4	4,960	5,799	0.78
								LARGE								
13	30,246,340	90	29,908,287	225	4	480	572	1.12	30,734,651	27	30,374,186	159	5	600	715	1.17
14	30,411,195	3,528	30,022,214	439	2	1,080	1,266	1.28	31,051,132	618	30,823,280	918	4	2,160	2,532	0.73
15	30,588,133	384	30,353,457	425	4	960	1,132	0.77	31,021,183	115	30,749,329	374	4	960	1,132	0.88
16	25,758,210	7,200	30,172,475	1,330	2	3,240	3,786	−17.14	29,504,661	7,200	30,880,457	2,576	4	6,480	7,572	−4.66
17	30,661,929	1,875	30,463,966	280	2	1,008	1,182	0.65	31,026,054	320	30,860,622	532	4	2,016	2,364	0.53
18	21,972,718	7,200	30,528,955	3,294	2	9,828	11,472	−38.94	26,291,930	7,200	31,034,607	6,561	4	19,656	22,944	−18.04
Average		3,379		999	3	2,766	3,235	−8.71		2,580		1,853	4	5,312	6,210	−3.23

*Time limit of 7,200 seconds.

we first discuss the worst case scenario for the computation time and then its average case. For the sake of simplicity, we discuss the protection levels formulation, but similar conclusions can be drawn for the formulation without protection levels. Note that for each test network the worst case scenario in terms of computation time coincides with the largest problem instance. Obviously, in the SMALL network the differences between DEM^P and EFV^P are mild. The computation time of EFV^P in the largest problem instance is 55 seconds, whereas that of DEM^P is 362 seconds. In terms of average time, EFV^P takes 15 seconds, and DEM^P 69 seconds. The advantage of the EFV decomposition algorithm is more notorious in the MEDIUM and the LARGE networks. In the MEDIUM network, the computation time of EFV^P in the largest problem instance is 257 seconds, whereas DEM^P reaches the time limit of 7,200 seconds. In terms of average time, EFV^P takes 77 seconds, and DEM^P 1,623 seconds. In the LARGE network, the computation time of EFV^P in the largest problem instance is 3,294 seconds, whereas DEM^P again reaches the time limit of 7,200 seconds. In terms of average time, EFV^P takes 999 seconds, and DEM^P 3,379 seconds, where there are two problem instances out of six reaching the time limit. Finally, the computation time of DEM is frequently of one order of magnitude higher than that of EFV.

Compared with DEM, the EFV decomposition algorithm trades solution accuracy for computation time. From Table 2, we have that the average *error* of EFV^P and EFV^N in the SMALL network is 0.95% and 1.17%, respectively. In the MEDIUM network, the corresponding figures are −1.39% and 0.78%, where there is one problem instance in which the solution found by DEM^P after 7,200 seconds is worse than the one found by EFV^P, $error = -13.47\%$. Finally, the average *error* of EFV^P and EFV^N in the LARGE network is −8.71% and −3.23%, respectively. For this test network, and for both formulations, there were two problem instances for which EFV found a better solution than DEM within the time limit. For the formulation with protection levels, $error = -17.14\%, -38.94\%$, whereas for the one without protection levels, $error = -4.66\%, -18.04\%$. Given the scale of the problem instances, these results suggest that the EFV decomposition algorithm generally achieves good solution accuracy.

5.4. Results on Revenue Performance

In this section, we present results on the revenue performance of our nonadditive bid prices and compare them with the revenues generated by state-of-the-art additive approaches. In total, we present revenue results on seven methods. We test nonadditive bid prices derived from both EFV^P and EFV^N. As benchmarking, we use the revenues generated by the additive bid prices derived from DEM^P and DEM^N using the dual variables of the resource constraints. We also test additive bid prices derived from the popular single-period models; DLP (Wong, Koppelman, and Daskin 1993), RLP (Talluri and van Ryzin 1998), and the perfect hindsight (PH) model are tested as well. PH consists of solving a DLP using the actual demand instead of mean demand and its optimal value is used as an upper bound of the achievable revenue; see Talluri and van Ryzin (1998) and DeMiguel and Mishra (2008).

As usual in the literature, we use a rolling horizon simulation to test the revenues of the different approaches; see, e.g., DeMiguel and Mishra (2008) and Talluri and van Ryzin (1999). Next, we describe the details of the simulation. We end the section with the discussion on the reported revenues.

5.4.1. Rolling Horizon Simulation. The input data to the simulation is the length of the simulation horizon, denoted by T^s, and the sequence of booking requests. These booking requests are put into a simulated sales process controlled by a bid-price policy. The simulation is carried out on a rolling-horizon basis. At the beginning of period $t^s \in \{1, \ldots, T^s\}$, bid prices are calculated using the corresponding model, which is then used to decide which booking requests to accept in period t^s. At the end of period t^s, the remaining capacity is updated. This process is repeated for $t^s = 1, \ldots, T^s$.

In our experiments, we have set $T^s = 6$. The booking requests are generated as follows. Using our demand model, we generate the vector of (actual) demands over the simulation horizon. For each period, the total number of booking requests for each bundle-class combination is equal to the corresponding actual demand, and the bookings for different bundle-class combinations arrive in random order.

In terms of the input data used by each model, in simulation period t^s, PH uses the actual demand in period t^s, whereas DLP uses the (theoretical) mean demand μ^s_{ijt}. For the rest of tested models, demand scenarios are needed. At the beginning of each simulation period t^s, a scenario tree is generated, with time horizon equal to the remaining simulation horizon, i.e., $T = T^s + 1 - t^s$, and two children per node. The demands in the scenario tree are generated through random period-by-period demand sampling based on our demand model. To ensure a fair comparison, DEM^P, DEM^N, EFV^P, and DEM^N use the same scenario tree, whereas RLP uses the demand scenarios defined by the scenario tree.

5.4.2. Revenue Performance. As discussed in §5.2, the demand model has one parameter: the load factor η. Extreme values of demand load will yield very

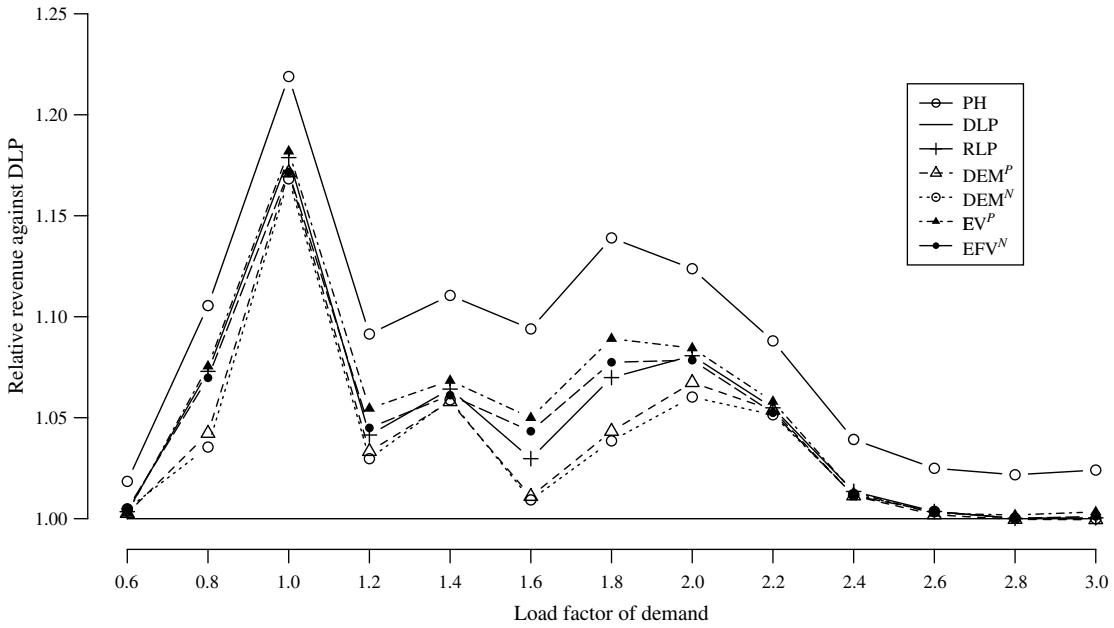

Figure 5 The Relative Revenues of the Seven Methods in the Small Network

similar revenues for all models. If demand is too low, all methods will simply accept all demand requests, whereas if demand is too high, all methods will accept demand requests for the very profitable bundles only. To cover an interesting range of load factors, we have chosen $\eta \in \{0.6, 0.8, \ldots, 3.0\}$.

For each value of η, we perform 100 simulation runs, each run with a different random realization of the demand. For each method, we calculate the total revenue across the 100 runs and report its relative total revenue against that of DLP. Figures 5–7 plot the relative revenues against η for each test network. To help with the discussion on revenue performance, Table 3 shows the average relative revenue across the entire range of η, denoted by AvgRev, for each test network.

Throughout the three plots, the following two observations hold. First, the curve of PH is much higher than the rest, confirming that PH gives a very loose upper bound of the achievable revenue. Second, the relative revenues of all methods, except for DLP, are above one for all values of η in the Medium and the Large networks. Thus in these two test networks, DLP is outperformed by the other six methods. For

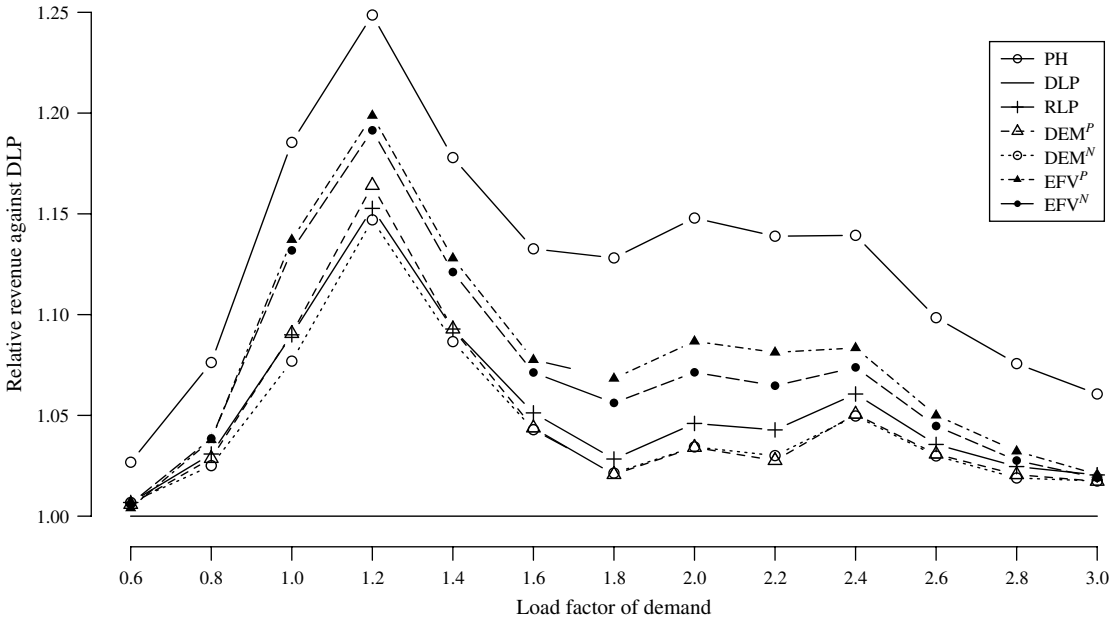

Figure 6 The Relative Revenues of the Seven Methods in the Medium Network

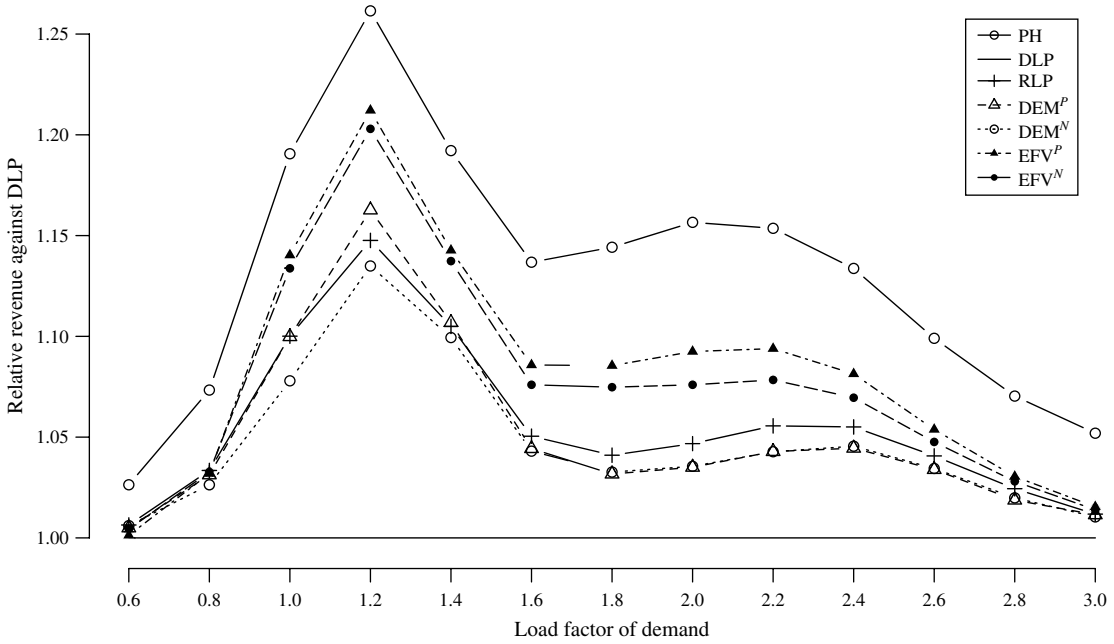

Figure 7 The Relative Revenues of the Seven Methods in the Large Network

the Small network, this is true for 11 values of η out of the 13 we have chosen. For $\eta = 2.8$, both DEM^P and DEM^N have a relative revenue slightly below one, whereas for $\eta = 3.0$, only DEM^N is slightly below DLP. Therefore, PH and DLP will not be included in our discussion below. The conclusions on relative performance of the remaining five methods are similar in the three test networks.

In terms of AvgRev, EFV^P is consistently the best method, closely followed by EFV^N. RLP ranks third, with a clear lower AvgRev in both the Medium and the Large networks. In the Medium network the respective AvgRev of EFV^P and RLP are 1.0774 and 1.0525, and in the Large network these figures are 1.0821 and 1.0553. RLP is closely followed by DEM^P and DEM^N, both having a very similar performance in the Small and the Medium networks. In the Small network the respective AvgRev of DEM^P and DEM^N are 1.0381 and 1.0363, and in the Medium network these figures are 1.0483 and 1.0452.

As said above, the five models yield similar revenues for extreme values of the demand load. From Figures 5–7, we can observe that the methods are very similar when either $\eta \leq 0.8$ or $\eta \geq 2.8$. If we exclude these four values and recalculate the average relative revenue, AvgRev^m, the dominance of EFV^P and EFV^N over the other three methods is even more significant for the Medium and the Large networks; see Table 4. In the Medium network, the respective AvgRev^m of EFV^P and RLP are 1.1013 and 1.0667. Thus, EFV^P yields an increase of 0.0346 over RLP in terms of AvgRev^m, whereas this increase is equal to 0.0249 when considering the full range of η. In the Large network, EFV^P yields an increase in AvgRev^m over RLP of 0.0384, whereas this increase is equal to 0.0268 when considering extreme values of the demand load.

For the Medium and the Large networks, the outperformance of EFV over RLP and DEM observed for the average relative revenue also holds for each value of $\eta \in \{1.0, 1.2, \ldots, 2.6\}$. For instance, in the Medium network and for $\eta = 1.8$, the respective relative revenue of EFV^P and EFV^N is 1.0683 and 1.0562, while that of RLP, DEM^P, and DEM^N is 1.0284, 1.0207, and 1.0214. In the Large network, the respective relative revenue of EFV^P and EFV^N is 1.0855 and 1.0748, whereas that of RLP, DEM^P, and DEM^N is 1.0410, 1.0317, and 1.0327. Similar conclusions can be given for other nonextreme values of η.

Table 3 Average Relative Revenues of the Seven Methods Across Full Range of η

Network	PH	DLP	RLP	DEM^P	DEM^N	EFV^P	EFV^N
Small	1.0846	1.0000	1.0472	1.0381	1.0363	1.0527	1.0477
Medium	1.1259	1.0000	1.0525	1.0483	1.0452	1.0774	1.0706
Large	1.1300	1.0000	1.0553	1.0514	1.0468	1.0821	1.0750

Table 4 Average Relative Revenues of the Seven Methods Across Middle Range of η

Network	PH	DLP	RLP	DEM^P	DEM^N	EFV^P	EFV^N
Small	1.1033	1.0000	1.0596	1.0501	1.0480	1.0669	1.0605
Medium	1.1553	1.0000	1.0667	1.0617	1.0576	1.1013	1.0918
Large	1.1631	1.0000	1.0714	1.0669	1.0607	1.1098	1.0996

6. Conclusions

The computation of bid prices for network revenue management along a time horizon has been considered in this paper. The uncertainty in demand is modeled by means of scenarios yielding an SP formulation. We proposed to solve two existing SP models from the literature using the EFV decomposition algorithm, which brings two main advantages. First, the EFV decomposition algorithm requires significantly less computation time than does the optimization of DEM by plain use of optimization engines. Problem instances with 72 pairs of bundle-fare classes have been solved in less than one minute, with 800 pairs in less than five minutes, and with 4,000 pairs in less than one hour. The time taken by DEM was, in general, of one order of magnitude higher, whereas for the three largest problem instances, and after two hours, the expected revenue returned by DEM was below that obtained by EFV by 13.47%, 17.14%, and 38.94%, respectively. Such difference in computation time makes EFV more useful than DEM in practice because bid prices usually need to be updated on a daily basis in real world revenue management systems, if not more frequently. Second, contrary to the traditional additive bid prices, the EFV approach is able to define nonadditive bid prices on bundles directly. Numerical results based on a rolling horizon simulation in three test networks show that in general, the nonadditive EFV bid prices give better revenues for middle-range values of the load factor of demand, whereas the differences among all the approaches we have tested are insignificant for extreme values.

Several future research directions are worth considering. First, it will be interesting to investigate the effect of the network structure on the revenue of nonadditive and additive bid prices and different methods for generating them. If we can understand exactly how network structure affects the relative performance of different methods, then in practice we will be able to choose the method that fits the network structure the best. Second, a parallel implementation of the EFV decomposition approach, in which the subproblems in each stage are solved in parallel, would allow handling larger sets of reference levels, ensuring even better objective values. Third, in addition to the EFV decomposition algorithm, scenario reduction techniques can be used to decrease the size of the problem instances being fed to the algorithm. Finally, it will be interesting to investigate the added value of nonadditive prices in more elaborated demand models where the customer is offered a set of products to choose from instead of a single one.

Acknowledgments

The authors thank the two anonymous referees and the associate editor for their helpful comments to improve both the exposition as well as the computational experience. This research has been partially supported by Comunidad de Madrid [Grants URJC-CM-2008-CET-3703, RIESGOS-CM] and the Ministry of Science and Innovation, Spain [Grant MTM2009-14087-C04-01].

References

Adelman D (2007) Dynamic bid prices in revenue management. *Oper. Res.* 55(4):647–661.

Akan M, Ata B (2009) Bid–price controls for network revenue management: Martingale characterization of optimal bid prices. *Math. Oper. Res.* 34(4):912–936.

Ball MO, Queyranne M (2009) Toward robust revenue management: Competitive analysis of online booking. *Oper. Res.* 57(4):950–963.

Bertsimas D, Popescu I (2003) Revenue management in a dynamic network environment. *Transportation Sci.* 37(3):257–277.

Birge JR (1985) Decomposition and partitioning methods for multistage stochastic linear programs. *Oper. Res.* 33(5):989–1007.

Birge J, Louveaux FV (1997) *Introduction to Stochastic Programming* (Springer, New York).

Cristóbal MP, Escudero LF, Monge JF (2009) On stochastic dynamic programming for solving large-scale planning problems under uncertainty. *Comput. Oper. Res.* 36(8):2418–2428.

Chen L, Homem-de-Mello T (2010) Re-solving stochastic programming models for air revenue management. *Ann. Oper. Res.* 177(1):91–114.

DeMiguel V, Mishra N (2008) What multistage stochastic programming can do for network revenue management. Working paper, London Business School, UK.

Donohue CJ, Birge JR (2006) The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse. *Algorithmic Oper. Res.* 1(1):20–30.

D'Sylva E (1982) O and D seat assignment to maximize expected revenue. Technical report, Boeing Commercial Airplane Company, Seattle.

Emich K, Möller W, Römisch W (2010) Airline network revenue management under uncertainty by Lagrangian relaxation. *12th Internat. Conf. Stochastic Programming, Halifax, Canada*.

Escudero LF, Garín A, Merino M, Pérez G (2007) A two-stage stochastic integer programming approach as a mixture of branch-and-fix coordination and Benders decomposition schemes. *Ann. Oper. Res.* 152(1):395–420.

Escudero LF, Garín MA, Merino M, Pérez G (2009) A general algorithm for solving two-stage stochastic mixed 0–1 first-stage problems. *Comput. Oper. Res.* 36(9):2590–2600.

Escudero LF, Garín MA, Merino M, Pérez G (2010a) An exact algorithm for solving large-scale two-stage stochastic mixed–integer problems: Some theoretical and computational aspects. *Eur. J. Oper. Res.* 204(1):105–116.

Escudero LF, Garín MA, Merino M, Pérez G (2010b) On BFC-MSMIP strategies for scenario cluster partitioning and twin node families branching selection and bounding for multistage stochastic mixed integer programming. *Comput. Oper. Res.* 37(4):738–753.

Glover F, Glover R, Lorenzo J, McMillan C (1982) The passenger mix problem in the scheduling airlines. *Interfaces* 12(3):73–79.

Haensel A, Mederer M, Schmidt H (2011) Revenue management in the car rental industry: A stochastic programming approach. *J. Revenue Pricing Management* 11(1):99–108.

Higle JL, Sen S (2005) A stochastic programming model for network resource utilization in the presence of multiclass demand uncertainty. Wallace SW, Ziemba WT, eds. *Applications of Stochastic Programming. MPS-SIAM Series on Optimization* (SIAM/MPS, Philadelphia), 299–313.

IBM ILOG (2007) CPLEX 11.0, http://www.ilog.com/products/cplex.

Kunnumkal S, Topaloglu H (2010) Computing time-dependent bid prices in network revenue management problem. *Transportation Sci.* 44(1):38–62.

Möller A, Römisch W, Weber K (2004) A new approach to O–D revenue management based on scenario trees. *J. Revenue Pricing Management* 3(3):265–276.

Möller A, Römisch W, Weber K (2008) Airline network revenue management by multistage stochastic programming. *Computational Management Sci.* 5(4):355–377.

Powell WB (2007) *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (Wiley, Hoboken, NJ).

Ross SM (1995) *Introduction to Stochastic Dynamic Programming* (Academic Press, Orlando, FL).

Smith BC, Penn CW (1988) Analysis of alternative origin–destination control strategies. *Proc. Twenty Eight Annual AGIFORS Sympos., New Seabury, MA.*

Talluri KT, van Ryzin GJ (1998) An analysis of bid-price controls for network revenue management. *Management Sci.* 44(11):1577–1593.

Talluri KT, van Ryzin GJ (1999) A randomized linear programming method for computing network bid prices. *Transportation Sci.* 33(2):207–216.

Talluri KT, van Ryzin GJ (2004) *The Theory and Practice of Revenue Management* (Springer).

Topaloglu H (2009a) On the asymptotic optimality of the randomized linear programming method for network revenue management. *Euro. J. Oper. Res.* 197(3):884–896.

Topaloglu H (2009b) Using Lagrangian relaxation to compute capacity-dependent bid prices in network revenue management. *Oper. Res.* 57(3):637–649.

van Slyke RM, Wets R (1969) *L*-shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.* 17(4):638–663.

Wets RJ-B (1974) Stochastic programs with fixed recourse: The equivalent deterministic program. *SIAM Rev.* 16(3):309–339.

Wollmer RD (1992) An airline seat management model for a single leg route when lower fare classes book first. *Oper. Res.* 40(1):26–37.

Wong JT, Koppelman FS, Daskin MS (1993) Flexible assignment approach to itinerary seat allocation. *Transportation Res. Part B: Methodological* 27(1):33–48.