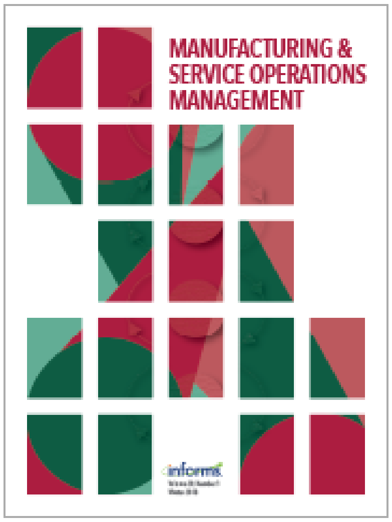


Manufacturing & Service Operations Management

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Assign-to-Seat: Dynamic Capacity Control for Selling High-Speed Train Tickets

Feng Zhu, Shaoxuan Liu, Rowan Wang, Zizhuo Wang

MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Articles in Advance, pp. 1–18
https://pubsonline.informs.org/journal/msom
ISSN 1523-4614 (print), ISSN 1526-5498 (online)

Assign-to-Seat: Dynamic Capacity Control for Selling High-Speed Train Tickets

Feng Zhu,[a] Shaoxuan Liu,[b,*] Rowan Wang,[c,*] Zizhuo Wang[d,*]

[a] Institute for Data, Systems, and Society, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139; [b] SJTU-BOC Institute of Technology and Finance and Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai 200030, China; [c] SUSTech Business School, Southern University of Science and Technology, Shenzhen 518055, China; [d] School of Data Science, The Chinese University of Hong Kong, Shenzhen, Shenzhen 518172, China

*Corresponding author

Contact: fengzhu@mit.edu, https://orcid.org/0000-0003-4979-4879 (FZ); liusx@sjtu.edu.cn (SL); wangy2021@sustech.edu.cn, https://orcid.org/0000-0002-7665-4460 (RW); wangzizhuo@cuhk.edu.cn, https://orcid.org/0000-0003-0828-7280 (ZW)

Received: January 23, 2022
Revised: October 8, 2022
Accepted: December 25, 2022
Published Online in Articles in Advance: February 14, 2023

https://doi.org/10.1287/msom.2023.1188

Abstract. ***Problem definition*:** We consider a revenue management problem that arises from the selling of high-speed train tickets in China. Compared with traditional network revenue management problems, the new feature of our problem is the *assign-to-seat* restriction. That is, each request, if accepted, must be assigned instantly to a single seat throughout the whole journey, and later adjustment is not allowed. When making decisions, the seller needs to track not only the total seat capacity available, but also the status of each seat. ***Methodology/results*:** We build a modified network revenue management model for this problem. First, we study a static problem in which all requests are given. Although the problem is NP-hard in general, we identify conditions for solvability in polynomial time and propose efficient approximation algorithms for general cases. We then introduce a bid-price control policy based on a novel maximal sequence principle. This policy accommodates nonlinearity in bid prices and, as a result, yields a more accurate approximation of the value function than a traditional bid-price control policy does. Finally, we combine a dynamic view of the maximal sequence with the static solution of a primal problem to propose a "re-solving a dynamic primal" policy that can achieve uniformly bounded revenue loss under mild assumptions. Numerical experiments using both synthetic and real data document the advantage of our proposed policies on resource-allocation efficiency. ***Managerial implications*:** The results of this study reveal connections between our problem and traditional network revenue management problems. Particularly, we demonstrate that by adaptively using our proposed methods, the impact of the assign-to-seat restriction becomes limited both in theory and practice.

Funding: S. Liu's research is partly supported by the National Natural Science Foundation of China (NSFC) [Grant NSFC-72072117]. Z. Wang's research is partly supported by the NSFC [Grant NSFC-72150002].

Supplemental Material: The online appendix is available at https://doi.org/10.1287/msom.2023.1188.

Keywords: revenue management • capacity control • dynamic programming • bid-price control

1. Introduction

Revenue management (RM) has played an important role in many industries over the past few decades. Starting with the airline industry in the 1980s, revenue management transformed various industries (e.g., airline, hotel, and car rental) by revolutionizing the ways in which sales decisions are made (Cross 1997). This trend of transformation has accelerated in the last several years. Given the rapid growth of information technology and the arrival of big data, every firm is hoping to take advantage of data to make better decisions, increase sales, and generate higher revenue.

In this work, we study a revenue management problem that arises in the high-speed railway industry in China. This industry is growing rapidly, and high-speed trains have become a vital transportation mode that brings great convenience to the Chinese people. For example, the Beijing–Shanghai high-speed train has carried more than 825 million passengers as of 2018 since the operations began in 2011. It also posted revenue of 31.2 billion RMB during the year 2018 alone (*Global Times* 2018). Across China, high-speed trains have transported more than 2 billion passengers as of 2018, and the number of passengers increases at an annual rate of more than 15% (*Global Times* 2019).

Despite such high demand and fast growth, China's high-speed railway industry is still struggling to make a profit and pay off its debt, mainly because of the huge costs for construction and operation (Murayama 2017). In light of the large number of passengers, one might suppose that profits could be increased by using advanced dynamic pricing strategies similar to those in the airline industry. At the moment, however, train ticket prices remain fully regulated by the Chinese government; this means that the fares between two cities are fixed, regardless of the time of purchase. Nonetheless, the train company can decide whether to close out sales for an itinerary. Indeed, in many cases, it is difficult to buy train tickets for a short itinerary, even though there are seats available; the reason is that the train company may reserve those seats for longer itineraries. So whether to open up an itinerary for sale is controlled by the train company, which results in a capacity-control problem. In addition, the fares may not be additive in segments; that is, longer (resp., shorter) itineraries tend to have a lower (resp., higher) per-mile price. For example, the Beijing–Shenzhen train G79 makes seven stops en route, including Wuhan as one stop. The fares for the Beijing–Wuhan and Wuhan–Shenzhen segments are (respectively) 520.5 renminbi (RMB) and 538 RMB, whereas the whole-trip Beijing–Shenzhen fare is only 936.5 RMB (all fares are as of July 1, 2019), or about 12% less than the sum of the segment fares. It follows that selling a seat to two passengers for "sub-journeys" could generate more revenue than selling that seat to a single passenger for the entire journey. At the same time, reserving seats for shorter journeys runs the risk of being unable to find a matching demand and so leaving empty seats on the train.

Furthermore, the train company faces a challenge known as the *assign-to-seat* restriction. Regulations require that, when accepting a passenger's request for an itinerary, the train company must assign a single numbered seat to that passenger; this means that there must be an available seat throughout the entire journey in order for a request to be accepted. In the airline industry, a trip with multiple segments typically consists of different flights. Because passengers need to get off the previous flight before boarding the next one, it makes no difference whether a passenger sits in the same seat throughout. Yet, this is not the case when one is traveling by train. In the Chinese high-speed train system (and perhaps many other train systems), each passenger is assigned to a single seat for his entire trip, and the corresponding seat number is given to the passenger at the time of ticket booking. In other words, regulations forbid serving a passenger via a combination of multiple seats. Note also that the booking decision needs to be made instantly and that a seat assignment cannot be altered afterward. Hence, the assign-to-seat restriction makes the associated capacity-allocation problem challenging and distinct from similar problems studied in the literature.

1.1. Research Questions and Contributions

We intend to shed light on the problem just described and to propose some practical capacity-allocation policies. In particular, we investigate the following questions.

1. How can we model such a capacity-allocation problem, given its unique restrictions? How can we efficiently track the system dynamics, especially for trains with a large number of seats?

2. Are there any simple and easy-to-implement capacity-allocation policies? Will a given policy perform well under different scenarios?

3. What are the connections and differences between this problem and the classical network revenue management problem?

To answer these questions, we construct a modified network revenue management model. We consider a discrete-time model, in which passengers arrive sequentially, each requesting (with some probability) a certain itinerary. The price for each itinerary is fixed. The train company must decide, in each time period, whether to accept a request for a certain itinerary; if accepted, it must then identify a seat to which the request is assigned, given the availability status of the remaining seats on the train. Observe that the latter question is not faced in traditional network revenue management problems, where only the total capacity matters (as in the airline setting) or where the assignment decision can be made at a later time (as in the hotel setting).

We begin our analysis by considering a static problem in which all passenger requests are given in advance. The static problem is useful both for understanding the problem's structure and for informing the allocation rule in the dynamic setting. We show that the static problem is NP-hard for a general capacity matrix (i.e., we allow the capacity matrix to reflect any availability status on the train, with certain seats taken on certain segments), *regardless* of the price structure (e.g., even when prices have a linear structure). Yet, if the initial capacity matrix satisfies a certain structure (which we term the "nonoverlapping or sharing endpoints" (NSE) structure), then there exists a polynomial-time algorithm that solves the static allocation problem. Moreover, we propose efficient approximation algorithms for the general problem. Our analysis connects the problem to traditional network RM problems and highlights how the assign-to-seat restriction affects seat allocation. The principal managerial insight is that, regardless of the price structure, the assign-to-seat restriction has no effect if (a) the assignment of requests can be delayed to the end of the time horizon and (b) the capacity matrix reflects a strongly NSE structure.

We then analyze the dynamic model, starting with bid-price control policies. In our setting, it is necessary to consider how bid-prices can be generated to determine which seat is allocated to a request. This consideration is closely related to the problem of how to track the dynamics of a capacity matrix. We propose two ways of generating bid-prices: One is based on the dual formulation of the static model; the other is based on a formulation that looks at the longest consecutive available segments, or what we call the *maximal sequence*, of the seats. Both types of bid-prices can be generated by using the approximate dynamic programming (ADP) approach, but the latter allows for *nonlinear* bid-prices and leads to more accurate approximation of the value functions. Encouraged by our results on bid-price control policies, we propose a *re-solving a dynamic primal* (RDP) policy that re-solves a primal problem under the maximal sequence formulation in each time period and then uses that solution to guide the allocation. We quantify the RDP policy's asymptotic performance for general cases. We show that, under an additional property (which we call the "no early ending" (NEE) property) on the set of arrival rates, the RDP approach achieves a uniformly bounded revenue loss. These results, when combined with those obtained for the static problem, lead to a critical managerial insight: The revenue loss resulting from the assign-to-seat restriction is asymptotically bounded if the set of arrival rates is NEE and the initial capacity matrix is strongly NSE. To validate the performance of our proposed policies, we conduct numerical experiments using real data from the China Railway High-speed. These experiments help to conclude that both policies based on maximal sequence achieve excellent performance.

In short, our paper is the first to study the dynamic capacity-allocation problem in a railway setting with the assign-to-seat requirement, a setting that differs in several aspects from the settings of previous revenue management problems. Our analysis illuminates the structure of such problems, and we propose practical allocation policies. The results reported here can serve as a first step toward understanding such problems and, ultimately, increasing the utilization of train seats.

Before proceeding, we introduce notations to be used throughout. We use $\mathbb{N} = \{0, 1, 2, \dots\}$ to denote the set of all natural numbers and $[N] = \{1, \dots, N\}$ to denote the set of all positive integers that are no larger than N. We use $\mathbf{e}_i$ to denote a row vector of 0's, but with 1 at the ith entry, and $\mathbf{e}_{ij}$ to denote a row vector of 0's, but with 1 from the ith entry to the jth entry. The dimension of $\mathbf{e}_i$ or $\mathbf{e}_{ij}$ will be clear from the context. For any set S, we use $|S|$ to denote its cardinality. Finally, for two vectors or matrices $\mathbf{a}$ and $\mathbf{b}$, we use $\mathbf{a} \leqslant \mathbf{b}$ to denote component-wise inequalities.

Our paper proceeds as follows. In the rest of this section, we review related literature. In Section 2, we formulate our problem. In Section 3, we study the static model. In Section 4, we study the dynamic model and propose several policies. We perform numerical experiments in Section 5. Section 6 concludes the paper.

1.2. Literature Review

Our study is related to the literature on network revenue management and interval scheduling. In what follows, we review the two lines of research and discuss their relation to our work.

1.2.1. Network Revenue Management

Broadly speaking, our work is closely related to the quantity-based *network revenue management* problem, which has been widely studied in the literature since Williamson (1992). The network RM problem can be fully characterized by a dynamic programming (DP) formulation. However, a crucial challenge is that the number of states grows exponentially with the size of the problem, making it impractical to solve directly. There have been various attempts to circumvent this difficulty—for example, by deriving bid-prices or booking-limit controls from static formulations or by approximating the value function with some simple structures.

A seminal work in the literature on booking-limit control is that of Gallego and van Ryzin (1997), who study a static model and propose make-to-stock and make-to-order policies. Theoretically, booking-limit control exhibits the property of asymptotic optimality. However, it lacks flexibility when dealing with stochastic demand and may not perform well for practical-sized problems. Talluri and van Ryzin (1998) are among the first to propose bid-price control policies. Since then, an abundant literature has focused on deriving delicate bid prices and tighter bounds on the value functions. Bertsimas and Popescu (2003) consider a certainty equivalence control policy that directly uses the static model's optimal value to approximate the initial value function. Adelman (2007) offers a framework for obtaining an upper bound on the value functions via an approximate DP approach. Under this approach, the DP is viewed as a linear program (LP), approximation structures are substituted for value functions, and then the LP is solved with far fewer variables. Much of the subsequent research followed this line (see, e.g., Zhang and Adelman 2009, Zhang 2011, Tong and Topaloglu 2013, and Kunnumkal and Talluri 2015). There is also a stream of literature that uses Lagrangian relaxation to approximate the value functions (see, e.g., Topaloglu 2009, Kunnumkal and Topaloglu 2010, Tong and Topaloglu 2013, and Kunnumkal and Talluri 2015).

A useful technique that is often applied in practice is *re-solving*. The idea is that the performance of a static policy could be improved by periodically incorporating updated state information. The impact of re-solving has been extensively studied in the literature. Cooper (2002) gives a counterexample that shows how re-solving could actually reduce collected revenue. Jasin and Kumar (2013) point out that re-solving does not help for a wide range of deterministic policies. Yet, in a variety of settings, re-solving has been shown to perform well in terms of both theory and practice (see, e.g., Maglaras and Meissner 2006, Secomandi 2008, and Chen and Homem-de Mello 2010). In recent years, researchers begin to explore new forms of policies, such as those based on probabilistic allocation, for which elegant theoretical properties are established (see, e.g., Reiman and Wang 2008, Jasin and Kumar 2012, and Bumpensanti and Wang 2020).

We now explain the relation between network RM problems and the problem studied in our paper. First, our problem is related to the *railway* revenue management problems (see, e.g., Ciancimino et al. 1999 and Armstrong and Meissner 2010), which are a special case of general network RM problems. In railway RM problems, each "product" is usually a consecutive combination of legs, and our problem inherits this property. However, we incorporate the assign-to-seat restriction, which is not addressed in the existing literature. This difference requires keeping track of not only each leg's total remaining capacity, but also each seat's detailed occupation status. As we will show, such a new feature indeed yields different analysis and unique results for the corresponding revenue management problem. Second, our problem is also related to the *hotel* revenue management literature on multiple stays (see, e.g., Goldman et al. 2002, Liu et al. 2008, Nadarajah et al. 2015, and Aydin and Birbil 2018). These works, likewise, do not incorporate an assign-to-seat restriction, which is the key ingredient in our problem.

Last, but not least, there is a line of research that addresses "online packing" problems, which are closely related to network RM problems. In such problems, each arriving request consists of a combination of items, and there is a capacity constraint on the total amount of each item. Dynamic algorithms have been proposed for this problem under various settings (see, e.g., Devanur and Hayes 2009, Molinaro and Ravi 2013, Agrawal et al. 2014, Kesselheim et al. 2014, and Banerjee and Freund 2020). In contrast, our problem features additional structure because each request consists of *consecutive* seats; moreover, our problem incorporates an assign-to-seat restriction. These differences make our problem different from typical packing problems.

1.2.2. Interval Scheduling

From a technical perspective, our work is closely related to studies of interval scheduling, which is a special type of scheduling problem. Interval scheduling problems are formulated in terms of "machines" and "jobs." Each job is represented by an *interval* that indicates the time during which it must be carried out. The function of machines is to execute jobs, and there may exist multiple types of machines. The goal is to schedule the jobs on the machines in a way that minimizes total costs or maximizes total rewards. For comprehensive reviews of the literature on interval scheduling, readers are referred to Schmidt (2000), Kolen et al. (2007), and Kovalyov et al. (2007). In the following, we review only those works that are very relevant to our paper.

Arkin and Silverberg (1987) are among the earliest to consider scheduling jobs with fixed starting and ending times and different weights. Under the assumption of identical machines, they propose an algorithm that runs polynomially in the number of jobs. Brucker and Nordmann (1994) extend the case of identical machines to that of nonidentical machines, but consider jobs with identical weights. They call such a problem a "*k*-track assignment problem" and show that establishing the existence of a valid schedule for given jobs and machines is, in general, NP-hard. The authors also propose a dynamic programming method to solve the problem. Kolen and Kroon (1993) investigate the computational complexity of a broader range of *k*-track assignment problems when the weights are arbitrary and the types of machines are given. Many other works follow this line of research and aim to maximize the total weight of scheduled jobs in different settings, such as considering identical machines (see, e.g., Carlisle and Lloyd 1995 and Bouzina and Emmons 1996), allowing for the processing of multiple jobs on a single machine (see, e.g., Faigle et al. 1999 and Angelelli et al. 2014), and adding constraints on overall processing time (see, e.g., Eliiyi and Azizoglu 2006).

A special case of the interval scheduling problem is the *job interval scheduling problem* (JISP), in which jobs are packaged into several groups, such that the schedule admits at most one job in each group. For this NP-hard problem, Spieksma (1999) applies integer optimization to formalize the problem with equal weights. By considering the linear optimization relaxation, the author obtains a $1/2$-approximation algorithm. Chuzhoy et al. (2006) propose a randomized algorithm that improves the ratio to $1 - 1/e - \varepsilon$ for any arbitrary positive number ε. Many other works have sought to obtain approximation algorithms with theoretical guarantees (see, e.g., Bar-Noy et al. 2001a, b and Bhatia et al. 2007).

Now, we explain the connections and differences between our study and those in this stream of literature. In a static setting when all requests are known, our problem is a special case of the weighted JISP. It follows that some theoretical properties or algorithms of the JISP could be applied to the static setting of our problem. However, different from the works in this field, we study how the aggregation of different jobs affects the problem solution. In our problem, there are often many identical jobs with identical weights, and, thus, we would like to identify the conditions under which an aggregation formulation could be useful. This aggregation approach is a unique aspect of our analysis, which means that our proposed algorithm and its complexity results will be different from those in the literature. In the dynamic setting, our problem is connected to *online* interval scheduling problems (see, e.g., Seiden 1998, Erlebach and Spieksma 2003, Im and

Wang 2011, and Shalom et al. 2014). When jobs arrive in an adversarial order and the number of identical machines are limited, but the types of job vary, it is typical to analyze the competitive ratios. We highlight that our study focuses on the case with a fixed number of job/machine (i.e., ticket) types, but a large number of machines (i.e., seats), and the jobs arrive in a stochastic and online fashion. Furthermore, we allow the initial seat status to be arbitrary. This feature, if translated into the online interval scheduling setting, means that the machines can be nonidentical or occupied during some intervals at the beginning. Because of these differences, we tackle the dynamic problem with approaches and analysis that are completely different from those in the online interval scheduling literature. In addition, we obtain asymptotically optimal policies for the dynamic problem.

2. Model

We consider the problem of a passenger railway service company selling train tickets for a particular route with N homogeneous seats. The route consists of $M+1$ stops or, equivalently, M *legs*. Here, a leg represents the trip between two adjacent stops; the first leg is denoted as leg 1, and the last leg is denoted as leg M. The seller can sell itineraries from leg i to leg j for any $j \geqslant i$, which we denote as $i \to j$, at a price p_{ij}. Our model assumes that the ticket prices p_{ij} for all i and j are fixed, which is common in many regulated industries. At the time a passenger purchases a train ticket, he must be assigned to a fixed seat for the entire trip. In other words, assigning a passenger to different seats on different legs during the itinerary is not allowed.

We adopt a discrete-time model. Time is "discretized" to $1, \ldots, T$, where 1 is the start of the selling horizon and T is the end. In each period t, a passenger with request $i \to j$ arrives with probability λ_{ij}^t. We assume each time period is sufficiently short; thus, $\sum_{1 \leqslant i \leqslant j \leqslant M} \lambda_{ij}^t \leqslant 1$ for all t. The probability that *no* customer arrives in period t is $\lambda_0^t = 1 - \sum_{1 \leqslant i \leqslant j \leqslant M} \lambda_{ij}^t$. For any $1 \leqslant t_1 \leqslant t_2 \leqslant T$, we use $d_{ij}^{[t_1,t_2]}$ to denote the number of requests $i \to j$ that are received in $[t_1, t_2]$. In addition, we set $\lambda_{ij}^{[t_1,t_2]} = \mathbb{E}[d_{ij}^{[t_1,t_2]}]$ as the expected number of requests $i \to j$ in $[t_1, t_2]$.

At the beginning of each period, based on the remaining seats available on different legs, the seller decides which itineraries to offer in the current period. At the same time, the seller must also determine the seat-number assignment for each itinerary offered. These decisions dictate whether an arriving passenger can find his requested itinerary available for purchase and, if so, the seat number assigned to him. We call such a decision rule a *policy* for the seller and denote it by π. More precisely, let $C^t \in \{0,1\}^{N \times M}$ be the matrix such that $C_{k\ell}^t$ denotes whether leg ℓ on seat k is available at the beginning of period t. We call C^t the *capacity matrix*, which characterizes the state of seats at the beginning of period t. For the sake of notational simplicity, we put $C_{k0}^t = C_{k(M+1)}^t = 0$ for all k and t. Then, a policy π can be interpreted as a mapping from time period t and the capacity matrix C^t to a set of binary variables $u_{k,ij}^t$, where $u_{k,ij}^t \in \{0,1\}$ represents whether a request $i \to j$ arriving in period t will be accepted and assigned to seat k. In our model, seat assignment is determined at the time of purchase and cannot be changed at a later time. Also, we assume that any rejected request is lost. In addition, we do not allow overbooking. The seller's objective is to identify a policy that leads to the highest expected revenue from ticket sales throughout the entire selling horizon.

3. The Static Problem

We now examine the structure and properties of the static problem, in which all passenger requests are known in advance. Analyzing the static problem is a common approach in the literature on network revenue management. Such analysis has two purposes. First, the static problem can serve as a useful offline benchmark, against which to measure the performance of any dynamic policy. Second, because the static problem is equivalent to collecting all requests and carrying out the final decision at the end of the booking horizon, analyzing this problem could help identify policies that are efficient in the dynamic setting.

In the static problem, the number of requests $i \to j$ is known to the seller, and we denote it by $d_{ij} \in \mathbb{N}$. The seller must determine the quantity of each type of request to accept in order to maximize the revenue. Let $x_{k,ij}$ indicate whether seat k is used to serve one request $i \to j$. Then, the static allocation problem can be formulated as the following integer program (IP):

$$
\begin{aligned}
\text{maximize} \quad & \sum_{1 \leqslant i \leqslant j \leqslant M} p_{ij} \sum_{k=1}^{N} x_{k,ij} \\
\text{subject to} \quad & \sum_{k=1}^{N} x_{k,ij} \leqslant d_{ij}, && \forall 1 \leqslant i \leqslant j \leqslant M, \\
& \sum_{(i,j): i \leqslant \ell \leqslant j} x_{k,ij} \leqslant C_{k\ell}, && \forall k \in [N], \ell \in [M], \\
& x_{k,ij} \in \{0,1\}, && \forall k \in [N], 1 \leqslant i \leqslant j \leqslant M.
\end{aligned}
\tag{1}
$$

Here, we adopt a formulation with a general capacity matrix C, where $C_{k\ell}$ represents whether seat k is available on leg ℓ. This formulation is useful because we hope to use the static model to instruct allocation during the selling horizon (when certain seats have already been taken). If all seats are available, as they are at the start of the sales horizon, then one can set all $C_{k\ell}$'s equal to one. Later, we show that, in fact, the profile of C affects the problem's complexity.

Because (1) is an integer program, it is natural to consider its LP relaxation given as follows:

$$\begin{aligned}
&\text{maximize} && \sum_{1\leqslant i\leqslant j\leqslant M} p_{ij}\sum_{k=1}^{N} x_{k,ij} \\
&\text{subject to} && \sum_{k=1}^{N} x_{k,ij} \leqslant d_{ij}, && \forall 1\leqslant i\leqslant j\leqslant M, \\
& && \sum_{(i,j):i\leqslant \ell\leqslant j} x_{k,ij} \leqslant C_{k\ell}, && \forall k\in[N], \ell\in[M], \\
& && x_{k,ij} \geqslant 0, && \forall k\in[N], 1\leqslant i\leqslant j\leqslant M.
\end{aligned} \tag{2}$$

Here, we ignore the $x_{k,ij}\leqslant 1$ constraints because the second group of constraints in (2) already ensures that $x_{k,ij}\leqslant 1$. The first question of interest is whether there is a gap between the IP Formulation (1) and its LP Relaxation (2). The following example shows that, in general, there could be a gap between (1) and (2), even when the prices p_{ij}'s satisfy some nice properties.

Example 1. Consider the capacity matrix, shown in Figure 1(a), with three seats and six legs. We assume that the passengers' requests are as follows: $d_{ij} = 1$ if $(i,j)\in\{(1,2),(1,5),(2,4),(3,6),(5,6)\}$ and $d_{ij} = 0$ otherwise. Also, we set $p_{ij} = \sum_{i\leqslant t\leqslant j} v_t$, where $v_1 = v_2 = v_3 = \frac{1}{2}v_4 = v_5 = \frac{1}{2}v_6 = 1$. In this setting, the optimal value of IP (1) is 16 (the optimal allocation is shown in Figure 1(b)), and the optimal value of LP (2) is 16.5 (the optimal allocation is shown in Figure 1(c)). In this example, prices are *linear*. That is, we could endow each leg with a unit price, and the price of each ticket is the sum of the unit prices of its occupied legs. □

Example 1 indicates that, even when prices are linear, there might be an integrality gap between (1) and (2). In fact, we show that solving the static problem is NP-hard in general.

Theorem 1. *Problem (1) is NP-hard.*

The proof of Theorem 1 and other technical results can be found in the online appendix. Despite the hardness of solving the general problem, the question remains of whether we can efficiently solve (1) under certain conditions. As we will show, the structure of the capacity matrix C plays an important role in determining the problem's computational complexity. If the capacity matrix C satisfies certain properties, then we can prove that the problem becomes polynomial-time solvable and that (1) and (2) share the same optimal objective value.

Before presenting our results, we introduce some definitions that characterize the structure of the capacity matrices. We use $C_k\in\{0,1\}^{1\times M}$ to denote the k^{th} row of the capacity matrix C.

Definition 1 (Maximal Sequence). Let $C\in\{0,1\}^{N\times M}$ be a given capacity matrix. Then, we call $[u,v]$ a *maximal sequence of seat* k in C and denote it as $[u,v]\sim C_k$, if and only if

$$C_{ku} = C_{k(u+1)} = \cdots = C_{kv} = 1 \quad\text{and}\quad C_{k(u-1)} = C_{k(v+1)} = 0.$$

Here, we define $C_{k0} = C_{k(M+1)} = 0$ for all k. Furthermore, we call $[u,v]$ a *maximal sequence in* C and denote it as $[u,v]\sim C$, if there exists a $k\in[N]$ such that $[u,v]$ is a maximal sequence of seat k. We also define $\mathcal{M}_{uv}(C)$ as the set of seats that contain $[u,v]$ as a maximal sequence; that is, $\mathcal{M}_{uv}(C) = \{k\in[N]\,|\,[u,v]\sim C_k\}$. □

Definition 2 (NSE and Strongly NSE). Given a capacity matrix $C\in\{0,1\}^{N\times M}$, we say that C has the *nonoverlapping or sharing endpoints* property, or simply that C is NSE, if for *any* two maximal sequences $[u_1,v_1]$ and $[u_2,v_2]$ in C, one of the following holds:

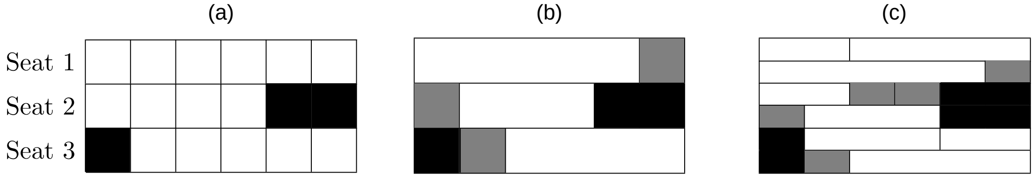
- $u_1 = u_2$ or $v_1 = v_2$ (sharing at least one endpoint);
- $u_1 > v_2$ or $v_1 < u_2$ (nonoverlapping).

Furthermore, we say that C is *strongly NSE* if one of the following holds:

- $u_1 = u_2$ or $v_1 = v_2$ (sharing at least one endpoint);
- $u_1 > v_2 + 1$ or $v_1 < u_2 - 1$ (strongly nonoverlapping). □

Next, we introduce a formulation that is based only on the aggregated capacity—that is, without the assign-to-seat restriction. Let $c_\ell = \sum_{k=1}^{N} C_{k\ell}$ be the total number of seats with leg ℓ available. We set $c_0 = c_{M+1} = 0$. Let x_{ij} denote the number of requests $i\to j$ that are accepted.

Figure 1. Illustrations of Example 1



Notes. (a) The initial capacity matrix. (b) Optimal allocation based on (1). (c) Optimal allocation based on (2). The black blocks represent seat–leg pairs that are already occupied and, hence, unavailable for allocation; the white and gray blocks represent pairs that are (respectively) allocated to demand requests and unused in the allocation. Observe that, in (c), the fraction of requests can be accepted and the seats are used to serve fractional requests.

Then, the aggregated allocation problem can be written as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{1 \leqslant i \leqslant j \leqslant M} p_{ij} x_{ij} \\ \text{subject to} \quad & 0 \leqslant x_{ij} \leqslant d_{ij}, \qquad \forall 1 \leqslant i \leqslant j \leqslant M, \\ & \sum_{(i,j): i \leqslant \ell \leqslant j} x_{ij} \leqslant c_\ell, \quad \forall \ell \in [M]. \end{aligned} \tag{3}$$

Problem (3) has certain nice properties. First, both the number of variables and the number of constraints are $O(M^2)$ and therefore do not depend on N. More importantly, it always has an integral optimal solution. To see this, we note that the constraints of (3) are totally unimodular (Ciancimino et al. 1999). Moreover, Problem (3) can be viewed as a network flow problem, which can be solved in strongly polynomial time (Arkin and Silverberg 1987). Because of the nice properties of (3), it is enticing to ask whether (3) could provide the guidance for solving (1) under certain circumstances. Note that, in general, the gap between (1) (or (2)) and (3) can be arbitrarily large. For example, consider $N = 2L$, $M = 2$, and $C_{k\ell} = \mathbb{1}\{k \leqslant L, \ell = 1\} + \mathbb{1}\{k > L, \ell = 2\}$. Let $d_{12} = L$, $d_{11} = d_{22} = 0$ and $p_{11} = p_{22} = p_{12} = 1$. Then, the objective value of (1) (or (2)) is zero because no request can be accepted, whereas that of (3) is L. Despite the potential gap in general, Theorem 2 shows that there is a strong tie between (1) and (3), and such a tie is related to the NSE property of the capacity matrix.

Theorem 2. *For any fixed positive prices $\{p_{ij}\}$, (1) and (3) have the same optimal value for any nonnegative integers $\{d_{ij}\}$ if and only if C is strongly NSE. Furthermore, if C is NSE, then (1) can be solved in polynomial time.*

For general capacity matrices that do not satisfy the NSE property, Theorem 2 indicates that one may not obtain an optimal solution through solving (3). Nevertheless, we provide efficient approximation algorithms for those general cases in the online appendix.

Theorem 2 leads to the following corollary, which will be useful in our subsequent discussion.

Corollary 1. *If C is strongly NSE, then for any nonnegative real numbers $\{d_{ij}\}$, (2) and (3) have the same optimal value.*

By Theorem 2, if the capacity matrix C is strongly NSE, then, from the *static* point of view, we can achieve the same revenue with or without the assign-to-seat restriction and regardless of the demand level. For other classes of capacity matrices, this property does not hold. From the *dynamic* point of view and within the special class of strongly NSE matrices, if we are allowed to assign seats for the requests after the final period T, then as long as the accepted requests satisfy the total capacity constraint, we can always find a feasible assignment. In practice, the initial capacity matrix C at the beginning of the selling horizon is completely unoccupied, which means C is strongly NSE. Therefore, the difference between the revenue achieved by any dynamic policy with and without the assign-to-seat restriction can also be viewed as the value of delayed assignment.

We also give some remarks on the difference between our problem and the problem in hotel revenue management. In the hotel industry, if there are multiple-day stays (bookings), then there is also an assign-to-seat restriction: customers expect to remain in the same room during their stays. However, in contrast to our problem, the hotel can wait until customers check in to assign room numbers. When the hotel is planning for a certain time horizon until a specified time period, the availability of each room is consecutive in time and ends on an identical day, forming a strongly NSE matrix. Our results indicate that, as long as customer requests satisfy the hotel's total capacity constraints for each day, the hotel can always find a feasible assignment. Therefore, even though there is an equivalent of the assign-to-seat restriction in the hotel industry, the possibility of delayed assignment simplifies the problem. The assignment in our problem must be made at the time of request, which renders the problem far more challenging.

4. The Dynamic Problem

Having analyzed the static problem, we now focus on the dynamic allocation problem. One critical insight from the static case is that the *maximal sequence* provides a unique and useful perspective to tackle our problem. In this section, we first propose a bid-price control policy with nonlinear bid prices built on the notion of maximal sequence. Then, based on re-solving a delicate LP inspired by the maximal sequence approach, we propose a policy that achieves a tight asymptotic loss, as compared with the hindsight optimum (i.e., the full-information benchmark; see definition later in the text). Bid-price control and re-solving are two classical ideas in the literature on network revenue management and have been well studied (see, e.g., Gallego and van Ryzin 1997, Talluri and van Ryzin 1998, and Reiman and Wang 2008). In this section, we investigate how one can overcome the challenges entailed by a dynamic context and utilize the special structures imposed by the assign-to-seat restriction. In addition, we propose policies that are implementable in practice. We note that another classical type of policy studied in the network RM literature is the booking-limit control policy. In the online appendix, we discuss in detail how we design such a policy for our problem to circumvent the challenge from the assign-to-seat restriction. We also present a result on the asymptotic property of the booking-limit control policy for our problem.

We start by describing the basic elements in the framework of our analysis. An instance $\mathcal{I}$ of the dynamic problem is defined by three parts: the (initial) capacity matrix $C \in \{0,1\}^{N\times M}$, the price set $\{p_{ij}\}_{i\leqslant j}$, and the set of arrival probabilities $\{\lambda_{ij}^t\}_{i\leqslant j, t\in[T]}$. We write the instance as $\mathcal{I} = \langle C, \{p_{ij}\}_{i\leqslant j}, \{\lambda_{ij}^t\}_{i\leqslant j, t\in[T]}\rangle$. To evaluate the performance of any policy, we use an *asymptotic regime*, which is a commonly used approach in the revenue management literature (see, e.g., Gallego and van Ryzin 1997 and Talluri and van Ryzin 1998). Given any $\mathcal{I}$, we define a sequence of problems indexed by $\theta \in \mathbb{N}$ with $C(\theta) \in \{0,1\}^{\theta N\times M}$ and $\{\lambda_{ij}^t(\theta)\}_{i\leqslant j, t\in[\theta T]}$ satisfying

$$C(\theta)_{(k+(s-1)N)\ell} = C_{k\ell}, \quad \forall s \in [\theta], k\in[N], \ell\in[M]$$
$$\text{and} \quad \lambda_{ij}^t(\theta) = \lambda_{ij}^{\lceil t/\theta\rceil}, \quad \forall t\in[\theta T],\ i\leqslant j.$$

That is, we scale the number of seats (with the same profile) and the length of the time horizon proportionally.

For any policy π, let $V_\theta^\pi(\mathcal{I})$ denote the expected revenue collected under π in the θ^{th} problem for instance $\mathcal{I}$. Among all policies, a special one is the *dynamic programming* policy, as it achieves the maximal expected revenue. A formal DP formulation is given as follows. Let $f^t(C)$ denote the maximal expected value to go at the beginning of period t with capacity matrix C. Then, the recursive formula for $f^t(C)$ can be written as:

$$f^t(C) = \max_{u^t\in\mathcal{D}^t(C)} \left\{ \sum_{i\leqslant j} \lambda_{ij}^t \left(\sum_{k=1}^N p_{ij} u_{k,ij}^t + f^{t+1}\left(C - \sum_{k=1}^N u_{k,ij}^t \mathbf{e}_k^\top \mathbf{e}_{ij} \right) \right) + \lambda_0^t f^{t+1}(C) \right\}, \quad \forall t\in[T], C\geqslant 0,$$
$$f^{T+1}(C) = 0, \quad \forall C \geqslant 0, \tag{4}$$

where $\mathcal{D}^t(C) = \{u_{k,ij}^t \in \{0,1\},\ \forall i,j,k \mid \sum_{k=1}^N u_{k,ij}^t \leqslant 1,\ \forall i,j,$ and $u_{k,ij}\mathbf{e}_k^\top \mathbf{e}_{ij} \leqslant C,\ \forall i,j,k\}$. We can also rewrite (4) in a more compact form as follows:

$$f^t(C) = \mathbb{E}_{(i,j)\sim\lambda^t}\left[\max_{\substack{k\in[N]:\\ C_k \geqslant \mathbf{e}_{ij}}} \{ f^{t+1}(C - \mathbf{e}_k^\top \mathbf{e}_{ij}) + p_{ij}, f^{t+1}(C)\} \right], \quad \forall t\in[T], C\geqslant 0,$$
$$f^{T+1}(C) = 0,\ \forall C\geqslant 0, \tag{5}$$

where $(i,j)\sim\lambda^t$ stands for the probability distribution of the incoming request in time period t. For notational brevity, we assume that there is a probability λ_0^t such that $(i,j) = (0,0)$, and for that case $\{k\in[N] \mid C_k \geqslant \mathbf{e}_{ij}\} = \emptyset$.

Apart from DP, another special "policy" is the *hindsight optimum* (HO), where the decision maker has full information on the demand realization for the entire time horizon and optimizes over the allocation schemes. We put "policy" in quotation marks here because a decision maker can never know the full demand realization ahead. Hence, it is impossible for the HO to be attained by an actual policy. For any sample path, the hindsight optimum for that sample path is computed by solving a (relaxed) static Problem (2), and $V_\theta^{\text{HO}}(\mathcal{I})$ is the expected value over all sample paths. It should be clear that, for any policy π, we have

$$V_\theta^\pi(\mathcal{I}) \leqslant V_\theta^{\text{DP}}(\mathcal{I}) \leqslant V_\theta^{\text{HO}}(\mathcal{I}).$$

It is well known that DP, despite its optimality, is computationally complex, owing to the "curse of dimensionality." Thus, we use HO as our benchmark to evaluate the performance of any policy π. More precisely, we are interested in constructing a computationally efficient policy π such that the gap between HO and π, denoted by $V_\theta^{\text{HO}}(\mathcal{I}) - V_\theta^\pi(\mathcal{I})$, has a near-optimal rate that depends on θ. This performance metric is often used in the network RM literature (see, e.g., Jasin and Kumar 2013, Banerjee and Freund 2020, and Bumpensanti and Wang 2020).

4.1. Bid-Price Control (BPC) Policies

A class of policy that is popular in the network RM literature is the *bid-price control policy*. For this policy, in each time period, each piece of resource is associated with a *bid-price* that captures the resource's fair value at that time, and allocations are made based on those bid-prices. Bid-price control policies are flexible (one can use different methods to calculate the bid-prices) and easy to implement in practice (after bid-prices are calculated, the allocation is usually simple). In this section, we investigate bid-price control policies for our problem. We propose two different ways of dynamically computing bid-prices: One is based on the dual formulation of the static model; the other is based on a more delicate formulation of the problem by considering the longest consecutive available segments, or *maximal sequence*, of the seats. We describe (a) how bid-prices based on maximal sequence are related to (approximate) dynamic programming approaches and (b) why the policy based on the maximal sequence approach is better than the canonical bid-price policy for making dynamic capacity-allocation decisions.

4.1.1. Traditional BPC Policy. We first give a brief description of how canonical bid-prices are obtained using standard methods based on the static Model (2). Consider the dual problem of (2), where d_{ij} is replaced by the expected total demand $\lambda_{ij} = \sum_t \lambda_{ij}^t$. Let z_{ij} and $\beta_{k\ell}$ be the dual variables for (respectively) the first and second constraint. Then, the dual problem of (2) can be written as follows:

$$\begin{aligned}
\text{minimize}_{z,\beta} \quad & \sum_{1\leqslant i\leqslant j\leqslant M} \lambda_{ij} z_{ij} + \sum_{k=1}^N \sum_{\ell=1}^M C_{k\ell}\beta_{k\ell} \\
\text{subject to} \quad & z_{ij} + \sum_{\ell=i}^j \beta_{k\ell} \geqslant p_{ij}, \qquad \forall 1\leqslant i\leqslant j\leqslant M, k\in[N], \\
& z_{ij} \geqslant 0, \qquad \forall 1\leqslant i\leqslant j\leqslant M, \\
& \beta_{k\ell} \geqslant 0, \qquad \forall k\in[N], \ell\in[M].
\end{aligned} \tag{6}$$

Here, $\beta_{k\ell}$ can be interpreted as the static bid-price for the k^{th} seat on the ℓ^{th} leg. When a request $i \to j$ arrives, we can calculate $p_{ij} - \sum_{\ell=i}^{j} \beta_{k\ell}$ for all k and choose $\arg\max_k \{p_{ij} - \sum_{\ell=i}^{j} \beta_{k\ell}\}$ as the seat to allocate for that request. In practice, one can re-solve (6) to obtain time-dependent bid-prices $\{\beta_{k\ell}^t\}$ and use $\beta_{k\ell}^t$ instead of $\beta_{k\ell}$. The bid-price control policy based on the static model, or BPC-S, is stated formally in Algorithm 1. We note that the canonical bid-prices can also be obtained from an ADP approach (see, e.g., Adelman 2007) using the DP Formulation (5). We explain it in the online appendix.

Algorithm 1 (Bid-Price Control with Static Model (BPC-S))

```
for t = 1, ..., T do
    Observe a request of type i^t → j^t;
    if {k | e_{i^t j^t} ≤ C_k^t} = ∅ then Reject the request;
    else
        Solve (6) with C = C^t and λ_ij = λ_ij^{[t,T]}, and
        obtain an optimal solution β^t = {β^t_{kℓ}}.
        Set k^t = arg max_{k∈[N]} {p_{i^t j^t} − Σ_{ℓ=i^t}^{j^t} β^t_{kℓ} | e_{i^t j^t} ≤ C_k^t}.
        Break ties arbitrarily;
        if p_{i^t j^t} − Σ_{ℓ=i^t}^{j^t} β^t_{k^t ℓ} ≥ 0 then
            Allocate the request to seat k^t and let
            C^{t+1} ← C^t − e_{k^t}^T e_{i^t j^t};
        else Reject the request and let C^{t+1} = C^t;
        end
    end
end
```

4.1.2. BPC Policy Based on Maximal Sequence. In the BPC-S policy, we treat each leg in each seat *separately*. However, a request always occupies a series of consecutive legs within a seat. In order to capture the value of a seat, it may be helpful to consider the legs *jointly*. In the following, we consider an alternative bid-price formulation based on the idea of maximal sequence, and we derive a different set of bid-prices and, hence, a different (and better-performing) control policy. The core idea, as we explain next, is to reformulate the dynamic system using maximal sequences instead of the capacity matrix C.

Let $\mathcal{A}$ denote the set of all $M \times M$ upper triangular matrices. For any capacity matrix C, we construct a projection $f: C \to \mathcal{A}$ in (7):

$$f(C)_{uv} = |\mathcal{M}_{uv}(C)| \quad \text{for all } 1 \leqslant u \leqslant v \leqslant M, \tag{7}$$

where $\mathcal{M}_{uv}(\cdot)$ is given in Definition 1. In other words, the (u,v)th entry of $f(C)$ equals the total number of maximal sequences $[u,v] \sim C$. Let $\mathcal{A}_{ij}$ be the *action set*:

$$\mathcal{A}_{ij} = \{R \in \mathbb{R}^{M\times M} | \exists (u,v) : u \leqslant i \leqslant j \leqslant v \text{ s.t.}$$
$$R_{u(i-1)} = -\mathbb{1}\{u < i\},\ R_{(j+1)v} = -\mathbb{1}\{j < v\},\ R_{uv} = 1\}.$$

We explain this definition of $\mathcal{A}_{ij}$ as follows. For any request $i \to j$, if we accept it, then we can assign it only to a seat that is unoccupied on legs i to j. If we assign the request to seat k with $[u,v] \sim C_k$ ($u \leqslant i \leqslant j \leqslant v$), then $[u,v]$ is split into $[u,i-1] \sim C_k$ and $[j+1,v] \sim C_k$. In other words, we consume one unit of $[u,v] \sim C_k$ while creating one unit of $[u,i-1] \sim C_k$ and $[j+1,v] \sim C_k$. This is why $R_{uv} = 1$, whereas $R_{u(i-1)} = -\mathbb{1}\{u < i\}$ and $R_{(j+1)v} = -\mathbb{1}\{j < v\}$. We then track the dynamics of $\mathcal{M}(C)$ in (8):

$$\begin{aligned}
&\mathcal{M}_{uv}(C) \leftarrow \mathcal{M}_{uv}(C) \setminus \{k\}, && f(C)_{uv} \leftarrow f(C)_{uv} - 1,\\
&\mathcal{M}_{u(i-1)}(C) \leftarrow \mathcal{M}_{u(i-1)}(C) \cup \{k\}, && \\
& && f(C)_{u(i-1)} \leftarrow f(C)_{u(i-1)} + 1 \quad \text{if } u < i,\\
&\mathcal{M}_{(j+1)v}(C) \leftarrow \mathcal{M}_{(j+1)v}(C) \cup \{k\}, && \\
& && f(C)_{(j+1)v} \leftarrow f(C)_{(j+1)v} + 1 \quad \text{if } v > j.
\end{aligned} \tag{8}$$

Now, we present the DP formulation from the maximal sequence perspective. For any $A \in \mathcal{A}$, consider the following DP:

$$\begin{aligned}
&v^t(A) = \mathbb{E}_{(i,j)\sim\lambda^t}\left[\max_{\substack{R\in\mathcal{A}_{ij}\\ R\leqslant A}} \{v^{t+1}(A-R) + p_{ij}, v^{t+1}(A)\}\right], && \forall t \in [T], A \geqslant 0,\\
&v^{T+1}(A) = 0, && \forall A \geqslant 0.
\end{aligned} \tag{9}$$

Here, $(i,j) \sim \lambda^t$ represents the probability distribution of an incoming request in time period t. For notational brevity, we assume the existence of a probability λ_0^t such that $(i,j) = (0,0)$, in which case $\{R : R \leqslant A | R \in \mathcal{A}_{ij}\} = \emptyset$. Using classical dynamic programming techniques, we can solve the following program to compute $v^1(A)$ for any given A:

$$\begin{aligned}
\text{minimize} \quad & \tilde{v}^1(A)\\
\text{subject to} \quad & \tilde{v}^t(\bar{A}) \geqslant \mathbb{E}_{(i,j)\sim\lambda^t}\\
& \left[\max_{\substack{R\in\mathcal{A}_{ij}\\ R\leqslant\bar{A}}} \{\tilde{v}^{t+1}(\bar{A}-R) + p_{ij}, \tilde{v}^{t+1}(\bar{A})\}\right], && \forall t \in [T], \bar{A} \geqslant 0,\\
& \tilde{v}^{T+1}(\bar{A}) \geqslant 0, && \forall \bar{A} \geqslant 0.
\end{aligned} \tag{10}$$

However, directly solving (10) remains computationally prohibitive. Now, we adopt the ADP approach (see, e.g., Adelman 2007). We approximate $\tilde{v}^t(A)$ as

$$\tilde{v}^t(A) = \theta^{\dagger t} + \sum_{u\leqslant v} A_{uv}\beta_{uv}^{\dagger}. \tag{11}$$

The term $\beta_{uv}^{\dagger}$ can be viewed as the approximated value for each maximal sequence $[u,v] \sim C$, or, in our context, the value of having an additional empty (unoccupied) seat from leg u to leg v. We define $\beta_{uv}^{\dagger}$ as the bid-price for the maximal sequence $[u,v]$. Note that here the approximation structure is a quasi-static formulation, where θ is time-dependent, but β is not.

Plugging (11) into (10) and reorganizing terms yields

$$\begin{aligned}
\text{minimize}_{\beta^{\dagger}, z^{\dagger}} \quad & \sum_{i \leqslant j} \lambda_{ij} z_{ij}^{\dagger} + \sum_{u \leqslant v} A_{uv} \beta_{uv}^{\dagger} \\
\text{subject to} \quad & z_{ij}^{\dagger} + \beta_{uv}^{\dagger} \geqslant p_{ij} + \beta_{u(i-1)}^{\dagger} + \beta_{(j+1)v}^{\dagger}, \quad \forall u \leqslant i \leqslant j \leqslant v, \\
& z_{ij}^{\dagger} \geqslant 0, \beta_{ij}^{\dagger} \geqslant 0, \quad \forall i \leqslant j, \\
& \beta_{ij}^{\dagger} = 0, \quad \forall i > j.
\end{aligned} \tag{12}$$

In the online appendix, we provide a detailed derivation of (12). After obtaining the bid-prices $\{\beta_{uv}^{\dagger}\}$ from (12), we can use them to make seat-allocation decisions. Intuitively, the additional value gained from allocating $i \to j$ into $[u,v] \sim C$ equals $p_{ij} + \beta_{u(i-1)}^{\dagger} + \beta_{(j+1)v}^{\dagger} - \beta_{uv}^{\dagger}$. When making a decision, we choose the seat that yields the maximum gain. In practice, we can re-solve (12) to obtain time-dependent bid-prices $\{\beta_{uv}^{\dagger t}\}$. This policy is stated formally in Algorithm 2.

Algorithm 2 (Bid-Price Control with Maximal Sequence (BPC-M))

```
for t = 1, ..., T do
    Observe a request of type i^t → j^t;
    if {(u,v) | u ≤ i^t ≤ j^t ≤ v, M_uv(C^t) ≠ ∅} = ∅ then
        Reject the request;
    else
        Solve (12) with A = f(C^t) and λ_ij = λ_ij^[t,T], and
        obtain an optimal solution {β_uv^†t};
        Set (u^t, v^t) = arg max_{(u,v)~C^t} {p_{i^t j^t} + β^†t_{u(i^t−1)} + β^†t_{(j^t+1)v}
        − β^†t_uv}. Break ties arbitrarily;
        if p_{i^t j^t} + β^†t_{u^t(i^t−1)} + β^†t_{(j^t+1)v^t} − β^†t_{u^t v^t} ≥ 0 then
            Allocate the request to a seat k^t ∈ M_{u^t v^t}(C^t);
            Update {M_uv(C^t)} according to (8);
            else Reject the request;
        end
    end
end
```

4.1.3. Relation Between the BPC Policies. Now that we have introduced two types of bid-prices, we are well positioned to investigate the relation between them. Theorem 3 shows that there is an advantage in bid-prices based on the maximal sequence approach.

Theorem 3. *For any $t \in [T]$ and $C \in \{0,1\}^{N \times M}$ and for any group of bid-prices $\{\beta_{k\ell}^{t}\}$ in (6), there exists a group of bid-prices $\{\beta_{uv}^{\dagger t}\}$ in (12) such that*

$$\beta_{uv}^{\dagger t} = \min_{k \in [N]} \left\{ \sum_{\ell : u \leqslant \ell \leqslant v} \beta_{k\ell}^{t} \right\} \quad \textit{for all } u \leqslant v. \tag{13}$$

Although we can show (see the proof of Theorem 3) that (6) and (12) share the same objective value, Theorem 3 indicates that, using bid-prices calculated at any given state C, BPC-M always results in a lower approximated value function than BPC-S does, for *any* (other) capacity matrix C'. In the meantime, by the property of the ADP approach (see, e.g., Adelman 2007), both bid-price approaches give upper bounds on the value function at any state, and, thus, it follows that BPC-M approximates the value function more accurately than BPC-S does.

The reason why BPC-M may lead to a more accurate approximation is that it allows for *nonlinear* bid-prices, which yield potentially better approximation structure and performance compared with traditional *linear* bid-prices. Note that, for any fixed t, $\beta_{k\ell}^{t}$ depends only on k and ℓ. Therefore, the value of consecutive legs under BPC-S must follow a linear structure with respect to legs; BPC-M relaxes this restriction and allows for a more flexible structure for the values. Intuitively, a long maximal sequence should have a greater intrinsic value than the sum of its short components, whose legs are disjointed. Such a relation can be reflected under bid-prices derived from BPC-M, but not BPC-S.

4.2. Re-solving a Dynamic Primal: Tight Asymptotic Loss

In Section 4.1, we proposed a bid-price control policy based on the maximal sequence approach. The BPC-M policy better captures the dynamic change of the state space and features greater flexibility than the canonical bid-price control policy does. However, the bid-price policies are established via a "dual" formulation, which might lose some information contained in the primal problem. Literature on online decision making has proposed another solution approach based on re-solving a primal problem, which can lead to strong theoretical results (see, e.g., Jasin and Kumar 2012, Vera and Banerjee 2019, Banerjee and Freund 2020, and Bumpensanti and Wang 2020). In this section, we propose a policy called *re-solving a dynamic primal*. The policy combines the advantages of re-solving with the maximal sequence approach. Theoretically, the policy achieves tight asymptotic loss compared with the full-information benchmark.

4.2.1. Description of RDP. We start by introducing the following LP:

$$\begin{aligned}
\text{maximize}_{\gamma} \quad & \sum_{i \leqslant j} p_{ij} \sum_{\substack{(u,v): \\ u \leqslant i \leqslant j \leqslant v}} \gamma_{uijv}, \\
\text{subject to} \quad & \sum_{\substack{(u,v): \\ u \leqslant i \leqslant j \leqslant v}} \gamma_{uijv} + \gamma_{0ij0} = d_{ij}, \quad \forall i \leqslant j, \\
& \sum_{\substack{(i,j): \\ u \leqslant i \leqslant j \leqslant v}} \gamma_{uijv} \leqslant \sum_{\substack{(k,\ell): \\ v+1 \leqslant k \leqslant \ell}} \gamma_{u(v+1)k\ell} \\
& + \sum_{\substack{(k,\ell): \\ \ell \leqslant k \leqslant u-1}} \gamma_{\ell k(u-1)v} + A_{uv}, \quad \forall u \leqslant v, \\
& \gamma_{uijv} \geqslant 0, \quad \forall u \leqslant i \leqslant j \leqslant v, \\
& \gamma_{0ij0} \geqslant 0, \quad \forall i \leqslant j.
\end{aligned} \tag{14}$$

We call (14) the *dynamic primal*, which is based on the maximal sequence formulation.

There are two ways to explain (14). The first one is a direct interpretation. Namely, the variable γ_{uijv} can be viewed as the number of times that request $i \to j$ is allocated into a $[u,v] \sim C$, and we explicitly use γ_{0ij0} to denote the number of $i \to j$ rejected. The first group of constraints in (14) means that the number of allocated requests, plus the rejected ones, equals the total number of requests $i \to j$. The second group of constraints requires a careful examination. The left-hand side is the number of times we *consume* a $[u,v] \sim C$ by assigning $i \to j$ into $[u,v]$; the right-hand side is the number of times that $[u,v]$ is *generated* throughout the whole time horizon, plus the number of $[u,v]$ available at the beginning. Note that $[u,v]$ is generated when (a) a request $v+1 \to k$ is assigned to some $[u,\ell]$ or (b) a request $k \to u-1$ is assigned to some $[\ell,v]$, which correspond to the first two terms on the right-hand side, respectively. Thus, this group of constraints means that the number of any $[u,v] \sim C$ left *in the end* is nonnegative. The last line of constraints simply states that the number of requests for which we should take a specific action (allocating $i \to j$ into some $[u,v]$, or rejecting it) should be nonnegative.

An alternative view of (14) is from the Approximate DP (12). Some simple derivations allow us to verify that (14) is the dual program of (12). Such a relation also justifies that (14) is a primal allocation problem under the maximal sequence approach.

Now, we formally present the RDP policy in Algorithm 3.

Algorithm 3 (Re-solving a Dynamic Primal (RDP))

for $t = 1, \ldots, T$ **do**
- Observe a request of type $i^t \to j^t$;
- Solve (14) with $A = f(C^t)$ and $d = \lambda^{[t,T]}$ as well as under the constraints

$$\gamma_{ui^tj^tv} \leqslant A_{uv}, \quad \forall (u,v): u \leqslant i^t \leqslant j^t \leqslant v, \tag{15}$$

  and obtain an optimal solution $\{\gamma^{\mathrm{RDP},t}\}$;
- Set $(u^t, v^t) = \arg\max_{(u,v)}\{\gamma^{\mathrm{RDP},t}_{ui^tj^tv}\}$. Break ties arbitrarily;
- Allocate $i^t \to j^t$ to $[u^t, v^t]$ ($[u^t, v^t] = [0,0]$ means that the request is rejected);

end

For the RDP policy, in each time period, we solve (14) using the expected remaining demand. Unlike standard network RM problems, in our problem, the number of remaining resources for each maximal sequence is not monotonically nonincreasing, because shorter maximal sequences may be generated by breaking down longer ones during the allocation process. To overcome the challenge due to this difference and to ensure the assignment's feasibility, we add an additional group of constraints (15) during the assignment process. In essence, (15) requires that the realized request $i^t \to j^t$ can be allocated only into some $[u,v] \sim C^t$. Then, we assign the request to the maximal sequence that has the maximum number of assignments in the primal solution for the corresponding itinerary. The following lemma states that the objective value of (14) is equal to that of (2). Somewhat surprisingly, adding constraints (15) does not affect the objective value of (14).

Lemma 1. *Let $A = f(C)$, where f is defined by (7). Then, (14) has the same objective value as (2). Furthermore, for any given $i^t \leqslant j^t$, (14) always has an optimal solution that satisfies (15).*

The proof of Lemma 1 makes full use of our interpretation of γ. In particular, we build a direct transformation from any optimal solution of (2) to a feasible solution of (14) combined with (15) by constructing an allocation scheme *dynamically*.

Now, we introduce our main result. We first introduce a property of the input parameters.

Definition 3 (No Early Ending (NEE)). We say that the set of arrival rates $\{\lambda^t_{ij}\}_{i \leqslant j, t \in [T]}$ satisfies the *NEE* property if

$$\inf_{\substack{i \leqslant j:\\ \lambda^{[1,T]}_{ij} > 0}} \lambda^T_{ij} > 0.$$

The NEE property states that, for each possible type of request ($\lambda^{[1,T]}_{ij} > 0$), there must be a positive probability that such a request arrives in the last period ($\lambda^T_{ij} > 0$). Equivalently speaking, the sales of all possible tickets end in the same time period. In particular, when the demand process is stationary, the NEE property is automatically satisfied.

Next, we characterize the asymptotic loss incurred by RDP for any input parameters.

Theorem 4. *For any $\mathcal{I} = \langle C, \{p_{ij}\}_{i \leqslant j}, \{\lambda^t_{ij}\}_{i \leqslant j, t \in [T]} \rangle$, we have $V^{\mathrm{HO}}_\theta(\mathcal{I}) - V^{\mathrm{RDP}}_\theta(\mathcal{I}) = O(\sqrt{\theta})$. Furthermore, if $\{\lambda^t_{ij}\}_{i \leqslant j, t \in [T]}$ satisfies the NEE property, then $V^{\mathrm{HO}}_\theta(\mathcal{I}) - V^{\mathrm{RDP}}_\theta(\mathcal{I}) = O(1)$.*

Theorem 4 states that (a) the gap between HO and RDP is asymptotically upper-bounded by $O(\sqrt{\theta})$, and (b) this gap can be further reduced to a constant if the NEE property is satisfied. Notice that the NEE property in Definition 3 depends only on the base problem, but not the scaling parameter θ.

Here, we present a brief roadmap for the proof of Theorem 4. We first consider a variant of (14) by imposing lower bounds for γ in the last group of constraints. That is, we consider a set of constraints: $\gamma_{uijv} \geqslant \hat{\gamma}_{uijv}$ for all $u \leqslant i \leqslant j \leqslant v$. For each sample path, we decompose the loss between RDP and HO into θT increments, where each increment is characterized by the gap between two objective values of the variants of (14) with different lower bounds for γ. Then, we upper-bound each increment. We show that each increment is uniformly upper-bounded by a constant that depends only on $\{p_{ij}\}$. Meanwhile, we use concentration inequalities to establish that the probability of each increment being strictly positive decreases at a reasonably fast rate. As a result, the sum of these increments scales in the order of $O(\sqrt{\theta})$ in the general case (the first result displayed in Theorem 4). Furthermore, if the NEE property is satisfied, then the decreasing rate becomes exponential, and so the sum of the increments converges to a constant (the second result in Theorem 4).

We point out that the novelty of our policy lies in the design of the dynamic primal (14), which exploits the special structure brought by the assign-to-seat restriction. Note that the maximal sequence formulation leads to a nonmonotonic state transition, which makes our analysis more challenging than those in the previous works, where the number of remaining resources are always nonincreasing (see, e.g., Vera and Banerjee 2019, Banerjee and Freund 2020, and Bumpensanti and Wang 2020). We overcome the challenges by analyzing some unique features of our problem. Particularly, we show that adding (15) does not change the objective value. In the online appendix, we further propose a probabilistic allocation policy and prove that it can also achieve an $O(1)$ asymptotic loss under NEE arrival rates. However, for both the deterministic and probabilistic allocation policies, using the Maximal Sequence Formulation (14) is vital for the analysis because the size of its action space associated with each request is uniformly controlled by M and is *not* related to θ or N. In contrast, directly applying (2) would lead to an unbounded action space as the problem scales. Therefore, although the re-solving idea has been discussed in the literature, our analysis is uniquely designed for the setting considered.

Now, a question arises naturally: Are the results in Theorem 4 "tight"? More precisely, when the NEE property is not satisfied, is the gap of $O(\sqrt{\theta})$ tight? Several previous works (e.g., Jasin and Kumar 2013, Vera and Banerjee 2019, Banerjee and Freund 2020, and Bumpensanti and Wang 2020) have also obtained $O(1)$ asymptotic loss compared with HO, by assuming that the arrival rates either are stationary or satisfy some structural properties. In what follows, we give an affirmative answer to the question just posed. We show that the gap is *not* caused by the design of our RDP policy. Under the current asymptotic regime, in fact, there could exist an intrinsic $\Omega(\sqrt{\theta})$ gap between HO and DP if the sales of different itineraries do not end at the same time. As a result, the gap between HO and RDP scales in the order of $\Omega(\sqrt{\theta})$. We illustrate this point through Proposition 1; it states that, in the worst case, the asymptotic gap between HO and DP is lower-bounded by $\Omega(\sqrt{\theta})$.

Proposition 1. *Define an instance $\mathcal{I}_0$ as follows. Let $M = 3$, $n = 1$, $T = 4$, and $C = 1^{N\times M}$. There are two types of requests, $1 \to 2$ and $2 \to 3$. The arrival rates are $\lambda_{12}^1 = \lambda_{12}^2 = 1/2$, $\lambda_{12}^3 = \lambda_{12}^4 = 0$, $\lambda_{23}^1 = \lambda_{23}^2 = 0$, and $\lambda_{23}^3 = \lambda_{23}^4 = 1/2$. The prices are $p_{12} = p_{23}/2 < p_{23}$. Then, $V_\theta^{\mathrm{HO}}(\mathcal{I}_0) - V_\theta^{\mathrm{DP}}(\mathcal{I}_0) = \Omega(\sqrt{\theta})$.*

When constructing $\mathcal{I}_0$, we divide the entire time horizon into two halves. In the first half, only requests $1 \to 2$ arrive with a lower price p_1. In the second half, only requests $2 \to 3$ arrive with a higher price p_2. Evidently, it suffices for the decision maker to consider only the capacity level for leg 2. The decision maker must decide how many $1 \to 2$ requests to accept before observing $2 \to 3$ demands. On one hand, there is a constant possibility that the number of $2 \to 3$ requests is greater than θ, in which case no $1 \to 2$ requests should be accepted under HO. On the other hand, there is likewise a constant possibility that the number of $2 \to 3$ requests is lower than $\theta - \sqrt{\theta}$, in which case at least $\sqrt{\theta}$ of the $1 \to 2$ requests should be accepted under HO. However, because the acceptance decision for $1 \to 2$ must be made before the entire demand is realized, it follows that no matter what decision the seller makes, there is a constant possibility of incurring an $\Omega(\sqrt{\theta})$ loss. Therefore, $V_\theta^{\mathrm{HO}}(\mathcal{I}_0) - V_\theta^{\mathrm{DP}}(\mathcal{I}_0) = \Omega(\sqrt{\theta})$. The same intuition extends to more general cases when the set of arrival rates does *not* satisfy the NEE property. Our example also gives some intuition on why the asymptotic loss can be reduced to $O(1)$ when the NEE property is satisfied. Briefly speaking, if arrival rates are NEE, then the wrong decision made at the beginning by a policy may have the potential to be corrected later. Particularly, if up to time $\theta(T-1)$, the number of accepted requests for each type deviate much compared with the optimal allocation obtained from the HO, then we still have chances to rectify the total number of requests being accepted for each type to compensate the loss incurred previously, given that each type of request has a positive probability to appear in the final θ time periods.

Finally, we remark that the $\Omega(\sqrt{\theta})$ gap discussed previously is not because of the assign-to-seat restriction. After all, in the instance $\mathcal{I}_0$ of Proposition 1, we only need to consider the total capacity for leg 2. Thus, our result highlights an important bottleneck in traditional quantity-based network RM problems. In most previous studies, arrival rates are assumed to be stationary, under which one can achieve an $O(1)$ asymptotic loss (Jasin and Kumar 2013, Bumpensanti and Wang 2020). However, we find that, in general, the NEE property plays a key role in obtaining an asymptotically bounded loss compared with the full-information benchmark. When the set of arrival rates does not satisfy the NEE property, it might not be possible to achieve a bounded loss asymptotically. Therefore, our result is also complementary to the traditional network RM literature.

5. Numerical Experiments

In this section, we use both synthetic and real data to conduct numerical experiments, thereby testing the performance of the dynamic policies proposed in Section 4. In particular, we consider the BPC-S and BPC-M policies described in Section 4.1, as well as the RDP policy in Section 4.2. We start with a description of our experimental settings and some implementation details.

5.1. Settings and Implementation Details

5.1.1. Synthetic Data. In the numerical tests with synthetic data, our goal is to test the theoretical results in Section 4 numerically. For that purpose, we fix $M = 6$ and consider seven groups of parameters with $T = 5N$ for $N \in \{100, 200, 500, 1000, 2000, 5000, 10000\}$. In each parameter setting, the prices of each itinerary are chosen as $p_{ij} = \lfloor 10 \times (j-i+1)^{4/5} \rfloor$. This choice reflects that, in practice, the true prices are usually subadditive in the distance between two stops. For arrival probability, we assume that, in each time period, there is a probability $\lambda_0 = 0.2$ that no passenger arrives. We consider two different cases concerning the arrival probability of each passenger type.

- **Case 1.** *Homogeneous arrival.* $\lambda_{ij}^t \propto 1$, which means all itineraries arrive with equal probability.
- **Case 2.** *Inhomogeneous arrival with shorter itineraries arriving first.* We partition the time horizon into M episodes, where the s^{th} episode ($s = 1, \dots, M$) is $(\lfloor (s-1)T/M \rfloor, \lfloor sT/M \rfloor]$. In episode s, each request of length s arrives with equal probability $0.5/(M+1-s)$; all other types of requests arrive with lower (but homogeneous) probability $0.3/(M(M+1)/2 - M + s - 1)$ (recall that with probability $\lambda_0 = 0.2$, no passenger arrives).

In all of our tests, we set the starting capacity matrix $C = 1^{N \times M}$ and fix the re-solving frequency f as once a time. In each test, we run 100 simulated sample paths.

5.1.2. Real Data. In practice, the decision maker may never know what true arrival probabilities are and, thus, must estimate them using past information. Using real data allows us to answer the following two questions.

1. How do different policies perform in a realistic setting?
2. How do estimation errors affect policy performance?

Through a collaboration with the China Railway High-speed, we have the access to booking information (from August to September 2019) for train G315, which departs daily from Jinan at 10:00 in the morning and arrives in Chongqing at 22:30 in the evening. At the beginning of the selling horizon, the train is always completely unoccupied. There are 13 intermediate stations along the route (i.e., $M = 14$). The booking information is confined to second-class carriages, with a total number of approximately 1,000 seats. The data consist of the booking time, itinerary, and assigned seat of each accepted request, but do not contain any request that was not accepted.

To examine the performance of different policies in real settings, we regard the data for each day as a sample path of sequential requests. For the policies we propose, it is sufficient to estimate $\lambda^{[t,T]}$ rather than λ^t itself. On day d and for any time t_d prior to the end of booking horizon T_d, we take the average over the empirical intensities obtained from a set of days $S_d = \{d-1, d-2, d-7, d-14, d-21\}$ to estimate the intensities on day d (this estimate reflects traffic on recent days and on the same day of the week in the past). More precisely, we have

$$\lambda_{d,ij}^{[t_d,T_d]} = \frac{1}{|S_d|} \sum_{d' \in S_d} \sum_{r=1}^{n_{d'}} \mathbb{1}\{\text{request } r \text{ is } i \to j\} \cdot \mathbb{1}\{\text{the timing of request } r \geqslant T_{d'} - T_d + t_d\}. \quad (16)$$

Here, on day d', there are $n_{d'}$ requests indexed by r. In our numerical experiments, we simply choose T_d as 22:30 on each day d. The advantage of this estimation procedure is that we need not explicitly cut the entire booking horizon into discrete time intervals. We acknowledge that the above estimation method is a rough one. In practice, firms may be able to apply other relevant information to generate more accurate forecasts.

To gain insight on how estimation errors affect performance, we also run the experiments using the true intensities (i.e., assuming we know the arrival patterns of that day in advance). That is, we set

$$\lambda_{d,ij}^{[t_d,T_d]} = \sum_{r=1}^{n_d} \mathbb{1}\{\text{request } r \text{ is } i \to j\} \cdot \mathbb{1}\{\text{the timing of request } r \geqslant t_d\}. \quad (17)$$

In the numerical experiments with real data, we set the prices $\{p_{ij}\}$ to be the real prices for each itinerary. The initial capacity matrix C is chosen as $1^{N \times M}$ for $N \in \{800, 600, 400\}$. The choice of N characterizes different scarcity levels of the resource, with fewer seats corresponding to greater resource scarcity. Because the train's true capacity is close to 1,000 passengers, it follows that an N of {800, 600, 400} corresponds roughly to a 20%, 40%, and 60% increase (respectively) in the arrival intensity. The data contain only accepted requests, so if we use the estimated intensity directly, then most requests would be accepted, in which case it would be difficult to discern how different policies yield different results. It is therefore reasonable to consider these "shrunk capacity" settings. Again, we acknowledge that such a method may induce bias in the demand of different itineraries, as, in reality, longer itineraries may be more likely to be rejected. However, as our primary goal is to compare the performance of different algorithms, rather than to conduct a comprehensive empirical study, we adopt a simple method and leave the question of designing more accurate estimation methods to future studies.

Our experiment runs from 26 August (Monday) to 26 September (Thursday) for a total number of 32 days. Here, we start on 26 August because (a) (16) requires three weeks of data to estimate the intensities, and (b) Monday was selected as the beginning of the testing period. We delete the last four days in September because they are close to the Chinese National Holidays (1–7 October), when travel demand can differ markedly from that during normal times.

5.1.3. Tie-Breaking Rule. Both BPC and RDP policies may face "ties" when two or more different decisions lead to the same maximal value. Our implementation adopts the following tie-breaking rule. Suppose that, in some period t, we are about to accept a request $i \to j$, and there is a set of available seats $\{k_w\}_{w=1,2,...}$ (w is the index) with maximal sequence $[u_w, v_w] \sim C^t_{k_w}$ as candidates (line (6) in Algorithms 1 and 2; line (4) in Algorithm 3), where $u_w \leqslant i \leqslant j \leqslant v_w$. In this case, we select w^* such that $u_{w^*} \geqslant u_w$ for all $w \neq w^*$; and if $u_w = u_{w^*}$ for some w, then $v_w \geqslant v_{w^*}$. We assign the request to a seat that has a maximal sequence $[u_{w^*}, v_{w^*}]$. In addition, for the RDP policy, we prefer accepting to rejecting. In other words, when facing a request $i \to j$, this rule orders all maximal sequences that contain leg i to leg j as

$$[i,j] \prec \cdots \prec [i,M] \prec [i-1,j] \prec \cdots \prec [i-1,M] \prec \cdots \prec [1,j] \\ \prec \cdots \prec [1,M] \prec [0,0],$$

and then selects the smallest one under this order.

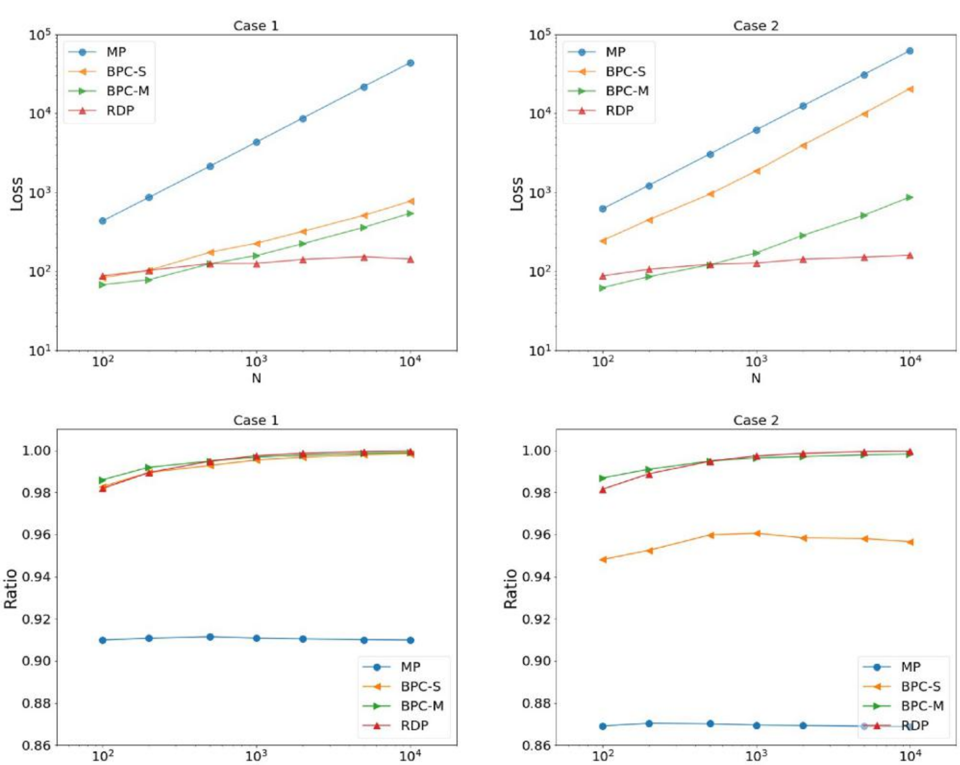
5.1.4. Policies. In addition to testing BPC-S, BPC-M, and RDP, we also consider HO. As before, HO makes an allocation decision only after all requests are known and, thus, achieves the optimal value for the Relaxed Static Model (2). In addition, HO always yields a higher revenue than any other policy and can therefore serve as a benchmark for the performance evaluation. In addition, we consider a myopic policy (MP), where a request is always accepted when there are available seats and the allocation is decided by the aforementioned tie-breaking rule.

Now, we fix a particular scenario. For each policy $\pi \in$ {BPC-S, BPC-M, RDP, HO, MP} and each sample path ω, we record the revenue $\mathrm{rev}^{\pi}(\omega)$ obtained from π. The *loss of* π, denoted as L_π, is defined as the average of $\mathrm{rev}^{\mathrm{HO}}(\omega) - \mathrm{rev}^{\pi}(\omega)$ over all sample paths ω. The *ratio* of π, denoted as R_π, is defined as the average of $\mathrm{rev}^{\pi}(\omega)/\mathrm{rev}^{\mathrm{HO}}(\omega)$ over all sample paths ω. Note that in the synthetic (resp., real) data setting, the number of sample paths for each scenario is 100 (resp., 1).

5.2. Results and Interpretation

5.2.1. Synthetic Data. The results are presented in Figure 2, where we apply a log scale to both the horizontal and vertical axes when plotting L_π for various policies. In both cases, the loss trend is clear when one observes the slope of lines in different markers. The myopic policy performs the worst, as it incurs an asymptotically linear loss.

Figure 2. (Color online) Results Using Synthetic Data



In Case 1, BPC and RDP policies perform much better for both medium- and large-sized problems. Of the two BPC policies, BPC-M performs consistently better than BPC-S. Figure 2 shows that $\log L_{\text{BPC-S}} - \log L_{\text{BPC-M}}$ (i.e., the difference between the logarithm loss of BPC-S and BPC-M) is consistently positive: The loss of BPC-M approximately equals a proportion $\alpha < 1$ of the loss of BPC-S. For the RDP policy, the loss is almost constant as N increases. This finding is consistent with our theoretical result in Section 4.2, which states that the loss of RDP is uniformly bounded.

In Case 2, most of the preceding observations are still valid, except that BPC policies perform very differently. In this challenging case, BPC-S incurs a linear loss with respect to the problem size, whereas BPC-M performs appreciably better. The advantage of BPC-M over BPC-S, which stems from using the maximal sequence structure, is thus confirmed.

Another result worth noticing is that, for medium-sized problems ($N \leqslant 500$), BPC-M may perform better than RDP. This outcome is compatible with our theoretical results, because RDP's bounded-loss property is of the asymptotic type.

5.2.2. Real Data. For the numerical tests with real data, in Figure 3, we plot R_π, the ratio of the revenue achieved by a policy to that achieved by HO, for different policies on all 32 days. Solid lines mark the results based on using estimated parameters in (16) with our proposed policies; dotted lines correspond to those using real parameters in (17). Note that MP does not use past data, and, thus, the dotted line coincides with the solid line. Gray bars are used to mark the optimal revenue rev_{HO} that can be achieved on each day (with the scale on the figures' right-hand axes). We summarize our main findings as follows.

1. *Performance quality.* Both BPC and RDP perform much better than MP. Summing over the 32 days, we calculate the ratio between the total revenue collected by different policies and the optimal policy for different seat numbers. The results are presented in Table 1. Among the BPC policies, BPC-M consistently performs better than BPC-S, generating a 0.5%–2.2% revenue increase.

2. *Parameter sensitivity.* There is a sharp decrease in demand around 15 September, before and after which demand appears to be normal. We can see that BPC policies could perform adaptively to such a change, whereas the RDP performance is unsatisfying. This result could imply that RDP is more sensitive to estimation errors than BPC policies. Meanwhile, it is clear from the dotted lines that RDP always performs the best when we use the real parameters. The implication is that RDP should be the preferred choice under accurate estimation of parameter values.

3. *Resource scarcity.* If the number of seats is large (say, $n = 800$, and so the resource is less scarce), then the performance of RDP is inferior even to BPC-S. However, if the number of seats is relatively small (say, $n = 400$, and so the resource is more scarce), then RDP performs better than BPC-M in most cases. Hence, we conclude that RDP is most useful when demand significantly exceeds resources.

We also examine effects of the assign-to-seat restriction. We test BPC-A, which corresponds to bid-price policies, and RDP-A, which corresponds to the RDP policy. Both BPC-A and RDP-A determine whether to accept or reject a request during each time period, but are not required to assign a seat until the end of the time horizon. According to Theorem 2, if the accepted requests satisfy the total capacity constraints, then one can always find a feasible assignment at the end. Therefore, we can estimate the possible loss caused by the requirement that seats should be assigned immediately. Note that BPC-A and RDP-A are not valid policies in our problem; we consider them only for the purpose of assessing the possible loss due to the assign-to-seat restriction. These two policies are presented formally in Algorithms 4 and 5.

Algorithm 4 (BPC-A Policy)

for $t = 1, \ldots, T$ **do**
  Observe a request of type $i^t \to j^t$;
  Compute the dual prices of (3) with $d = \lambda^{[t,T]}$ as the bid prices for each leg $\{\beta_\ell^t\}$;
  If $p_{i^t j^t} \geqslant \sum_{i_t \leqslant \ell \leqslant j_t} \beta_\ell^t$ and there is enough capacity, then we accept the request.
    Otherwise, we reject the request.
end

Algorithm 5 (RDP-A Policy)

for $t = 1, \ldots, T$ **do**
  Observe a request of type $i^t \to j^t$;
  Solve (3) with $d = \lambda^{[t,T]}$ and obtain an optimal solution $\{x^{\text{RDP-A},t}\}$;
  If $x_{i^t j^t}^{\text{RDP-A},t} \geqslant d_{i^t j^t}^{[t,T]}/2$ (i.e., if accepting the request is more preferred than not) and there is enough capacity, then we accept the request. Otherwise, we reject the request.
end

We use the same set of data to test the performance of BPC-A and RDP-A and compare them with the policies in Section 4. Table 2 summarizes the results. We find that the performance of policies based on the maximal sequence approach (i.e., BPC-M and RDP) are comparable to their counterparts when there is no assign-to-seat restriction. In particular, the performance gap between settings with and without the restriction is less than 1.2%. This gap diminishes and approaches zero as the resource becomes more scarce. This shows that in practical settings, the assign-to-seat restriction has limited effect on system performance under our proposed policies.

Figure 3. (Color online) Results Using Real Data

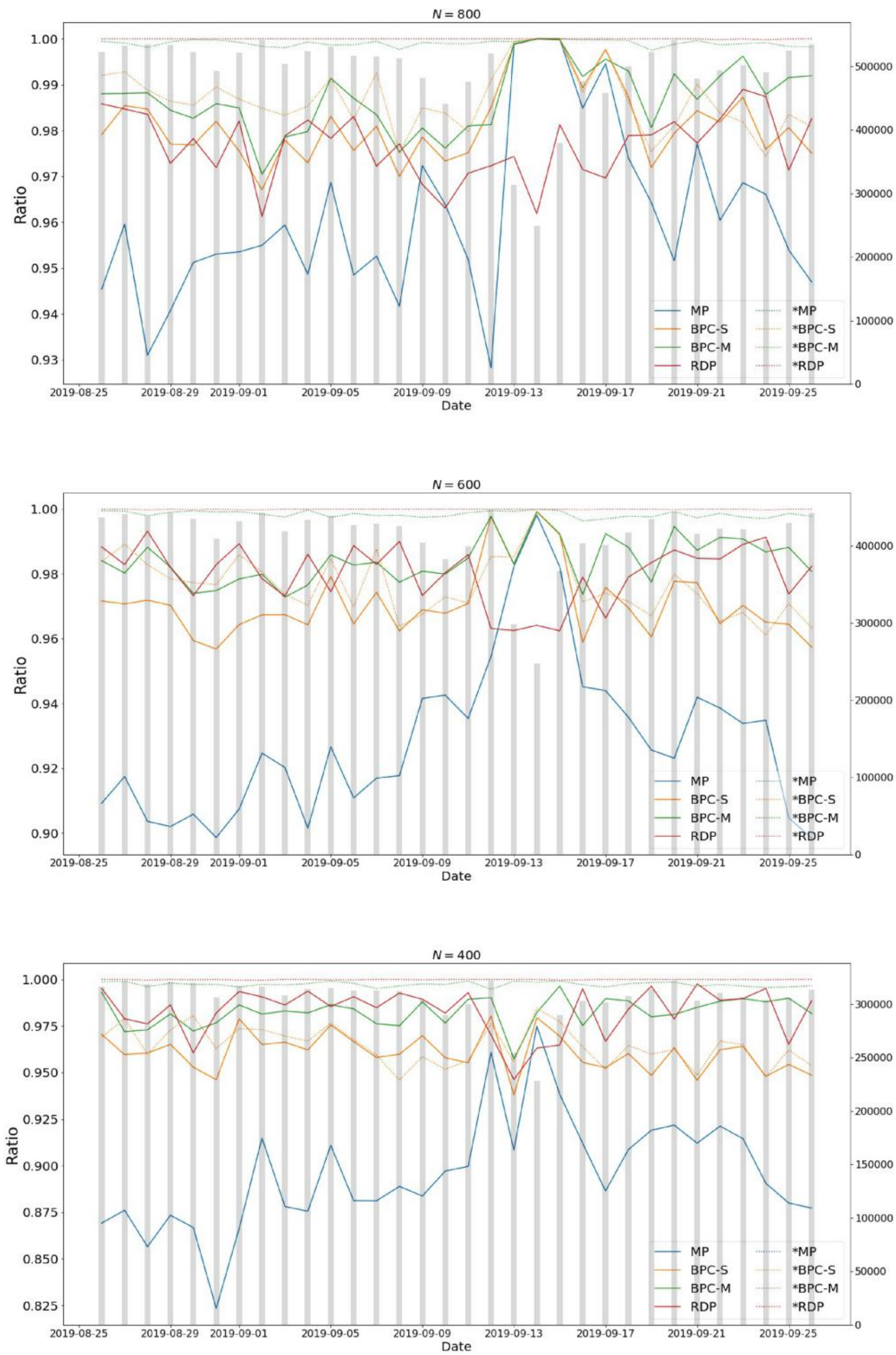


Table 1. Average Performance of Different Policies Compared with the Optimal Policy

N	MP (%)	BPC-S (%)	BPC-M (%)	RDP (%)
800	96.00	98.08	98.66	97.71
600	92.70	97.03	98.39	98.05
400	89.51	96.07	98.26	98.33

5.2.2.1. Summary. Our numerical experiments reveal that BPC-S, BPC-M, and RDP are all effective policies with superior performance to that of the myopic policy for the dynamic capacity-allocation problems studied in this paper. More specifically, BPC-M achieves better performance than BPC-S, and RDP generates bounded loss asymptotically. Both BPC-M and RDP can achieve near-

Table 2. Average Performance of Different Policies Compared with the Optimal Policy

N	MP (%)	BPC-S (%)	BPC-M (%)	BPC-A (%)	RDP (%)	RDP-A (%)
800	96.00	98.08	98.66	98.50	97.71	98.83
600	92.70	97.03	98.39	98.44	98.05	98.77
400	89.51	96.07	98.26	98.37	98.33	98.68

optimal revenues. Moreover, the difference between revenues generated with and without the assign-to-seat restriction is limited. In real applications, for medium-sized problems with a rough parameter estimation, we advocate BPC-M as the preferred policy, whereas for large-sized problems with accurate demand forecasting, RDP is preferable.

6. Conclusion

This paper addresses a dynamic allocation problem that arises from the practice of selling high-speed train tickets. The problem is different from the traditional ones studied in the network revenue management literature, owing to the special feature of real-time seat assignment. After analyzing the static and then the dynamic version of this problem, we propose efficient allocation control policies. Particularly, we introduce a bid-price control policy based on a novel maximal sequence principle. The policy accommodates nonlinearity in bid prices and, as a result, yields a more accurate approximation of the value function than a traditional bid-price control policy does. We also propose a re-solving a dynamic primal policy that can achieve uniformly bounded revenue loss under mild assumptions. Numerical experiments involving real data from the practice show the effectiveness of our proposed approaches.

There are several directions for future research. From the theoretical perspective, it would be interesting to study the asymptotic performance of the proposed bid-price control policies. From a practical perspective, it would be useful to consider simpler and easy-to-implement control policies (e.g., opening and closing certain request types during certain time intervals) and then evaluate their effectiveness. It would be illuminating also to incorporate more features into our model. For example, passengers may have the freedom to select window or aisle seats. Our analysis could be extended to consider such options.

Acknowledgements

The authors thank the department editor, the associate editor, and the referees for their helpful suggestions.

References

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

Agrawal S, Wang Z, Ye Y (2014) A dynamic near-optimal algorithm for online linear programming. *Oper. Res.* 62(4):876–890.

Angelelli E, Bianchessi N, Filippi C (2014) Optimal interval scheduling with a resource constraint. *Comput. Oper. Res.* 51:268–281.

Arkin EM, Silverberg EB (1987) Scheduling jobs with fixed start and end times. *Discrete Appl. Math.* 18(1):1–8.

Armstrong A, Meissner J (2010) Railway revenue management: Overview and models. Department of Management Science, Lancaster University Working Papers, Lancaster, UK.

Aydin N, Birbil SI (2018) Decomposition methods for dynamic room allocation in hotel revenue management. *Eur. J. Oper. Res.* 271(1):179–192.

Banerjee S, Freund D (2020) Uniform loss algorithms for online stochastic decision-making with applications to bin packing. *SIGMETRICS'20 Abstracts 2020 SIGMETRICS/Performance Joint Internat. Conf. Measurement Modeling Comput. Systems* (Association for Computing Machinery, New York), 1–2.

Bar-Noy A, Bar-Yehuda R, Freund A, Naor J, Schieber B (2001a) A unified approach to approximating resource allocation and scheduling. *J. ACM* 48(5):1069–1090.

Bar-Noy A, Guha S, Naor J, Schieber B (2001b) Approximating the throughput of multiple machines in real-time scheduling. *SIAM J. Comput.* 31(2):331–352.

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

Bhatia R, Chuzhoy J, Freund A, Naor JS (2007) Algorithmic aspects of bandwidth trading. *ACM Trans. Algorithms* 3(1):10–19.

Bouzina KI, Emmons H (1996) Interval scheduling on identical machines. *J. Global Optim.* 9(3–4):379–393.

Brucker P, Nordmann L (1994) The *k*-track assignment problem. *Computing* 52(2):97–122.

Bumpensanti P, Wang H (2020) A re-solving heuristic with uniformly bounded loss for network revenue management. *Management Sci.* 66(7):2993–3009.

Carlisle MC, Lloyd EL (1995) On the *k*-coloring of intervals. *Discrete Appl. Math.* 59(3):225–235.

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

Chuzhoy J, Ostrovsky R, Rabani Y (2006) Approximation algorithms for the job interval selection problem and related scheduling problems. *Math. Oper. Res.* 31(4):730–738.

Ciancimino A, Inzerillo G, Lucidi S, Palagi L (1999) A mathematical programming approach for the solution of the railway yield management problem. *Transportation Sci.* 33(2):168–181.

Cooper WL (2002) Asymptotic behavior of an allocation policy for revenue management. *Oper. Res.* 50(4):720–727.

Cross RG (1997) *Revenue Management: Hard-Core Tactics for Market Domination.* (Crown Business, New York).

Devanur NR, Hayes TP (2009) The AdWords problem: Online keyword matching with budgeted bidders under random permutations. *Proc. 10th ACM Conf. Electronic Commerce* (Association for Computing Machinery, New York), 71–78.

Eliiyi DT, Azizoglu M (2006) Spread time considerations in operational fixed job scheduling. *Internat. J. Production Res.* 44(20):4343–4365.

Erlebach T, Spieksma FCR (2003) Interval selection: Applications, algorithms, and lower bounds. *J. Algorithms* 46(1):27–53.

Faigle U, Kern W, Nawijn WM (1999) A greedy on-line algorithm for the *k*-track assignment problem. *J. Algorithms* 31(1):196–210.

Gallego G, van Ryzin G (1997) A multiproduct dynamic pricing problem and its applications to network yield management. *Oper. Res.* 45(1):24–41.

Global Times (2018) Beijing-Shanghai high-speed rail makes $4.4B. Accessed January 31, 2023, https://www.globaltimes.cn/content/1168123.shtml.

Global Times (2019) China's high-speed rail carries record 10 billion passengers. Accessed May 13, 2019, http://en.people.cn/n3/2019/0513/c90000-9577296.html.

Goldman P, Freling R, Pak K, Piersma N (2002) Models and techniques for hotel revenue management using a rolling horizon. *J. Revenue Pricing Management* 1(3):207–219.

Im S, Wang Y (2011) Secretary problems: Laminar matroid and interval scheduling. *Proc. Twenty-Second Annu. ACM-SIAM Sympos. Discrete Algorithms* (Society for Industrial and Applied Mathematics, Philadelphia), 1265–1274.

Jasin S, Kumar S (2012) A re-solving heuristic with bounded revenue loss for network revenue management with customer choice. *Math. Oper. Res.* 37(2):313–345.

Jasin S, Kumar S (2013) Analysis of deterministic LP-based booking limit and bid price controls for revenue management. *Oper. Res.* 61(6):1312–1320.

Kesselheim T, Tönnis A, Radke K, Vöcking B (2014) Primal beats dual on online packing LPs in the random-order model. *Proc. Forty-Sixth Annu. ACM Sympos. Theory Comput.* (Association for Computing Machinery, New York), 303–312.

Kolen AWJ, Kroon LG (1993) On the computational complexity of (maximum) shift class scheduling. *Eur. J. Oper. Res.* 64(1):138–151.

Kolen AWJ, Lenstra JK, Papadimitriou CH, Spieksma FCR (2007) Interval scheduling: A survey. *Naval Res. Logist.* 54(5):530–543.

Kovalyov MY, Ng CT, Cheng TCE (2007) Fixed interval scheduling: Models, applications, computational complexity and algorithms. *Eur. J. Oper. Res.* 178(2):331–342.

Kunnumkal S, Talluri K (2015) On a piecewise-linear approximation for network revenue management. *Math. Oper. Res.* 41(1):72–91.

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

Liu S, Lai K, Wang S (2008) Booking models for hotel revenue management considering multiple-day stays. *Internat. J. Revenue Management* 2(1):78–91.

Maglaras C, Meissner J (2006) Dynamic pricing strategies for multiproduct revenue management problems. *Manufacturing Service Oper. Management* 8(2):136–148.

Molinaro M, Ravi R (2013) The geometry of online packing linear programs. *Math. Oper. Res.* 39(1):46–59.

Murayama H (2017) China pays a high price for world's fastest train. Accessed October 6, 2019, https://asia.nikkei.com/Economy/China-pays-a-high-price-for-world-s-fastest-train.

Nadarajah S, Lim YF, Ding Q (2015) Dynamic pricing for hotel rooms when customers request multiple-day stays. Preprint, submitted August 4, https://dx.doi.org/10.2139/ssrn.2639188.

Reiman MI, Wang Q (2008) An asymptotically optimal policy for a quantity-based network revenue management problem. *Math. Oper. Res.* 33(2):257–282.

Schmidt G (2000) Scheduling with limited machine availability. *Eur. J. Oper. Res.* 121(1):1–15.

Secomandi N (2008) An analysis of the control-algorithm re-solving issue in inventory and revenue management. *Manufacturing Service Oper. Management* 10(3):468–483.

Seiden SS (1998) Randomized online interval scheduling. *Oper. Res. Lett.* 22(4–5):171–177.

Shalom M, Voloshin A, Wong PW, Yung FC, Zaks S (2014) Online optimization of busy time on parallel machines. *Theoret. Comput. Sci.* 560:190–206.

Spieksma FCR (1999) On the approximability of an interval scheduling problem. *J. Scheduling* 2(5):215–227.

Talluri K, van Ryzin G (1998) An analysis of bid-price controls for network revenue management. *Management Sci.* 44(1):1577–1593.

Tong C, Topaloglu H (2013) On the approximate linear programming approach for network revenue management problems. *INFORMS J. Comput.* 26(1):121–134.

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

Vera A, Banerjee S (2019) The Bayesian prophet: A low-regret framework for online decision making. *Performance Eval. Rev.* 47(1):81–82.

Williamson EL (1992) Airline network seat inventory control: Methodologies and revenue impacts. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

Zhang D (2011) An improved dynamic programming decomposition approach for network revenue management. *Manufacturing Service Oper. Management* 13(1):35–52.

Zhang D, Adelman D (2009) An approximate dynamic programming approach to network revenue management with customer choice. *Transportation Sci.* 43(3):381–394.