Management Science

Dynamic Assortment Optimization for Reusable Products with Random Usage Durations

Paat Rusmevichientong, Mika Sumida, Huseyin Topaloglu

To cite this article:
Paat Rusmevichientong, Mika Sumida, Huseyin Topaloglu (2020) Dynamic Assortment Optimization for Reusable Products with Random Usage Durations. Management Science 66(7):2820-2844. https://doi.org/10.1287/mnsc.2019.3346

informs
http://pubsonline.informs.org/journal/mnsc

MANAGEMENT SCIENCE
Vol. 66, No. 7, July 2020, pp. 2820–2844
ISSN 0025-1909 (print), ISSN 1526-5501 (online)

Dynamic Assortment Optimization for Reusable Products with Random Usage Durations

Paat Rusmevichientong,[a] Mika Sumida,[b] Huseyin Topaloglu[b]

[a] Marshall School of Business, University of Southern California, Los Angeles, California 90089; [b] School of Operations Research and Information Engineering, Cornell Tech, New York, New York 10044

Contact: rusmevic@marshall.usc.edu, http://orcid.org/0000-0001-9584-4203 (PR); ms3268@cornell.edu (MS); ht88@cornell.edu (HT)

Received: December 10, 2017
Revised: November 14, 2018
Accepted: March 5, 2019
Published Online in Articles in Advance: January 29, 2020

https://doi.org/10.1287/mnsc.2019.3346

Abstract. We consider dynamic assortment problems with reusable products, in which each arriving customer chooses a product within an offered assortment, uses the product for a random duration of time, and returns the product back to the firm to be used by other customers. The goal is to find a policy for deciding on the assortment to offer to each customer so that the total expected revenue over a finite selling horizon is maximized. The dynamic-programming formulation of this problem requires a high-dimensional state variable that keeps track of the on-hand product inventories, as well as the products that are currently in use. We present a tractable approach to compute a policy that is guaranteed to obtain at least 50% of the optimal total expected revenue. This policy is based on constructing linear approximations to the optimal value functions. When the usage duration is infinite or follows a negative binomial distribution, we also show how to efficiently perform rollout on a simple static policy. Performing rollout corresponds to using separable and nonlinear value function approximations. The resulting policy is also guaranteed to obtain at least 50% of the optimal total expected revenue. The special case of our model with infinite usage durations captures the case where the customers purchase the products outright without returning them at all. Under infinite usage durations, we give a variant of our rollout approach whose total expected revenue differs from the optimal by a factor that approaches 1 with rate cubic-root of Cmin, where Cmin is the smallest inventory of a product. We provide computational experiments based on simulated data for dynamic assortment management, as well as real parking transaction data for the city of Seattle. Our computational experiments demonstrate that the practical performance of our policies is substantially better than their performance guarantees and that performing rollout yields noticeable improvements.

History: Accepted by Yinyu Ye, optimization.
Funding: PR and HT received financial support from the National Science Foundation (NSF) Division of Civil, Mechanical and Manufacturing Innovation [Grants CMMI-1824860 and CMMI-1825406]. MS was supported by an NSF Graduate Research Fellowship. HT was partially funded by Schmidt Sciences.
Supplemental Material: The online appendix is available at https://doi.org/10.1287/mnsc.2019.3346.

Keywords: dynamic assortment optimization • reusable products • choice modeling

1. Introduction

Revenue management problems focus on making capacity-allocation decisions for limited inventories of products over a finite selling horizon. These problems have applications in areas as diverse as airline, hotel, electric power, healthcare, consumer credit, cruise line, and advertising capacity management (Ozer and Phillips 2012). The dynamic-programming formulations of revenue management problems are generally intractable because they require high-dimensional state variables that keep track of the remaining inventory of each product. Thus, computing the optimal policy is computationally difficult, so researchers have focused on approximate policies. In traditional application areas of revenue management problems, the customers purchase the products for final consumption. Some emerging industries, however, focus on renting out products such as computing capacity, fashion items, and storage units. In these industries, each customer requests a product, uses the product for a possibly random duration of time, and returns the product back to the firm, at which point the product can be used by other customers. For example, firms such as Amazon and Google offer cloud-computing services, where users utilize computing capacity for a certain duration of time before returning it. The firm needs to decide what type of computing capacity to offer to each user and what prices to charge. Firms such as Rent the Runway and GlamCorner rent fashion items to shoppers through their online platforms. The firm needs to decide which assortment of fashion items to offer to each shopper and at what prices. Firms such as CubeSmart and MakeSpace lease storage units, where customers return the leased storage units back after using them for a certain duration of time. The firm needs to decide what prices to charge for the storage units as a function of the current occupancy. Using real-time information on the availability of street parking spaces, city governments have the opportunity to dynamically adjust the price for parking spaces, where each driver uses a parking space for a certain duration of time before leaving and making it available for other drivers. When making capacity management decisions in such environments, the firm must consider the on-hand product inventories, as well as the products that are currently in use.

In this paper, we consider dynamic assortment problems with reusable products. In our problem setting, we have access to a set of products with limited inventories. Customers randomly arrive into the system. Among the products for which we currently have available units on-hand, we offer an assortment of products to the arriving customer. The customer either chooses a product from the offered assortment or decides to leave the system. If the customer chooses a product, then she uses the product for a random duration of time. After a usage duration, the customer returns the product. The returned product can be used to offer an assortment to another customer in the future. The goal is to find a policy for deciding on the assortment to offer to each customer so that the total expected revenue over a finite selling horizon is maximized.

Our dynamic-programming formulation of the problem allows for a broad class of choice models for describing the choice process of the customers, nonstationarities in the revenue structure, and arbitrary distributions for the random usage durations. In our formulation, the randomness in the usage duration is not resolved until the customer returns the rented product back, but we can also modify our formulation to address the case where the usage duration is revealed before the firm makes its assortment offering decision. To our knowledge, our model is the first revenue management model with limited inventories of reusable resources, where the customers can choose among the offered products according to a broad class of choice models, there can be nonstationarities in the revenue structure, and the distributions of the usage durations can be arbitrary. The dynamic-programming formulation of the problem requires a high-dimensional state variable that keeps track of the inventories of the products that are available on-hand, as well as the products that are currently in use. Therefore, finding the optimal policy is computationally difficult. We propose tractable policies that provide performance guarantees.

1.1. Main Contributions

In Section 3, we provide a method to construct linear approximations to the optimal value functions in the dynamic-programming formulation of the problem. Our method uses an efficient backward recursion over the time periods in the selling horizon. At each time period, we solve a static assortment problem, where we adjust the product revenues by time-dependent constants computed from the recursion and find an assortment of products that maximizes the expected adjusted revenue from a customer (Section 3.1). We show that the greedy policy with respect to our linear value function approximations is guaranteed to obtain at least 50% of the optimal total expected revenue (Section 3.2), but this policy turns out to be agnostic to inventory levels. Specifically, whether this policy offers a particular product at a particular time period does not depend on the exact on-hand inventory level of this product, as long as on-hand inventory is available. To remedy this shortcoming, we construct separable and nonlinear value function approximations, as discussed next.

In Section 4, we perform rollout on a static policy to construct separable and nonlinear value function approximations. We start with a static policy that offers the same assortment at a particular time period, irrespective of the state of the system (Section 4.1). We compute the total expected revenue obtained by the static policy, starting at each state and at each time period, which can be done by focusing on each product separately. We use these total expected revenues as the value function approximations at different states and at different time periods. In this way, we obtain separable and nonlinear value function approximations. In rollout, we use the greedy policy with respect to these value function approximations (Bertsekas and Tsitsiklis 1996). We show that the policy obtained through the rollout approach is also guaranteed to yield at least 50% of the optimal total expected revenue (Section 4.2). This policy is not agnostic to inventory levels, unlike the greedy policy with respect to linear value function approximations. We demonstrate that we can efficiently perform rollout when the usage duration follows a negative binomial distribution (Section 4.3) or when the usage duration is infinite (Section 4.4).

The case with infinite usage durations corresponds to the situation where the customers purchase the product outright, rather than renting. Under infinite usage durations, we also tailor our rollout approach to strengthen the performance guarantee. In particular, we use $C_{\min}$ to denote the smallest inventory of a product and R to denote the largest relative deviation of the price of a product over the selling horizon. If the prices of the products are stationary, then we have $R = 1$. We show how to construct separable and nonlinear value function approximations under infinite usage durations such that the greedy policy with respect to these value function approximations is guaranteed to obtain at least $\max\{\frac{1}{2}, 1 - \frac{R}{2\sqrt[3]{C_{\min}}}\}$ fraction of the optimal total expected revenue. Therefore, the tailored policy provides at least a half-approximate performance guarantee, but as the inventories of the products become large, the tailored policy becomes near-optimal. Our dynamic assortment problem with infinite usage durations corresponds to the choice-based revenue management problem over parallel flight legs operating between the same origin–destination pair, which is an important problem class that has been studied in the literature (Zhang and Cooper 2005, Liu and van Ryzin 2008, Dai et al. 2014).

In Section 5, we provide extensions of our results. We extend our approach to the case with multiple types of customers, each of whom makes choices according to a different choice model and rents the products according to different usage distributions (Section 5.1). In our setup, we know the type of a customer before offering an assortment, so that we can personalize the assortment according to known customer features. Because different customer types can have different usage distributions and we know the type of a customer before offering an assortment, this extension allows us to capture the case where the usage duration is revealed before we offer an assortment to a customer. We also extend our approach to the cases where we set the prices of the products rather than choosing an assortment to offer (Section 5.2) and when we only approximately solve the assortment-optimization problems used in the construction of our value function approximations (Section 5.3).

In Section 6, we provide computational experiments. We formulate a linear program that yields an upper bound on the optimal total expected revenue (Section 6.1). In our first set of computational experiments, we consider dynamic assortment management, where we offer an assortment of products to each arriving customer (Section 6.2). Our policies perform remarkably well when compared with the upper bound on the optimal total expected revenue and yield average improvements of 1%–10% when compared with other benchmarks. In our second set of computational experiments, we consider the problem of dynamically adjusting the prices for street parking spaces (Section 6.3). We treat each parking space as a reusable product with a random usage duration. Using data from the city of Seattle to estimate the model parameters, dynamically adjusting the prices improves the total expected revenues by 2%–7% when compared with static pricing.

1.2. Literature Review

We review the recent work on revenue management models with reusable products. Levi and Radovanovic (2010) study a model that assumes independent demands across products, without any choice behavior for the customers. Focusing on the infinite selling horizon setting with stationary demand, the authors establish a performance guarantee for a static policy that does not consider the real-time state of the system. Owen and Simchi-Levi (2017) extend this work to incorporate customer choice behavior and a finite selling horizon. The authors assume that the usage durations are exponentially distributed and note that this assumption can be relaxed under a stationary customer choice process. They develop a policy that is guaranteed to obtain $1/e$ fraction of the optimal total expected revenue. This policy may offer products for which there is no on-hand inventory. If the customer chooses a product for which there is no on-hand inventory, then she must leave without making a purchase. The policy in their paper is also static because it offers each assortment of products with a fixed probability that does not depend on the real-time state of the system. By contrast, our model can accommodate arbitrary distributions for the random usage durations and arbitrary nonstationarities in the choice process of the customers. The policy that we construct takes the real-time state of the system into consideration. As long as the choice process of the customers is governed by a random utility maximization principle, our policy never offers products for which there is no on-hand inventory.

Lei and Jasin (2016) develop a model with reusable resources, deterministic usage durations, and advance reservations. Their model includes multiple resources, and each product uses a different combination of resources. The authors give a data-dependent performance guarantee and show that their model is asymptotically optimal when the resource inventories and the number of time periods in the selling horizon scale up linearly at the same rate. Chen et al. (2017) study a model with multiple units of a single reusable product, random usage durations, and advance reservations. Their model allows random usage durations, but the usage duration is revealed at the time of the reservation. The authors also provide a data-dependent performance guarantee and show that their model is asymptotically optimal when the product inventory and the customer arrival rate scale up linearly at the same rate. In our work, we do not allow advance reservations, but we provide constant-factor performance guarantees that are not dependent on the problem data, work with arbitrary usage duration distributions, and allow the randomness in the usage durations to be resolved when the customer returns the product.

Golrezaei et al. (2014) study dynamic assortment problems with nonreusable products. In essence, their model is a special case of ours with infinite usage durations. Considering the case with multiple customer types, the authors construct a policy that is guaranteed to obtain at least 50% of the optimal total expected revenue, even when the sequence of customer type arrivals is chosen by an adversary. As the product inventories become arbitrarily large, the performance guarantee of their policy improves from 50% to $1 - 1/e$. When the type of a customer is drawn from a stationary distribution over the time periods, the performance guarantee further improves to 75%. The key idea in this work is to adjust revenue from the sale of each product by a revenue modifier, which is an increasing function of the current inventory of the product. The policy offers the assortment that maximizes the expected adjusted revenue from each customer. As the inventory of a product is depleted, its adjusted revenue decreases, and the policy is more likely not to offer this product. The proof of the performance guarantee in Golrezaei et al. (2014) is based on formulating a deterministic linear-programming approximation and constructing a dual feasible solution to the approximation by using the revenue modifier. We note that it is possible to formulate a similar linear program under reusable products, but this linear program has a capacity constraint for each product and at each time period, thus ensuring that the expected number of products that are on-hand and in use at any time period will not exceed the product inventory. The dual of this linear program is substantially more complicated, and it is not clear how to construct a feasible dual solution by using the revenue modifier.

Considering the problem setting in Golrezaei et al. (2014), we can tailor our results in this paper to nonreusable products to obtain stronger performance guarantees. Under nonreusable resources, we can give a variant of our rollout approach that is guaranteed to obtain $\max\{\frac{1}{2}, 1 - \frac{R}{2\sqrt[3]{C_{\min}}}\}$ fraction of the optimal total expected revenue, where $C_{\min}$ and R are as discussed earlier in this section. If, for example, the prices of the products are stationary and each product has at least 100 units of inventory, then this performance guarantee computes to be 89%. Similar to the policy in Golrezaei et al. (2014), all of our policies are based on adjusting the revenue from the sale of each product by a revenue modifier. The revenue modifiers in Golrezaei et al. (2014) are multiplicative, whereas our revenue modifiers are additive. The construction of the revenue modifiers in Golrezaei et al. (2014) only uses the current and initial inventory levels, whereas the construction of our revenue modifiers uses all of the problem data. Thus, the revenue modifier in Golrezaei et al. (2014) is robust, as it is insensitive to a large part of the problem data. Our computational experiments, however, indicate that using all of the problem data pays off, and our policies can perform noticeably better than the one in Golrezaei et al. (2014). Motivated by the online resource-allocation setting, Stein et al. (2019), Wang et al. (2016), and Gallego et al. (2016) also consider problems that involve allocating products to customers arriving over time and provide policies with performance guarantees, but this stream of work does not deal with reusable products either.

In the work discussed so far, the initial inventories of the products are exogenously given. There is work on computing stocking quantities at the beginning of the selling horizon when the customers arriving over time choose among the products according to a certain choice model. The paper by van Ryzin and Mahajan (1999) gives a model to compute the optimal stocking quantities under the assumption that a customer can choose a product for which there is no on-hand inventory, in which case she leaves without a purchase, possibly resulting in a penalty or emergency procurement cost. Other work considers the case where a customer chooses only among the products for which there is on-hand inventory. Mahajan and van Ryzin (2001) use stochastic descent to compute stocking quantities without a performance guarantee. Honhon et al. (2010) use a choice model based on ranked preference lists and compute the optimal stocking quantities through a dynamic program, whose running time is exponential in the number of products. Under ranked preference lists, Aouad et al. (2019) give approximation algorithms, whereas Goyal et al. (2016) give polynomial-time approximation schemes. Under the multinomial logit model, Aouad et al. (2018) provide approximation algorithms. These papers do not consider reusable products.

Finally, our work is related to revenue management problems under customer choice. Zhang and Cooper (2005) compute upper bounds on the optimal value functions for the choice-based parallel-flights problem. Gallego et al. (2004) focus on network revenue-management problems and study static policies extracted from a deterministic linear program. Adelman (2007) constructs linear value function approximations when customers arrive into the system to purchase fixed products without a choice process. His approach yields upper bounds on the optimal value functions, but without a performance guarantee. Tong and Topaloglu (2013) show that the approach in Adelman (2007) boils down to solving a linear program whose dimensions increase only linearly with the numbers of itineraries, flights, and time periods. Liu and van Ryzin (2008) develop dynamic-programming decomposition methods for decomposing the dynamic-programming formulation of the network revenue management problem by the flight legs. The authors obtain separable and nonlinear value function approximations, also without a performance guarantee. Zhang and Adelman (2009) and Vossen and Zhang (2015) extend the work of Adelman (2007) to include a customer choice process. Their approach requires solving a linear program whose number of constraints increases exponentially with the number of itineraries. Therefore, the linear program is solved by using column generation. We can solve the column generation subproblem efficiently under some choice models, but there is no a priori bound on the number of columns that need to be generated to obtain the optimal solution. Overall, the work discussed in this paragraph constructs linear and nonlinear value function approximations, but without performance guarantees.

1.3. Organization

In Section 2, we provide a dynamic-programming formulation for our dynamic assortment problem with reusable products. In Section 3, we design a policy that is guaranteed to obtain at least 50% of the optimal total expected revenue. This policy uses linear value function approximations. In Section 4, we use rollout on a static policy to obtain separable and nonlinear value function approximations. The resulting policy is also guaranteed to obtain at least 50% of the optimal total expected revenue. In Section 5, we describe extensions to the cases where we have multiple customer types, we make pricing decisions, and we can solve the assortment problems with errors. In Section 6, we give our computational experiments. In Section 7, we conclude.

2. Problem Formulation

We have a set of products with limited inventories. At each time period in the selling horizon, we decide on the set of products to offer. A customer arriving into the system either chooses to rent one of the offered products or decides to leave the system without renting anything. We capture the choice process of the customers through a discrete choice model. If the customer chooses to rent one of the offered products, then she uses the product for a random duration of time by paying an upfront fee and a per-period rental fee for each time period that she uses the product. After using the product for a random duration of time, the customer returns the product, at which point we can rent the product to another customer. Our goal is to find a policy for maximizing the total expected revenue over the selling horizon. We describe the problem data, state, and transition dynamics, followed by a dynamic-programming formulation of the problem.

2.1. Problem Data

We have n products indexed by $\mathcal{N} = \{1, \ldots, n\}$. For each product $i \in \mathcal{N}$, let $C_i \in \mathbb{Z}_+$ denote its initial inventory level. There are T time periods in the selling horizon indexed by $\mathcal{T} = \{1, \ldots, T\}$. Each time period corresponds to a small interval of time, and there is exactly one customer arrival at each time period. It is not difficult to extend our model to the case with at most one customer arrival at each time period. If we offer the subset of products S, then a customer arriving at time period t chooses product i with probability $\phi_i^t(S)$. Naturally, we have $\phi_i^t(S) = 0$ for all $i \notin S$. If a customer chooses to rent product i at time period t, then she immediately pays a one-time upfront fee of r_i^t. If a customer is renting product i during time period t, then she also pays a per-period rental fee of π_i^t. Depending on the specific application at hand, one of the fees can be zero; in addition, one or both of the fees can be stationary. The per-period rental fee can also depend on how long the product has been in use.

We use the generic random variable $\mathsf{Duration}_i$ to represent the random rental duration of product i. The random variable $\mathsf{Duration}_i$ has a probability mass function $f_i : \mathbb{Z}_{++} \mapsto [0, 1]$, where $\sum_{\ell=1}^{\infty} f_i(\ell) = 1$. We describe the rental duration in terms of its hazard rate $\rho_{i,\ell}$ associated with the probability mass function f_i, where for each $\ell \in \mathbb{Z}_+$, we have

$$\rho_{i,\ell} = \Pr\{\mathsf{Duration}_i = \ell + 1 \,|\, \mathsf{Duration}_i > \ell\} = \frac{f_i(\ell+1)}{\sum_{s=\ell+1}^{\infty} f_i(s)}.$$

The hazard rate $\rho_{i,\ell}$ is the probability that each unit of product i is returned after $\ell + 1$ periods, given that the product has been used for more than ℓ periods. Because $\sum_{s=1}^{\infty} f_i(s) = 1$, we have $\rho_{i,0} = f_i(1)$, so that $\rho_{i,0}$ is the probability that a unit of product i is used for exactly one time period. The usage durations of different units are assumed to be independent of each other.

At each time period t, the following sequence of events happen. We observe the state of the system, which consists of the current on-hand units and the outstanding units that are currently being rented by the customers. Based on the state, we decide which subset of products to offer. The customer arriving at time period t chooses a unit to rent or leaves the system without renting. We collect the upfront fee for the rented unit and the rent from all customers still using their rented units. Finally, we observe whether each customer with a rented unit of product decides to return the unit, including the customer who rented a unit at the current time period.

2.2. State and Transition Dynamics

To capture the state of the system at a generic time period, let $q_{i,0}$ denote the number of units of product i available as on-hand inventory at the beginning of the time period. Also, for $\ell \geq 1$, let $q_{i,\ell}$ denote the number of units of product i that have been used for exactly ℓ time periods at the beginning of the time period. Thus, we describe the state of the system by the vector $\boldsymbol{q} = (q_{i,\ell} : i \in \mathcal{N}, \ell = 0, 1, \ldots)$. For example, if $\boldsymbol{q}$ represents the state of the system at the beginning of time period t, then $q_{i,1}$ is the number of units of product i rented at time period $t-1$ and not returned by the beginning of time period t. Because $\sum_{\ell=0}^{\infty} q_{i,\ell} = C_i$, let $\mathcal{Q} = \{(q_{i,\ell} : i \in \mathcal{N}, \ell = 0,1,\ldots) : \sum_{\ell=0}^{\infty} q_{i,\ell} = C_i \, \forall\, i \in \mathcal{N}\}$ denote the set of all possible states. We assume that the system starts with no units in use. Thus, there will never be a unit in use for more than T time periods, which makes the effective set of possible states finite.

Consider the state $\boldsymbol{q}$ at the beginning of time period t. There are $q_{i,\ell}$ units of product i that have been used for exactly ℓ periods. By definition of the hazard rate, with probability $\rho_{i,\ell}$, each of the $q_{i,\ell}$ units will be returned by the beginning of time period $t+1$. Thus, if no purchase is made at time period t, then the number of units that will be available as on-hand inventory at the beginning of time period $t+1$ is $q_{i,0} + \sum_{\ell=1}^{\infty} \mathsf{Bin}(q_{i,\ell}, \rho_{i,\ell})$, where $\mathsf{Bin}(k,p)$ denotes a binomial random variable with parameters $k \in \mathbb{Z}_+$ and $p \in [0,1]$. Also, at the beginning of time period $t+1$, the number of units of product i that will have been rented out for $\ell+1$ periods will be $q_{i,\ell} - \mathsf{Bin}(q_{i,\ell}, \rho_{i,\ell})$, where the second term reflects the units that will be returned at time period t. Therefore, given the state $\boldsymbol{q}$ at the beginning of time period t, if there is no purchase by a customer, then the state $\boldsymbol{X}(\boldsymbol{q}) = (X_{i,\ell}(\boldsymbol{q}) : i \in \mathcal{N}, \ell = 0,1,\ldots)$ at the beginning of period $t+1$ is given by

$$X_{i,\ell}(\boldsymbol{q}) = \begin{cases} q_{i,0} + \sum_{s=1}^{\infty} \mathsf{Bin}(q_{i,s}, \rho_{i,s}) & \text{if } \ell = 0, \\ 0 & \text{if } \ell = 1, \\ q_{i,\ell-1} - \mathsf{Bin}(q_{i,\ell-1}, \rho_{i,\ell-1}) & \text{if } \ell \geq 2. \end{cases} \tag{1}$$

Note that if there is no purchase at time period t, then a unit that was on-hand will stay on-hand at time period $t+1$. Also, a unit that was in use will either be returned or it will stay in use. In the latter case, its usage duration will be at least two time periods. So, we have $X_{i,1}(\boldsymbol{q}) = 0$.

2.3. Dynamic-Programming Formulation

We use $\mathcal{F}$ to denote the collection of feasible subsets of products that we can offer to the customers at each time period, which captures the constraints that we may impose on the offered subset of products. To formulate the problem as a dynamic program, we denote a Bernoulli random variable with parameter $\rho \in [0,1]$ by $\mathsf{Z}(\rho)$. Also, viewing the state $\boldsymbol{q} = (q_{i,\ell} : i \in \mathcal{N}, \ell = 0,1,\ldots)$ as a vector, we let $\boldsymbol{e}_{i,k}$ be a unit vector with one in the (i,k)-th coordinate and zero everywhere else. Let $J^t(\boldsymbol{q})$ denote the maximum total expected revenue over the time periods $t,\ldots,T$, given that the system is in state $\boldsymbol{q}$ at the beginning of time period t. Then, using $\mathbb{1}_{\{\cdot\}}$ to denote the indicator function, we can compute the optimal value functions $\{J^t : t \in \mathcal{T}\}$ by solving the dynamic program

$$\begin{aligned} J^t(\boldsymbol{q}) &= \sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} \\ &+ \max_{S\in\mathcal{F}} \Bigg\{ \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\geq 1\}}\, \phi_i^t(S) \Big(r_i^t + \pi_i^t + \mathbb{E}\Big\{ \mathsf{Z}(\rho_{i,0})\, J^{t+1}\,(\boldsymbol{X}(\boldsymbol{q})) \\ &\qquad + (1 - \mathsf{Z}(\rho_{i,0}))\, J^{t+1}\big(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1}\big)\Big\}\Big) \\ &\qquad + \Big(1 - \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\geq 1\}} \phi_i^t(S)\Big)\, \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\Bigg\}, \end{aligned} \tag{2}$$

with the boundary condition that $J^{T+1} = 0$. In the dynamic-programming formulation above, we implicitly assume that even if $q_{i,0} = 0$, meaning that we do not have any on-hand inventory for product i, we can offer an assortment that includes product i. Note the indicator function; if a customer chooses a product with zero on-hand inventory, then she leaves the system without renting any products. The possibility of offering products with zero on-hand inventory may be unrealistic in certain settings. Shortly in this section, in Assumption 2.1, we impose rather mild assumptions on the discrete choice model $\{\phi_i^t(S) : i \in \mathcal{N}, S \subseteq \mathcal{N}\}$ and the set of feasible decisions $\mathcal{F}$ to ensure that the optimal policy never offers a product with zero on-hand inventory, even if we are allowed to do so. So, under Assumption 2.1, it follows that the dynamic-programming formulation above is equivalent to a dynamic-programming formulation that explicitly imposes a constraint to ensure that we must have nonzero on-hand inventory for each product that we offer.

In the dynamic program in (2), the term $\sum_{i\in\mathcal{N}} \pi_i^t \cdot \sum_{\ell=1}^{\infty} q_{i,\ell}$ captures the rent payments from customers with already rented units at the beginning of time period t. On the other hand, the term $r_i^t + \pi_i^t + \mathbb{E}\{\mathsf{Z}(\rho_{i,0})\, J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) + (1-\mathsf{Z}(\rho_{i,0}))\, J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\}$ corresponds to the expected revenue from a customer who selects product i at time period t. Here, $r_i^t + \pi_i^t$ reflects the one-time upfront payment and the per-period rent for the first rental period. Noting the definition of the hazard rate, we have $\rho_{i,0} = f_i(1)$. Therefore, the Bernoulli random variable $\mathsf{Z}(\rho_{i,0})$ takes a value of 1 if and only if the customer renting a unit of product i at time period t uses the product for exactly one time period. If $\mathsf{Z}(\rho_{i,0}) = 1$, then the unit is returned to the firm at the end of time period t, in which case, the state at the beginning of time period $t+1$ is $\boldsymbol{X}(\boldsymbol{q})$, which is identical to the state that we would have obtained when no rentals were made at time period t. On the other hand, if $\mathsf{Z}(\rho_{i,0}) = 0$, then the selected unit of product i will not be returned at the end of time period t. In this case, the selected unit of product i will not be on-hand at the beginning of time period $t+1$; instead, this unit will be in use for exactly one time period. Therefore, the state of the system at the beginning of time period $t+1$ will be $\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1}$. To simplify our dynamic-programming formulation, note that because the rental durations of different units are independent of each other, $\boldsymbol{X}(\boldsymbol{q})$ and $\mathsf{Z}(\rho_{i,0})$ are independent of each other as well. Therefore, we obtain

$$\begin{aligned}&\mathbb{E}\Big\{\mathsf{Z}(\rho_{i,0})\,J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) + (1-\mathsf{Z}(\rho_{i,0}))\,J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\\ &\quad - \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\} = \rho_{i,0}\,\mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\} + (1-\rho_{i,0})\\ &\quad \cdot \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\} - \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\\ &\quad = -(1-\rho_{i,0})\,\mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\},\end{aligned}$$

in which case, simply by arranging the terms, we can write the dynamic-programming formulation in (2) equivalently as

$$\begin{aligned}J^t(\boldsymbol{q}) &= \sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\\ &\quad + \max_{S\in\mathcal{F}}\Bigg\{\sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi_i^t(S)\Big(r_i^t + \pi_i^t - (1-\rho_{i,0})\\ &\qquad\quad \cdot \mathbb{E}\Big\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big)\Bigg\}.\end{aligned} \tag{3}$$

Note that $J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})$ captures the marginal value of renting one unit of product i to the customer at time period t.

Throughout the paper, we impose a mild assumption on the discrete choice model $\{\phi_i^t(S) : i \in \mathcal{N}, S \subseteq \mathcal{N}\}$ and the set of feasible decisions $\mathcal{F}$ to ensure that the optimal policy never offers a product with zero on-hand inventory. This assumption is given below.

Assumption 2.1 (Substitutability and Feasibility). *Adding more products to an assortment does not increase the selection probability; that is, for all $S \subseteq \mathcal{N}$ and $k \in \mathcal{N}$, $\phi_i^t(S \cup \{k\}) \le \phi_i^t(S)$ for all $i \in S$. In addition, if a set of products is feasible to offer, then so are all of its subsets; that is, if $A \in \mathcal{F}$, then $S \in \mathcal{F}$ for all $S \subseteq A$.*

The first assumption ensures that products are substitutable, and, thus, the probability of choosing any product never increases if more options become available. This assumption is rather mild, and it holds for all choice models that satisfy the random utility maximization principle, including the multinomial logit, nested logit, d-level logit, paired combinatorial logit, and many others. In addition, the second assumption on the collection of feasible subsets $\mathcal{F}$ also holds for a broad class of assortment constraints, such as a shelf-space constraint $\mathcal{F} = \{S \subseteq \mathcal{N} : \sum_{i\in S} c_i \le B\}$, where c_i is the space consumed by product i and B is the total shelf-space available. Under the assumption above, it is not difficult to see that the optimal policy never offers a product with zero on-hand inventory. In the maximization problem in (3), the profit contribution of product i is $\mathbb{1}_{\{q_{i,0}\ge 1\}} \times (r_i^t + \pi_i^t - (1-\rho_{i,0}) \cdot \mathbb{E}\{J^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - J^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\})$. Let S^* be an optimal solution to this maximization problem. In this case, observe that we can drop all products with nonpositive profit contributions from S^* without degrading the objective value of the maximization problem in (3) because, if we drop such products, then by the substitutability assumption, the selection probabilities of all other products increase, whereas by the feasibility assumption, the subset we obtain remains feasible. Thus, the new subset that we obtain in this fashion provides an objective value to the maximization problem in (3) that is at least as large as that provided by S^*. Because the profit contribution of product i is zero when $\mathbb{1}_{\{q_{i,0}\ge 1\}} = 0$, there exists an optimal policy that never offers a product with zero on-hand inventory.

Because all products are available at the beginning of the selling horizon, the optimal total expected revenue is $J^1(\sum_{i\in\mathcal{N}} C_i\, \boldsymbol{e}_{i,0})$. One source of difficulty in computing the optimal value functions $\{J^t : t \in \mathcal{T}\}$ is that the maximization problem in (3) is a combinatorial optimization problem that chooses the set of products to offer. However, there exist efficient algorithms to solve this problem under many discrete choice models, including the multinomial logit (Talluri and van Ryzin 2004, Rusmevichientong et al. 2014), nested logit (Davis et al. 2014, Gallego and Topaloglu 2014), d-level logit (Li et al. 2015), and paired combinatorial logit (Zhang et al. 2017), and under many different types of feasible sets $\mathcal{F}$ (Davis et al. 2013, Feldman and Topaloglu 2015, Desire et al. 2016). Later in the paper, we will also discuss how our results can be extended when we can only approximately solve the maximization problem in (3).

Although we can solve the maximization problem in (3) efficiently, to find the optimal policy, we need to compute the optimal value function $J^t(\boldsymbol{q})$ for each $\boldsymbol{q} \in \mathcal{Q}$ and $t \in \mathcal{T}$. The number of possible states $|\mathcal{Q}|$ grows exponentially with n and T, which makes it difficult to find the optimal policy. Thus, throughout the rest of the paper, we focus on developing approximate policies that are efficient to compute and have provable performance guarantees.

3. Linear Value Function Approximations

We develop an approach to construct linear approximations to the optimal value functions and analyze the performance of a policy that uses these approximations. In particular, we give a tractable recursion to come up with linear value function approximations. We show that if we use the greedy policy with respect to these linear value function approximations, then we obtain a policy that is guaranteed to obtain at least 50% of the optimal total expected revenue.

3.1. Specification of Linear Value Function Approximations

We consider an approximation $\hat{J}^t$ to the optimal value function J^t given by

$$\hat{J}^t(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} \sum_{\ell=0}^{\infty} \hat{v}^t_{i,\ell}\, q_{i,\ell}, \tag{4}$$

where, for $\ell \geq 1$, the parameter $\hat{v}^t_{i,\ell}$ represents the marginal value at time period t of each unit of product i that has been in use for ℓ periods, whereas the parameter $\hat{v}^t_{i,0}$ represents the marginal value of each unit of product i that is currently available as on-hand inventory at time period t. We propose computing $\hat{v}^t_{i,\ell}$ recursively as follows.

- **Initialization:** Set $\hat{v}^{T+1}_{i,\ell} = 0$ for all $i \in \mathcal{N}, \ell \geq 0$.
- **Backward Recursion:** For $t = T, T-1, \dots, 1$, we compute $\hat{v}^t_{i,\ell}$ by using $\{\hat{v}^{t+1}_{i,\ell} : i \in \mathcal{N}, \ell \geq 0\}$ as follows. Let $\hat{A}^t \in \mathcal{F}$ be an assortment such that

$$\hat{A}^t = \arg\max_{S\in\mathcal{F}} \sum_{i\in\mathcal{N}} \phi^t_i(S)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\Big(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\Big)\Big]. \tag{5}$$

Once $\hat{A}^t$ is computed, for each $i \in \mathcal{N}$, let

$$\hat{v}^t_{i,0} = \hat{v}^{t+1}_{i,0} + \frac{1}{C_i}\phi^t_i\big(\hat{A}^t\big)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\Big(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\Big)\Big]$$
$$\hat{v}^t_{i,\ell} = \pi^t_i + \rho_{i,\ell}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1} \qquad \forall\, \ell = 1, 2, \dots. \tag{6}$$

The above description completes the specification of the approximate value function $\hat{J}^t$. We shortly give the intuition behind our approach. Because we start the system with all units available as on-hand inventory, no unit will be in use for more than T time periods. Thus, we only need to compute $\hat{v}^t_{i,\ell}$ for $\ell = 0, 1, \dots, T$, so we can execute the above recursion in finite time.

We provide some intuition into the computation of $\hat{A}^t$. Intuitively speaking, we can interpret $\hat{A}^t$ as an ideal assortment to offer at time period t under the linear value function approximations when we ignore inventory availability. In particular, if we replace the value function J^{t+1} in the maximization problem on the right side of (3) with the linear approximation $\hat{J}^{t+1}(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} \sum_{\ell=0}^{\infty} \hat{v}^{t+1}_{i,\ell}\, q_{i,\ell}$ and drop the indicator function $\mathbb{1}_{\{q_{i,0}\geq 1\}}$ to ignore inventory availability, then the objective function of this maximization problem takes the form $\sum_{i\in\mathcal{N}} \phi^t_i(S)\,[r^t_i + \pi^t_i - (1-\rho_{i,0})(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1})]$, which is the same as the objective function of the maximization problem in (5). Next, we provide some intuition into the computation of $\hat{v}^t_{i,0}$, which measures the value of a unit of on-hand inventory for product i at time period t. Roughly speaking, assume that we offer the ideal assortment $\hat{A}^t$ at time period t, and if a customer selects product i at time period t, then we "direct" the customer to one of the C_i copies of product i with equal probability of $1/C_i$. In this case, the probability that a unit of product i "sees" a demand at time period t is $\phi^t_i(\hat{A}^t)\,\frac{1}{C_i}$. We write the recursion that we use to compute $\hat{v}^t_{i,0}$ in (6) equivalently as

$$\hat{v}^t_{i,0} = \frac{1}{C_i}\phi^t_i\big(\hat{A}^t\big)\Big[r^t_i + \pi^t_i + \rho_{i,0}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,0})\,\hat{v}^{t+1}_{i,1}\Big] + \bigg(1 - \frac{1}{C_i}\,\phi^t_i\big(\hat{A}^t\big)\bigg)\,\hat{v}^{t+1}_{i,0}.$$

On the left side above, $\hat{v}^t_{i,0}$ is the value of a unit of product i on-hand at time period t. If we offer the ideal assortment $\hat{A}^t$ at time period t, then a unit of product i sees a demand with probability $\frac{1}{C_i}\phi^t_i(\hat{A}^t)$. In this case, we collect the upfront fee r^t_i and the rent π^t_i for the first time period. As discussed earlier, with probability $\rho_{i,0} = f_i(1)$, the customer rents product i for exactly one time period, in which case she returns the product by the beginning of time period $t+1$. The value of a unit of on-hand inventory of product i at time period $t+1$ is $\hat{v}^{t+1}_{i,0}$. With probability $1-\rho_{i,0}$, the customer rents product i for more than one time period, in which case the product will have been rented out at the beginning of time period $t+1$ for exactly one period. The value of a unit of product i at time period $t+1$ that has been in use for one period is $\hat{v}^{t+1}_{i,1}$. This discussion provides the intuition for the term $r^t_i + \pi^t_i + \rho_{i,0}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,0})\,\hat{v}^{t+1}_{i,1}$ on the right side above. With probability $1 - \frac{1}{C_i}\phi^t_i(\hat{A}^t)$, the unit of product i does not see a demand, in which case this unit is still available at time period $t+1$, and the value of this unit is given by $\hat{v}^{t+1}_{i,0}$.

We can give a similar intuition for the recursion that is used to compute $\hat{v}^t_{i,\ell}$ for all $\ell = 1, 2, \dots$. Noting the recursion $\hat{v}^t_{i,\ell} = \pi^t_i + \rho_{i,\ell}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}$, recall that $\hat{v}^t_{i,\ell}$ on the left side is the value of a unit of product i that has been in use for ℓ periods at time period t. This unit of product i will certainly be used until the end of time period t, and we will obtain the rental fee of π^t_i. Furthermore, by the definition of the hazard rate $\rho_{i,\ell}$, a unit of product i that has been in use for ℓ periods at time period t will be returned by the beginning of the next time period with probability $\rho_{i,\ell}$, in which case the value of this on-hand unit at time period $t+1$ is $\hat{v}^{t+1}_{i,0}$, yielding the term $\rho_{i,\ell}\,\hat{v}^{t+1}_{i,0}$ on the right side. On the other hand, once again, by the definition of the hazard rate $\rho_{i,\ell}$, a unit of product i that has been in use for ℓ periods at time period t will not be returned by the beginning of time period $t+1$ with probability $1-\rho_{i,\ell}$. Therefore, this unit of product i will have been used for $\ell+1$ periods at the next time period, and the value of this unit at time period $t+1$ is $\hat{v}^{t+1}_{i,\ell+1}$, yielding the term $(1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}$ on the right side.

The discussion in the previous two paragraphs also provides a natural interpretation for our value function approximations. In particular, our value function approximations correspond to the total expected revenue that we obtain when we use a policy that manages each unit of product i independently. Focus on one particular unit of product i. At time period t, we always offer the assortment $\hat{A}^t$, in which case an arriving customer selects product i with probability $\phi_i^t(\hat{A}^t)$. If the customer selects product i, then we direct the customer to each unit of product i with equal probability of $1/C_i$. In this case, the unit of product i that we focus on sees a demand at time period t with probability $\phi_i^t(\hat{A}^t)\,\frac{1}{C_i}$. If the unit of product i that we focus on sees a demand, then it is rented, so we collect the upfront fee of r_i^t and the rent of π_i^t for the first time period. A unit that is rented stays in use for a random duration of time that is governed by the hazard rates $\{\rho_{i,\ell} : \ell \geq 0\}$, during which we collect the rent of π_i^t at each time period t that the unit is in use. After the usage duration has expired, the unit is returned. By the discussion in the previous two paragraphs, if we use a policy that manages each unit of product i independently in the way we just described, then $\hat{v}_{i,\ell}^t$ corresponds to the total expected revenue from a unit of product i that has been in use for exactly ℓ time periods at the beginning of time period t. Therefore, our value function approximations correspond to the value functions of a policy that manages each unit independently. Clearly, this policy does not pool the units of the same product together, so we certainly do not advocate using such a policy in practice. We will only use the value functions of this policy to construct value function approximations. It turns out that the greedy policy with respect to the value function approximations will have a performance guarantee.

Considering the effort to compute the parameters $\{\hat{v}_{i,\ell}^t : i \in \mathcal{N}, \ell = 0, \dots, T, t \in \mathcal{T}\}$, we need to solve problem (5) for each time period $t \in \mathcal{T}$. The number of operations to solve this problem depends on the underlying choice model. We use Opt to denote the number of operations to solve one instance of problem (5). Next, we need to compute $\{\phi_i^t(\hat{A}^t) : i \in \mathcal{N}, t \in \mathcal{T}\}$. The number of operations to compute these choice probabilities also depends on the underlying choice model. We use Prob to denote the number of operations to compute $\{\phi_i^t(S) : i \in \mathcal{N}\}$ for a fixed subset S and time period t. Once we compute $\{\phi_i^t(\hat{A}^t) : i \in \mathcal{N}, t \in \mathcal{T}\}$, we can use (6) to compute each one of the parameters $\{\hat{v}_{i,\ell}^t : i \in \mathcal{N}, \ell = 0, \dots, T, t \in \mathcal{T}\}$ in $O(1)$ operations. Thus, noting that there are $O(T^2 n)$ such parameters, we can compute all of the parameters $\{\hat{v}_{i,\ell}^t : i \in \mathcal{N}, \ell = 0, \dots, T, t \in \mathcal{T}\}$ in $O(T \times \mathsf{Opt} + T \times \mathsf{Prob} + T^2 n)$ operations. For example, if the customers choose according to the multinomial logit model, then we can solve one instance of problem (5) in $O(n \log n)$ operations (Talluri and van Ryzin 2004). Also, for a fixed subset S and time period t, we can compute $\{\phi_i^t(S) : i \in \mathcal{N}\}$ in $O(n)$ operations. In this case, we can compute all of the parameters $\{\hat{v}_{i,\ell}^t : i \in \mathcal{N}, \ell = 0, \dots, T, t \in \mathcal{T}\}$ in $O(Tn \log n + T^2 n)$ operations.

Lastly, although we use linear value function approximations, it is not difficult to see that the optimal value functions are not even separable by the products. In Online Appendix A, we give a problem instance with only one time period in the selling horizon, in which the optimal value functions are not separable by the products. We close this section with the next lemma, where we show that the marginal value of a unit of on-hand inventory becomes smaller as the end of the selling horizon approaches. We will use this property several times throughout the paper.

Lemma 3.1 (Properties of the Marginal Values). *The marginal value of on-hand inventory decreases over time; that is, $\hat{v}_{i,0}^t \geq \hat{v}_{i,0}^{t+1}$ for all $t \in \mathcal{T}$ and $i \in \mathcal{N}$.*

Proof. For notational brevity, let $\Delta_i^t = r_i^t + \pi_i^t - (1 - \rho_{i,0})(\hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1})$. We will shortly show the claim that $\phi_i^t(\hat{A}^t)\,\Delta_i^t \geq 0$ for all $i \in \mathcal{N}$. In this case, by the recursion in (6) that we use to compute $\hat{v}_{i,0}^t$, we have $\hat{v}_{i,0}^t = \hat{v}_{i,0}^{t+} + \frac{1}{C_i}\,\phi_i^t(\hat{A}^t)\,\Delta_i^t \geq v_{i,0}^{t+1}$, which is the desired result. To see the claim that $\phi_i^t(\hat{A}^t)\,\Delta_i^t \geq 0$ for all $i \in \mathcal{N}$, assume on the contrary that there exists some $k \in \mathcal{N}$ such that $\phi_k^t(\hat{A}^t)\,\Delta_k^t < 0$. Let $\mathcal{N}^+ = \{i \in \mathcal{N} : \Delta_i^t \geq 0\}$ and $\mathcal{N}^- = \{i \in \mathcal{N} : \Delta_i^t < 0\}$. By our assumption, there exists some $k \in \mathcal{N}^-$ such that $\phi_k^t(\hat{A}^t) > 0$. Furthermore, by Assumption 2.1, $\phi_i^t(\hat{A}^t \cap \mathcal{N}^+) \geq \phi_i^t(\hat{A}^t)$ for all $i \in \hat{A}^t \cap \mathcal{N}^+$. By the same assumption, because $\hat{A}^t \in \mathcal{F}$, we have $\hat{A}^t \cap \mathcal{N}^+ \in \mathcal{F}$. So, we get

$$\begin{aligned}\sum_{i \in \mathcal{N}} \phi_i^t\left(\hat{A}^t\right)\Delta_i^t &= \sum_{i \in \mathcal{N}^+} \phi_i^t\left(\hat{A}^t\right)\Delta_i^t + \sum_{i \in \mathcal{N}^-} \phi_i^t\left(\hat{A}^t\right)\Delta_i^t \\ &< \sum_{i \in \mathcal{N}^+} \phi_i^t\left(\hat{A}^t\right)\Delta_i^t = \sum_{i \in \hat{A}^t \cap \mathcal{N}^+} \phi_i^t\left(\hat{A}^t\right)\Delta_i^t \\ &\leq \sum_{i \in \hat{A}^t \cap \mathcal{N}^+} \phi_i^t\left(\hat{A}^t \cap \mathcal{N}^+\right)\Delta_i^t = \sum_{i \in \mathcal{N}} \phi_i^t\left(\hat{A}^t \cap \mathcal{N}^+\right)\Delta_i^t,\end{aligned}$$

where the first inequality follows because there exists some $k \in \mathcal{N}^-$ such that $\phi_k^t(\hat{A}^t) > 0$, the second equality holds because $\phi_i^t(\hat{A}^t) = 0$ for all $i \notin \hat{A}^t$, and the second inequality uses the fact that $\phi_i^t(\hat{A}^t \cap \mathcal{N}^+) \geq \phi_i^t(\hat{A}^t)$ for all $i \in \hat{A}^t \cap \mathcal{N}^+$. Because $\hat{A}^t \cap \mathcal{N}^+ \in \mathcal{F}$, the chain of inequalities above contradicts the fact that $\hat{A}^t$ is an optimal solution to problem (5). □

3.2. An Approximate Policy Using Marginal Values

We consider the greedy policy with respect to the value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$. If the system is in state q at time period t, then this policy offers the assortment $\hat{S}^t(q)$ given by

$$\hat{S}^t(q) = \arg\max_{S\in\mathcal{F}}\left\{\sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0}) \cdot \mathbb{E}\Big\{\hat{J}^{t+1}(\mathbf{X}(q)) - \hat{J}^{t+1}(\mathbf{X}(q) - \mathbf{e}_{i,0} + \mathbf{e}_{i,1})\Big\}\Big]\right\}$$
$$= \arg\max_{S\in\mathcal{F}} \sum_{i=1}^{n} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0}) \cdot \Big(\hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1}\Big)\Big], \tag{7}$$

where the second equality uses the definition of the value function approximations in (4). The next theorem is the main result of this section, giving a performance guarantee for this policy.

Theorem 3.2 (Performance of the Greedy Policy). *The total expected revenue of the greedy policy with respect to the value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$ is at least 50% of the optimal total expected revenue; that is, this policy is a half-approximation.*

The proof of this theorem makes use of the next lemma. Because we do not have any products in use at the beginning of the selling horizon, the initial state is $\sum_{i\in\mathcal{N}} C_i \mathbf{e}_{i,0}$. The next lemma relates the approximation $\hat{J}^1(\sum_{i\in\mathcal{N}} C_i \mathbf{e}_{i,0})$ to the optimal total expected revenue $J^1(\sum_{i\in\mathcal{N}} C_i \mathbf{e}_{i,0})$.

Lemma 3.3 (Expected Revenue Upper Bound). We have $J^1(\sum_{i\in\mathcal{N}} \cdot C_i \mathbf{e}_{i,0}) \le 2\hat{J}^1(\sum_{i\in\mathcal{N}} C_i \mathbf{e}_{i,0})$.

Proof. By Adelman (2007), we can obtain an upper bound on the optimal total expected revenue by using the objective value provided by any feasible solution to the linear program

$$\min \tilde{J}^1\left(\sum_{i\in\mathcal{N}} C_i \mathbf{e}_{i,0}\right)$$
$$\text{s.t. } \tilde{J}^t(q) \ge \sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{\tilde{J}^{t+1}(\mathbf{X}(q))\Big\} + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0}) \cdot \mathbb{E}\Big\{\tilde{J}^{t+1}(\mathbf{X}(q)) - \tilde{J}^{t+1}(\mathbf{X}(q) - \mathbf{e}_{i,0} + \mathbf{e}_{i,1})\Big\}\Big]$$
$$\forall q \in \mathcal{Q}, S \in \mathcal{F}, t \in \mathcal{T},$$

where the decision variables are $\{\tilde{J}^t(q) : q \in \mathcal{Q}, t \in \mathcal{T}\}$ and we follow the convention that $\tilde{J}^{T+1} = 0$. Define the constant $\hat{\beta}^t = \sum_{i\in\mathcal{N}} \hat{v}_{i,0}^t C_i$. We proceed to show that $\{\hat{\beta}^t + \hat{J}^t(q) : q \in \mathcal{Q}, t \in \mathcal{T}\}$ with $\hat{J}^t(q)$ as in (4)–(6) is a feasible solution to the linear program above. (Without the constant $\hat{\beta}^t$, the solution $\{\hat{J}^t(q) : q \in \mathcal{Q}, t \in \mathcal{T}\}$ is not necessarily feasible to the linear program.) As all units are available at the beginning of the selling horizon, a unit will never be in use for more than T time periods. Thus, we can assume that $\mathcal{Q}$ is a finite set, so the numbers of decision variables and constraints above are finite. By the definitions of $\hat{J}^{t+1}(q)$ in (4) and $\mathbf{X}(q)$ in (1), we get

$$\hat{\beta}^{t+1} + \mathbb{E}\Big\{\hat{J}^{t+1}(\mathbf{X}(q))\Big\}$$
$$= \hat{\beta}^{t+1} + \sum_{i\in\mathcal{N}}\left\{\hat{v}_{i,0}^{t+1}\Big[q_{i,0} + \sum_{\ell=1}^{\infty}\rho_{i,\ell}\, q_{i,\ell}\Big] + \sum_{\ell=1}^{\infty}\hat{v}_{i,\ell+1}^{t+1} \cdot \big[q_{i,\ell} - \rho_{i,\ell}\, q_{i,\ell}\big]\right\}$$
$$= \hat{\beta}^{t+1} + \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}_{i,0}^{t+1} + \sum_{\ell=1}^{\infty} q_{i,\ell}\Big[\rho_{i,\ell}\,\hat{v}_{i,0}^{t+1} + (1-\rho_{i,\ell}) \cdot \hat{v}_{i,\ell+1}^{t+1}\Big]\right\}.$$

Similarly, $\mathbb{E}\{\hat{J}^{t+1}(\mathbf{X}(q)) - \hat{J}^{t+1}(\mathbf{X}(q) - \mathbf{e}_{i,0} + \mathbf{e}_{i,1})\} = \hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1}$. So, if we evaluate the right side of the constraint in the linear program above at $\{\hat{\beta}^t + \hat{J}^t(q) : q \in \mathcal{Q},\, t \in \mathcal{T}\}$, then we obtain

$$\sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \hat{\beta}^{t+1} + \mathbb{E}\Big\{\hat{J}^{t+1}(\mathbf{X}(q))\Big\} + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0})\,\mathbb{E}\Big\{\hat{\beta}^{t+1} + \hat{J}^{t+1}(\mathbf{X}(q)) - \hat{\beta}^{t+1} - \hat{J}^{t+1}(\mathbf{X}(q) - \mathbf{e}_{i,0} + \mathbf{e}_{i,1})\Big\}\Big]$$
$$= \sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \hat{\beta}^{t+1} + \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}_{i,0}^{t+1} + \sum_{\ell=1}^{\infty} q_{i,\ell}\Big[\rho_{i,\ell}\,\hat{v}_{i,0}^{t+1} + (1-\rho_{i,\ell})\,\hat{v}_{i,\ell+1}^{t+1}\Big]\right\} + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0})\Big(\hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1}\Big)\Big]$$
$$= \sum_{i\in\mathcal{N}} \hat{v}_{i,0}^{t+1} C_i + \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}_{i,0}^{t+1} + \sum_{\ell=1}^{\infty} q_{i,\ell}\hat{v}_{i,\ell}^{t}\right\} + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi_i^t(S)\Big[r_i^t + \pi_i^t - (1-\rho_{i,0})\Big(\hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1}\Big)\Big],$$

where the second equality holds because we have $\hat{v}_{i,\ell}^t = \pi_i^t + \rho_{i,\ell}\,\hat{v}_{i,0}^{t+1} + (1-\rho_{i,\ell})\,\hat{v}_{i,\ell+1}^{t+1}$ by (6) and $\hat{\beta}^{t+1} = \sum_{i\in\mathcal{N}} \hat{v}_{i,0}^{t+1} C_i$. By a simple lemma, given as Lemma B.1 in Online Appendix B, if we let $\Delta_i^t = r_i^t + \pi_i^t - (1-\rho_{i,0})\,(\hat{v}_{i,0}^{t+1} - \hat{v}_{i,1}^{t+1})$, then $\sum_{i\in\mathcal{N}} \phi_i^t(\hat{A}^t)\,\Delta_i^t \ge \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}} \cdot \phi_i^t(S)\,\Delta_i^t$ for all $S \in \mathcal{F}$. Note that this inequality does not follow from the definition of $\hat{A}^t$ because although we have $\sum_{i\in\mathcal{N}} \phi_i^t(\hat{A}^t)\,\Delta_i^t \ge \sum_{i\in\mathcal{N}} \phi_i^t(S)\,\Delta_i^t$ by (5), we may have $\Delta_i^t < \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\Delta_i^t$ when $\Delta_i^t < 0$. Thus, using the chain of equalities above, we upper bound the right side of the constraint in the linear program as

$$\begin{aligned}
&\sum_{i\in\mathcal{N}} \hat{v}^{t+1}_{i,0}\, C_i + \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\,\hat{v}^{t}_{i,\ell}\right\}\\
&\quad + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi^t_i(S)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]\\
&\le \sum_{i\in\mathcal{N}} \hat{v}^{t+1}_{i,0}\, C_i + \sum_{i\in\mathcal{N}}\sum_{\ell=0}^{\infty} q_{i,\ell}\,\hat{v}^{t}_{i,\ell}\\
&\quad + \sum_{i\in\mathcal{N}} \phi^t_i\left(\hat{A}^t\right)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]\\
&= \sum_{i\in\mathcal{N}} \hat{v}^{t+1}_{i,0}\, C_i + \sum_{i\in\mathcal{N}}\sum_{\ell=0}^{\infty} q_{i,\ell}\,\hat{v}^{t}_{i,\ell} + \sum_{i\in\mathcal{N}} C_i\left(\hat{v}^{t}_{i,0} - \hat{v}^{t+1}_{i,0}\right)\\
&= \sum_{i\in\mathcal{N}} \hat{v}^{t}_{i,0}\, C_i + \sum_{i\in\mathcal{N}}\sum_{\ell=0}^{\infty} q_{i,\ell}\,\hat{v}^{t}_{i,\ell} = \hat{\beta}^t + \hat{J}^t(\boldsymbol{q}),
\end{aligned}$$

where the first inequality holds as $\hat{v}^t_{i,0} \ge \hat{v}^{t+1}_{i,0}$ by Lemma 3.1, the first equality follows from (6), and the last equality is by the definition of $\hat{\beta}^t$. By the chain of inequalities above, for any $\boldsymbol{q}\in\mathcal{Q}$, $S\in\mathcal{F}$, and $t\in\mathcal{T}$, if we evaluate the right side of the constraint at $\{\hat{\beta}^t + \hat{J}^t(\boldsymbol{q}) : \boldsymbol{q}\in\mathcal{Q}, t\in\mathcal{T}\}$, then the right side of the constraint is upper bounded by $\hat{\beta}^t + \hat{J}^t(\boldsymbol{q})$. So, the solution $\{\hat{\beta}^t + \hat{J}^t(\boldsymbol{q}) : \boldsymbol{q}\in\mathcal{Q}, t\in\mathcal{T}\}$ is feasible to the linear program, which implies that the objective value of the linear program evaluated at this solution is an upper bound on the optimal total expected revenue. The objective value of the linear program evaluated at the solution $\{\hat{\beta}^t + \hat{J}^t(\boldsymbol{q}) : \boldsymbol{q}\in\mathcal{Q}, t\in\mathcal{T}\}$ is $\hat{\beta}^1 + \hat{J}^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0}) = \hat{\beta}^1 + \sum_{i\in\mathcal{N}} \hat{v}^1_{i,0}\, C_i = 2\sum_{i\in\mathcal{N}} \hat{v}^1_{i,0}\, C_i = 2\hat{J}^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0})$. Thus, $2\hat{J}^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0})$ is an upper bound on the optimal total expected revenue. □

The greedy policy with respect to the value function approximations $\{\hat{J}^t : t\in\mathcal{T}\}$ offers the assortment $\hat{S}^t(\boldsymbol{q})$ in (7) when the system is in state $\boldsymbol{q}$ at time period t. Let $U^t(\boldsymbol{q})$ denote the total expected revenue under this greedy policy over the time periods $t,\dots,T$, given that we are in state $\boldsymbol{q}$ at time period t. We can compute $\{U^t : t\in\mathcal{T}\}$ by using the recursion

$$\begin{aligned}
U^t(\boldsymbol{q}) &= \sum_{i\in\mathcal{N}} \pi^t_i \sum_{\ell=1}^{\infty} q_{i,\ell}\\
&\quad + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi^t_i\left(\hat{S}^t(\boldsymbol{q})\right)\Big(r^t_i + \pi^t_i + \mathbb{E}\Big\{\mathsf{Z}(\rho_{i,0})\,U^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\\
&\quad + (1-\mathsf{Z}(\rho_{i,0}))\,U^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big)\\
&\quad + \Big(1 - \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi^t_i\left(\hat{S}^t(\boldsymbol{q})\right)\Big)\mathbb{E}\Big\{U^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\},
\end{aligned}$$

with the boundary condition that $U^{T+1} = 0$. In the recursion above, we use the same line of reasoning that we used for the dynamic-programming formulation in (2), but the decision is fixed as $\hat{S}^t(\boldsymbol{q})$. An observation that will shortly be useful is that U^{t+1} appears with a positive coefficient on the right side above. Therefore, if we replace U^{t+1} with a function H^{t+1} that satisfies $U^{t+1}(\boldsymbol{q}) \ge H^{t+1}(\boldsymbol{q})$, then the right side of the expression above becomes smaller. By using the same sequence of manipulations that we used to obtain the dynamic program in (3), we can write the above recursion equivalently as

$$\begin{aligned}
U^t(\boldsymbol{q}) &= \sum_{i\in\mathcal{N}} \pi^t_i \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{U^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\\
&\quad + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi^t_i(\hat{S}^t(\boldsymbol{q}))\Big(r^t_i + \pi^t_i - (1-\rho_{i,0})\\
&\quad \cdot \mathbb{E}\Big\{U^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - U^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big). \qquad (8)
\end{aligned}$$

The coefficients of U^{t+1} are not necessarily all positive on the right side above, but the last two recursions are equivalent. So, if we replace U^{t+1} on the right side above with a function H^{t+1} that satisfies $U^{t+1}(\boldsymbol{q}) \ge H^{t+1}(\boldsymbol{q})$, then the right side of the expression above still gets smaller.

Here is the proof of Theorem 3.2.

Proof of Theorem 3.2. We will use induction over the time periods to show that $U^t(\boldsymbol{q}) \ge \hat{J}^t(\boldsymbol{q})$ for all $\boldsymbol{q}\in\mathcal{Q}$ and $t\in\mathcal{T}$, where $\hat{J}^t(\boldsymbol{q})$ is as in (4). By definition, we have $\hat{v}^{T+1}_{i,\ell} = 0$ for all $i\in\mathcal{N}$, $\ell = 0,1,\dots,$ so that $\hat{J}^{T+1} = 0$. Furthermore, we have $U^{T+1} = 0$. Thus, the result holds at time period $T+1$. Assuming that $U^{t+1}(\boldsymbol{q}) \ge \hat{J}^{t+1}(\boldsymbol{q})$ for all $\boldsymbol{q}\in\mathcal{Q}$, we proceed to show that $U^t(\boldsymbol{q}) \ge \hat{J}^t(\boldsymbol{q})$ for all $\boldsymbol{q}\in\mathcal{Q}$. Using the same argument in the proof of Lemma 3.3, we have

$$\begin{aligned}
&\mathbb{E}\Big\{\hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\\
&\quad = \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\left[\rho_{i,\ell}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}\right]\right\}.
\end{aligned}$$

Similarly, we have $\mathbb{E}\{\hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - \hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\} = \hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}$. Thus, by the inductive hypothesis and the recursion defining $U^t(\boldsymbol{q})$ in (8), we obtain

$$\begin{aligned}
U^t(\boldsymbol{q}) &\ge \sum_{i\in\mathcal{N}} \pi^t_i \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{\hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q}))\Big\}\\
&\quad + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi^t_i\left(\hat{S}^t(\boldsymbol{q})\right)\Big(r^t_i + \pi^t_i - (1-\rho_{i,0})\\
&\quad \cdot \mathbb{E}\Big\{\hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q})) - \hat{J}^{t+1}(\boldsymbol{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big)\\
&= \sum_{i\in\mathcal{N}} \pi^t_i \sum_{\ell=1}^{\infty} q_{i,\ell} + \sum_{i\in\mathcal{N}}\Big\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\Big[\rho_{i,\ell}\,\hat{v}^{t+1}_{i,0}\\
&\quad + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}\Big]\Big\}\\
&\quad + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\phi^t_i\left(\hat{S}^t(\boldsymbol{q})\right)\Big[r^t_i + \pi^t_i\\
&\quad - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]
\end{aligned}$$

$$= \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\left[\pi^t_i + \rho_{i,\ell}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}\right]\right\} + \max_{S\in\mathcal{F}}\sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi^t_i(S)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big],$$

where the last equality follows from the fact that $\hat{S}^t(\boldsymbol{q})$ is, by (7), an optimal solution to the maximization problem on the right side above. Noting (6), for all $\ell \ge 1$, we have $\hat{v}^t_{i,\ell} = \pi^t_i + \rho_{i,\ell}\,\hat{v}^{t+1}_{i,0} + (1-\rho_{i,\ell})\,\hat{v}^{t+1}_{i,\ell+1}$. In this case, the expression on the right side of the chain of inequalities above can equivalently be written as

$$\sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\,\hat{v}^t_{i,\ell}\right\} + \max_{S\in\mathcal{F}}\sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\,\phi^t_i(S) \cdot\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]$$
$$\ge \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\,\hat{v}^t_{i,\ell}\right\} + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}} \cdot \phi^t_i\left(\hat{A}^t\right)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]$$
$$\ge \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\,\hat{v}^t_{i,\ell}\right\} + \sum_{i\in\mathcal{N}} \frac{q_{i,0}}{C_i} \cdot \phi^t_i\left(\hat{A}^t\right)\Big[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\Big]$$
$$= \sum_{i\in\mathcal{N}}\left\{q_{i,0}\,\hat{v}^{t+1}_{i,0} + \sum_{\ell=1}^{\infty} q_{i,\ell}\,\hat{v}^t_{i,\ell}\right\} + \sum_{i\in\mathcal{N}} q_{i,0}\left(\hat{v}^t_{i,0} - \hat{v}^{t+1}_{i,0}\right)$$
$$= \sum_{i\in\mathcal{N}}\sum_{\ell=0}^{\infty} q_{i,\ell}\,\hat{v}^t_{i,\ell} = \hat{J}^t(\boldsymbol{q}).$$

In the chain of inequalities above, to see that the second inequality holds, by the discussion in the proof of Lemma 3.1, we have $\phi^t_i(\hat{A}^t)\,[r^t_i + \pi^t_i - (1-\rho_{i,0})(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1})] \ge 0$ for all $i \in \mathcal{N}$. Also, by the definition of $\mathcal{Q}$, we have $q_{i,0} \le C_i$ for any $\boldsymbol{q} \in \mathcal{Q}$, which implies that $\mathbb{1}_{\{q_{i,0}\ge 1\}} \ge \frac{q_{i,0}}{C_i}$. The first equality follows from (6). The chain of inequalities above completes the induction argument so that we have $U^t(\boldsymbol{q}) \ge \hat{J}^t(\boldsymbol{q})$ for all $\boldsymbol{q} \in \mathcal{Q}$ and $t \in \mathcal{T}$. Because the initial state of the system is $\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0}$, the total expected revenue collected by the greedy policy is $U^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0})$. So, using the last inequality with $t = 1$ and $\boldsymbol{q} = \sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0}$, we get $U^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0}) \ge \hat{J}^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0}) \ge \frac{1}{2} J^1(\sum_{i\in\mathcal{N}} C_i\,\boldsymbol{e}_{i,0})$, where the second inequality follows by Lemma 3.3. □

We note that simple myopic approaches that ignore the future customer arrivals can perform arbitrarily poorly, as we demonstrate in Online Appendix C. In contrast, by Theorem 3.2, the greedy policy with respect to the value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$ is guaranteed to obtain at least 50% of the optimal total expected revenue. In our computational experiments, we demonstrate that the practical performance of this greedy policy can be substantially better than this theoretical performance guarantee. Despite having a performance guarantee, the greedy policy with respect to the value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$ has a somewhat undesirable feature. Consider two states $\boldsymbol{q} \in \mathcal{Q}$ and $\boldsymbol{q}' \in \mathcal{Q}$ such that $\{i \in \mathcal{N} : q_{i,0} \ge 1\} = \{i \in \mathcal{N} : q'_{i,0} \ge 1\}$. In other words, the set of products for which we have on-hand inventory is the same in the two states. In this case, by (7), we have $\hat{S}^t(\boldsymbol{q}) = \hat{S}^t(\boldsymbol{q}')$. Therefore, the decisions of the greedy policy depend on the set of products for which we have on-hand inventory, but not on the level of inventory for these products. The greedy policy does not differentiate between having too much or too little inventory of a product, as long as we have on-hand inventory for this product. In the next section, we develop a more sophisticated policy that explicitly takes the inventory levels into consideration, while still maintaining the performance guarantee of the greedy policy. Our computational experiments demonstrate that the latter policy can perform noticeably better than the greedy policy.

4. Improving the Policy Performance Through Rollout

To develop a policy that explicitly takes the inventory levels of the products into consideration, we build on a static policy that offers a fixed assortment at each time period. With the assortment $\hat{A}^t$ defined in (5), the static policy always offers the assortment $\hat{A}^t$ at time period t. Using an analysis similar to the one for the greedy policy with respect to the linear value function approximations discussed in the previous section, we show that the static policy obtains at least 50% of the optimal total expected revenue. Furthermore, the value functions associated with the static policy are separable by the products. We perform rollout on the static policy to obtain a policy that takes the inventory levels of the products into consideration, while still maintaining the performance guarantee of the static policy. Exploiting the fact that the value functions associated with the static policy are separable by the products, we show that we can efficiently perform rollout on the static policy when the usage durations follow a negative binomial distribution or when the customers purchase the products outright without returning them at all.

4.1. Properties of the Static Policy

We consider a static policy that always offers the assortment $\hat{A}^t$ at time period t regardless of the product availabilities, where $\hat{A}^t$ is defined in (5). If a customer chooses a product that is not available, then she leaves the system. By the next lemma, the static policy obtains at least 50% of the optimal total expected revenue. The proof is similar to the analysis of the greedy policy with respect to the linear value function approximations. The details are in Online Appendix D.

Lemma 4.1 (Performance of the Static Policy). *The total expected revenue of the static policy that offers assortment $\hat{A}^t$ at time period t is at least 50% of the optimal total expected revenue.*

Let $V^t(\boldsymbol{q})$ denote the total expected revenue under the static policy over the time periods $t, \ldots, T$, given that we are in state $\boldsymbol{q}$ at time period t. Similar to the dynamic program in (3), we can compute $\{V^t : t \in \mathcal{T}\}$ by using the recursion

$$\begin{aligned} V^t(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{V^{t+1}(\mathbf{X}(\boldsymbol{q}))\Big\} \\ + \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\, \phi_i^t\big(\hat{A}^t\big)\Big(r_i^t + \pi_i^t - (1-\rho_{i,0}) \\ \cdot \mathbb{E}\Big\{V^{t+1}(\mathbf{X}(\boldsymbol{q})) - V^{t+1}(\mathbf{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big), \end{aligned}$$

with the boundary condition that $V^{T+1} = 0$. The following lemma shows that $V^t(\boldsymbol{q})$ decomposes by products. The proof is in Online Appendix E. To facilitate our exposition, let $\boldsymbol{e}_\ell$ be the standard unit vector with one in the ℓ-th coordinate. Let $\boldsymbol{q}_i = (q_{i,\ell} : \ell = 0, 1, \ldots)$ denote the numbers of units of product i that have been in use for different numbers of time periods. By (1), the state of the units of product i at the next time period depends on the state of the units of product i at the current time period, but not on other products. Thus, $X_{i,\ell}(\boldsymbol{q})$ is a function of $\boldsymbol{q}_i$ only, which implies that we can write $X_{i,\ell}(\boldsymbol{q})$ as $X_{i,\ell}(\boldsymbol{q}_i)$, so we can define the vector $\mathbf{X}_i(\boldsymbol{q}_i) = (X_{i,\ell}(\boldsymbol{q}_i) : \ell = 0, 1, \ldots)$.

Lemma 4.2 (Decomposability by Products). *For each $t \in \mathcal{T}$ and $\boldsymbol{q} \in \mathcal{Q}$, we have $V^t(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} V_i^t(\boldsymbol{q}_i)$, where for each $i \in \mathcal{N}$, $\{V_i^t : t \in \mathcal{T}\}$ is computed by using the recursion*

$$\begin{aligned} V_i^t(\boldsymbol{q}_i) = \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell} + \mathbb{E}\Big\{V_i^{t+1}(\mathbf{X}_i(\boldsymbol{q}_i))\Big\} \\ + \mathbb{1}_{\{q_{i,0}\ge 1\}}\, \phi_i^t\big(\hat{A}^t\big)\Big(r_i^t + \pi_i^t - (1-\rho_{i,0}) \\ \cdot \mathbb{E}\Big\{V_i^{t+1}(\mathbf{X}_i(\boldsymbol{q}_i)) - V_i^{t+1}(\mathbf{X}_i(\boldsymbol{q}_i) - \boldsymbol{e}_0 + \boldsymbol{e}_1)\Big\}\Big), \quad (9) \end{aligned}$$

with the boundary condition that $V_i^{T+1} = 0$.

4.2. Rollout Policy Based on the Static Policy

We perform rollout on the static policy to obtain a policy that takes the inventory levels of the products into consideration. To perform rollout on the static policy, given that we are in a particular state at the current time period, we choose the decision that maximizes the immediate expected revenue at the current time period plus the expected revenue from the static policy starting from the state at the next time period. We refer to the policy obtained by performing rollout on the static policy as the rollout policy. The rollout policy ultimately corresponds to using $V^t(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} V_i^t(\boldsymbol{q}_i)$ as a separable nonlinear approximation to $J^t(\boldsymbol{q})$. Let $S_{\text{rollout}}^t(\boldsymbol{q})$ be the assortment offered by the rollout policy given that we are in state $\boldsymbol{q}$ at time period t. As $V^{t+1}(\boldsymbol{q})$ is the total expected revenue obtained by the static policy starting in state $\boldsymbol{q}$ at time period $t+1$, $S_{\text{rollout}}^t(\boldsymbol{q})$ is given by

$$\begin{aligned} S_{\text{rollout}}^t(\boldsymbol{q}) \\ = \arg\max_{S\in\mathcal{F}} \Big\{ \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\, \phi_i^t(S)\Big(r_i^t + \pi_i^t + \mathbb{E}\Big\{\mathsf{Z}(\rho_{i,0}) \\ \cdot V^{t+1}(\mathbf{X}(\boldsymbol{q})) + (1-\mathsf{Z}(\rho_{i,0}))\, V^{t+1}(\mathbf{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big) \\ + \Big(1 - \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}} \phi_i^t(S)\Big)\, \mathbb{E}\Big\{V^{t+1}(\mathbf{X}(\boldsymbol{q}))\Big\}\Big\} \\ = \arg\max_{S\in\mathcal{F}} \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}}\, \phi_i^t(S)\, \Big(r_i^t + \pi_i^t - (1-\rho_{i,0}) \\ \cdot \mathbb{E}\Big\{V^{t+1}(\mathbf{X}(\boldsymbol{q})) - V^{t+1}(\mathbf{X}(\boldsymbol{q}) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\Big\}\Big) \\ = \arg\max_{S\in\mathcal{F}} \sum_{i\in\mathcal{N}} \mathbb{1}_{\{q_{i,0}\ge 1\}} \phi_i^t(S)\, \Big(r_i^t + \pi_i^t - (1-\rho_{i,0}) \\ \cdot \mathbb{E}\Big\{V_i^{t+1}(\mathbf{X}_i(\boldsymbol{q}_i)) - V_i^{t+1}(\mathbf{X}_i(\boldsymbol{q}_i) - \boldsymbol{e}_0 + \boldsymbol{e}_1)\Big\}\Big). \end{aligned}$$

In the first equality above, we follow the same argument that we used to construct the dynamic program in (2), in which we find an assortment that maximizes the immediate expected revenue and the expected value function at the next time period under the optimal policy, but above, we use the value function of the static policy at the next time period. We arrive at the second equality by the same reasoning that we used to obtain the dynamic program in (3) from the dynamic program in (2). The third equality follows from the fact that the value functions of the static policy decompose by the products, as shown in Lemma 4.2.

It is a well-known result that the policy obtained by performing rollout on a base policy always performs at least as well as the base policy itself; see section 6.1.3 in Bertsekas and Tsitsiklis (1996). Therefore, the total expected revenue obtained by our rollout policy is at least as large as the total expected revenue obtained by the static policy. So, by Lemma 4.1, the rollout policy obtains at least 50% of the optimal total expected revenue as well. In many applications, a policy based on rollout tends to offer a dramatic improvement over the base policy. The key question is whether the rollout assortment $S_{\text{rollout}}^t(\boldsymbol{q})$ can be computed efficiently. Lemma 4.2 shows that the value function of the static policy is separable by the products, indicating that computing the value functions of the static policy through the recursion in (9) is more manageable than computing the value functions of the optimal policy through the dynamic program in (3). As discussed earlier, without loss of generality, we can assume that the vector $\boldsymbol{q}_i = (q_{i,\ell} : \ell = 0, 1, \ldots)$ is finite-dimensional, because we start with no units in use so that we always have $q_{i,\ell} = 0$ for all $\ell \geq T$. However, the state variable $\boldsymbol{q}_i = (q_{i,\ell} : \ell = 0, 1, \ldots)$ in the recursion in (9) is still a high-dimensional vector. In particular, the state space in this recursion is given by $\mathcal{Q}_i = \{(q_{i,\ell} \in \mathbb{Z}_+ : \ell = 0, 1, \ldots) \mid \sum_{\ell=0}^{\infty} q_{i,\ell} = C_i\}$ and computing the value function $V_i^t(\boldsymbol{q}_i)$ of the static policy for all $\boldsymbol{q}_i \in \mathcal{Q}_i$ is difficult.

In the remainder of this section, we consider two cases. First, if the usage duration follows a negative binomial distribution, then the value functions of the static policy can be computed efficiently. Second, if the customers purchase the products outright and never return them, then the value functions of the static policy can be computed efficiently as well. Once we compute the value functions $\{V^t : t \in \mathcal{T}\}$ of the static policy efficiently, we can solve the maximization problem above that defines $S^t_{\text{rollout}}(\boldsymbol{q})$ to find the assortment offered by the rollout policy. Note that the maximization problem that we solve to obtain the assortment $S^t_{\text{rollout}}(\boldsymbol{q})$ has the same structure as the maximization problem on the right side of the dynamic program in (3). Thus, once we compute the value functions $\{V^t : t \in \mathcal{T}\}$ of the static policy, as discussed at the end of Section 2, there are numerous choice models that render this maximization problem tractable. Lastly, we emphasize that, even if we cannot compute the value functions $\{V^t : t \in \mathcal{T}\}$ of the static policy, we can use simulation to estimate the expected revenue of the static policy, which still allows performing rollout on the static policy. Section 6.1.3 in Bertsekas and Tsitsiklis (1996) discusses using simulation to perform rollout. Naturally, the computational requirements of performing rollout inflate when we use simulation to estimate the total expected revenue of the static policy. Next, we discuss how to perform rollout efficiently when the usage durations have a negative binomial distribution.

4.3. Negative Binomial Usage Duration

In this section, we assume that for each product $i \in \mathcal{N}$, the usage duration is given as $\text{Duration}_i = 1 + \text{NegBin}(s_i, \eta_i)$, where $\text{NegBin}(s_i, \eta_i)$ denotes a negative binomial random variable with parameters $s_i \in \mathbb{Z}_{++}$ and $\eta_i \in [0, 1]$ taking values over $\{0, 1, \ldots\}$. A negative binomial random variable with parameters (s_i, η_i) corresponds to the sum of s_i independent geometric random variables, each with parameter η_i. Thus, a negative binomial random variable with parameters $(1, \eta_i)$ is equivalent to a geometric random variable with parameter η_i. As s_i increases, the probability mass function of a negative binomial random variable with parameters (s_i, η_i) becomes more symmetric. Even with $s_i = 3$, the probability mass function is rather symmetric. Therefore, a negative binomial random variable is quite flexible for modeling usage durations.

Noting that a negative binomial random variable with parameters (s_i, η_i) corresponds to the sum of s_i geometric random variables, we provide the following interpretation for our use of a negative binomial random variable for modeling the usage durations. At each time period, a customer is satisfied with product i with probability η_i. As soon as a customer is dissatisfied with the product for s_i times, she returns the product, ending her rental duration. Naturally, we do not advocate this interpretation as a model of how a customer makes a decision for keeping the product, but this interpretation provides us with the vocabulary to explain our model more clearly, as follows. If the usage durations have negative binomial distributions, then our state variable does not need to keep track of the numbers of units of each product i that have been in use for a certain duration of time. It is enough to use a state variable that keeps track of the numbers of customers who are using each product i and have been dissatisfied for a certain number of times. In this case, we can efficiently compute the value functions of the static policy, as long as s_i is relatively small.

We discuss how we can compute the value functions of the static policy by using a recursion similar to the one in (9) when the usage durations are negative binomial random variables.

State and Transition Dynamics. To compute the value functions of the static policy through a recursion similar to the one used in (9), we define

$w_{i,d}$ = number of customers who are using product i and have been dissatisfied for d times.

A customer using product i returns the product once she has been dissatisfied for s_i times, in which case, the product becomes available on-hand. Therefore, the s_i-dimensional vector $(w_{i,0}, \ldots, w_{i,s_i-1})$ captures the state of the customers using product i. The on-hand inventory of product i is given by $C_i - \sum_{d=0}^{s_i-1} w_{i,d}$. Under negative binomial usage durations, we use $\boldsymbol{w}_i = (w_{i,d} : 0 \leq d \leq s_i - 1)$ to denote the state vector for product i at the beginning of a generic time period. With this state representation, if no purchase is made at the current time period, then the new random state $\boldsymbol{F}_i(\boldsymbol{w}_i) = \big(F_{i,d}(\boldsymbol{w}_i) : 0 \leq d \leq s_i - 1\big)$ at the next time period is given by

$$F_{i,d}(\boldsymbol{w}_i) = \begin{cases} \text{Bin}(w_{i,0}, \eta_i) & \text{if} \quad d = 0, \\ \text{Bin}(w_{i,d}, \eta_i) + \big(w_{i,d-1} - \text{Bin}(w_{i,d-1}, \eta_i)\big) & \\ & \text{if} \quad d = 1, 2, \ldots, s_i - 1, \end{cases}$$

where we use the fact that for each d, the number of customers who continue to remain dissatisfied for d

times at the next time period is equal to $\mathsf{Bin}(w_{i,d}, \eta_i)$, because each customer is satisfied with the product with probability η_i, independently of each other. Furthermore, $w_{i,d-1} - \mathsf{Bin}(w_{i,d-1}, \eta_i)$ captures the number of customers who were dissatisfied for $d-1$ times at the beginning of the current time period and they were dissatisfied one more time in the current time period; in that case, these customers are dissatisfied for a total of d times at the next time period. These customers add up to the number of customers dissatisfied d times at the next time period.

Dynamic-Programming Formulation. With this state representation, we can compute the value functions of the static policy for each product i through the following recursion. We use $\boldsymbol{w}_i = (w_{i,0}, \ldots, w_{i,s_i-1})$ to capture the state of product i. Recall that the static policy offers the assortment $\hat{A}^t$ at each time period t. Given that the state of product i at time period t is $\boldsymbol{w}_i$, let $V_i^t(\boldsymbol{w}_i)$ be the total expected revenue from product i under the static policy over the time periods $t, \ldots, T$. Using the vectors $\boldsymbol{e}_0 = (1,0,0,\ldots,0) \in \mathbb{R}^{s_i}$ and $\boldsymbol{e}_1 = (0,1,0,\ldots,0) \in \mathbb{R}^{s_i}$, we can compute $\{V_i^t : t \in \mathcal{T}\}$ by using the recursion

$$\begin{aligned}
&V_i^t(\boldsymbol{w}_i)\\
&= \pi_i^t \sum_{d=0}^{s_i-1} w_{i,d} + \left(1 - \mathbb{1}_{\{\sum_{d=0}^{s_i-1} w_{i,d} < C_i\}} \phi_i^t\left(\hat{A}^t\right)\right) \mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i))\right\}\\
&\quad + \mathbb{1}_{\{\sum_{d=0}^{s_i-1} w_{i,d} < C_i\}} \phi_i^t\left(\hat{A}^t\right) \Big(r_i^t + \pi_i^t + \eta_i \mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i) + \boldsymbol{e}_0)\right\}\\
&\quad + (1-\eta_i)\mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i) + \boldsymbol{e}_1)\right\}\Big)\\
&= \pi_i^t \sum_{d=0}^{s_i-1} w_{i,d} + \mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i))\right\} + \mathbb{1}_{\{\sum_{d=0}^{s_i-1} w_{i,d} < C_i\}} \phi_i^t\left(\hat{A}^t\right)\\
&\quad \cdot \Big(r_i^t + \pi_i^t - \eta_i \mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i)) - V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i) + \boldsymbol{e}_0)\right\}\\
&\quad - (1-\eta_i)\mathbb{E}\left\{V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i)) - V_i^{t+1}(\boldsymbol{F}_i(\boldsymbol{w}_i) + \boldsymbol{e}_1)\right\}\Big),
\end{aligned} \tag{10}$$

with the boundary condition that $V_i^{T+1} = 0$. In the first equality above, for a customer to rent a unit of product i, we need to have product i available on-hand and the customer needs to choose product i. The number of units of product i available on-hand is given by $C_i - \sum_{d=0}^{s_i-1} w_{i,d}$, so the expression

$$1 - \mathbb{1}_{\left\{\sum_{d=0}^{s_i-1} w_{i,d} < C_i\right\}} \cdot \phi_i^t\left(\hat{A}^t\right)$$

captures the probability that a customer does not rent product i when we offer the assortment $\hat{A}^t$. If $\sum_{d=0}^{s_i-1} w_{i,d} < C_i$, then we have product i available on-hand. If the customer chooses product i, then she rents this product. With probability η_i, the customer renting product i at the current time period is satisfied, and she ends up being a customer with no dissatisfactions at the beginning of the next time period. With probability $1-\eta_i$, the customer renting product i at the current time period is dissatisfied, and she becomes a customer who is dissatisfied for one time at the beginning of the next time period. The second equality follows by arranging the terms. If $s_i = 1$, so that the usage durations for product i are geometric random variables, then the state variable $\boldsymbol{w}_i$ becomes the scalar $w_{i,0}$, in which case, the recursion above continues to hold as long as we set $\boldsymbol{e}_0 = 1$ and $\boldsymbol{e}_1 = 0$.

Discussion of the State Variable. We can reach the state variable $\boldsymbol{w}_i = (w_{i,d} : 0 \le d \le s_i - 1)$ that we used in (10) by starting from the state variable $\boldsymbol{q}_i = (q_{i,\ell} : \ell = 0,1,\ldots)$ that we used in (9). Recall that $q_{i,\ell}$ is the number of units of product i that have been in use for exactly ℓ time periods. Because the number of units of product i available on-hand is given by $q_{i,0} = C_i - \sum_{\ell=1}^{\infty} q_{i,\ell}$, we can use the state variable $(q_{i,\ell} : \ell = 1,2,\ldots)$, instead of $(q_{i,\ell} : \ell = 0,1,\ldots)$. Let $y_{i,d,\ell}$ be the number of customers who have been using product i for exactly ℓ time periods and have been dissatisfied for d times. By definition, we have $q_{i,\ell} = \sum_{d=0}^{s_i-1} y_{i,d,\ell}$. So, we can use the state variable $\boldsymbol{y}_i = (y_{i,d,\ell} : \ell = 1,2,\ldots, 0 \le d \le s_i - 1)$ instead of $(q_{i,\ell} : \ell = 1,2,\ldots)$, because given $(y_{i,d,\ell} : \ell = 1, 2,\ldots, 0 \le d \le s_i - 1)$, we can compute $(q_{i,\ell} : \ell = 1,2,\ldots)$ as $q_{i,\ell} = \sum_{d=0}^{s_i-1} y_{i,d,\ell}$. Lastly, under negative binomial usage durations, from the perspective of immediate expected revenues and state transitions, if we know the number of times a customer has been dissatisfied, then we do not need to know how long she has been using the product. Thus, letting $w_{i,d} = \sum_{\ell=1}^{\infty} y_{i,d,\ell}$ be the number of customers who are using product i and have been dissatisfied d times, we can use $\boldsymbol{w}_i = (w_{i,d} : 0 \le s_i - 1)$ as the state variable, which is precisely the state variable in (10).

In the recursion in (10), the state variable is an s_i-dimensional vector $(w_{i,0},\ldots,w_{i,s_i-1})$ such that $\sum_{d=0}^{s_i-1} \cdot w_{i,d} \le C_i$, so the number of states is $O(C_i^{s_i})$. Thus, when s_i is relatively small, we can compute the value functions of the static policy efficiently. For example, in our computational experiments, using transaction data from the city of Seattle, we find that the negative binomial distribution provides a reasonably good model for the duration of time for which the drivers park their vehicles. In our experiments, the fitted value for the parameter s_i was 2.

4.4. Infinite Usage Duration

In this section, we focus on the case in which the usage duration is infinity. This case corresponds to the situation where the customers buy the products outright, never returning them. Infinite usage durations have a number of interesting applications. In the retail setting, customers make purchases among substitutable products, in which case our model dynamically makes product assortment offerings to each individual customer as a function of the remaining product inventories (Topaloglu 2013, Golrezaei et al. 2014). Also, an important class of revenue management problems occurs on a flight network with parallel flights operating between the same origin–destination pair. In this setting, the customers make a purchase among multiple parallel flights on a particular departure date. Our model dynamically adjusts the assortment of flights offered to each individual customer as a function of the remaining flight capacities (Zhang and Cooper 2005, Liu and van Ryzin 2008, Dai et al. 2014). Under infinite usage durations, we proceed to discuss how we can compute the value functions of the static policy by using a recursion similar to the one in (9).

State and Transition Dynamics. Because the products are purchased outright, we assume that $\pi_i^t = 0$ for all $t \in \mathcal{T}$, so that there is no per-period rental fee. Because the products are not returned, we only need to keep track of the on-hand inventory of product i. We let $q_{i,0}$ be the number of units of product i on-hand and use $q_{i,0}$ as the state variable at the beginning of a time period. If the state of the system at the current time period is $q_{i,0}$ and a customer purchases product i, then the state of the system at the next time period is simply $q_{i,0} - 1$.

Dynamic-Programming Formulation. Given that we have $q_{i,0}$ units of product i on-hand, let $V_i^t(q_{i,0})$ be the total expected revenue from product i under the static policy over the time periods $t, \ldots, T$. We can compute $\{V_i^t : t \in \mathcal{T}\}$ by using the recursion

$$\begin{aligned} V_i^t(q_{i,0}) &= \left(1 - \mathbb{1}_{\{q_{i,0}\geq 1\}}\phi_i^t\left(\hat{A}^t\right)\right) V_i^{t+1}(q_{i,0}) \\ &\quad + \mathbb{1}_{\{q_{i,0}\geq 1\}}\, \phi_i^t\left(\hat{A}^t\right) \left(r_i^t + V_i^{t+1}(q_{i,0} - 1)\right) \\ &= V_i^{t+1}(q_{i,0}) + \mathbb{1}_{\{q_{i,0}\geq 1\}}\, \phi_i^t\left(\hat{A}^t\right) \\ &\quad \left(r_i^t - \left\{V_i^{t+1}(q_{i,0}) - V_i^{t+1}(q_{i,0} - 1)\right\}\right), \end{aligned} \tag{11}$$

with the boundary condition that $V_i^{T+1} = 0$. In the first equality above, if we have on-hand units of product i and a customer chooses product i, then we have one fewer on-hand unit at the next time period. The second equality follows by arranging the terms. Because the state variable $q_{i,0}$ in the recursion above is scalar, we can efficiently compute the value functions of the static policy under infinite usage durations. The recursion in (11) is similar to the one in revenue management problems with a single resource; see section 2.6.2 in Talluri and van Ryzin (2005).

Thus, under both negative binomial and infinite usage durations, we can efficiently perform rollout on the static policy; in that case, we obtain a policy that takes the inventory levels of the products into consideration, while still obtaining at least 50% of the optimal total expected revenue. It turns out that we can further strengthen our performance guarantee under infinite usage durations. In particular, we let $C_{\min} = \min_{i\in\mathcal{N}} C_i$ to capture the smallest inventory of a product. Also, we let $R = \max_{i\in\mathcal{N}}\{\frac{\max_{t\in\mathcal{T}} r_i^t}{\min_{t\in\mathcal{T}} r_i^t}\}$ to capture the largest relative deviation in the upfront fee for a product over the selling horizon. In Theorem F.2 in Online Appendix F, using a modified static policy based on a solution of a linear program, we can construct a tailored rollout policy that is guaranteed to obtain at least $\max\{\frac{1}{2}, 1 - \frac{R}{2\sqrt[3]{C_{\min}}}\}$ fraction of the optimal total expected revenue. Therefore, the tailored variant of our rollout approach always provides at least a half-approximate performance guarantee, but it becomes near-optimal as the inventories of the products become large. This performance guarantee is not an asymptotic performance guarantee. It holds for any value of the product inventories and the number of time periods in the selling horizon. For example, if the smallest product inventory is 100 and the upfront fees are stationary so that $C_{\min}=100$ and $R=1$, then the tailored variant of our rollout policy is guaranteed to obtain at least 89% of the optimal total expected revenue, regardless of the other problem parameters. In addition, we consider a standard regime where the inventories of the products and the number of time periods in the selling horizon scale up linearly at the same rate κ (Gallego and van Ryzin 1994). In Theorem F.7 in Online Appendix F, we also show that the tailored variant of our rollout policy obtains at least $1 - \frac{B}{\sqrt{\kappa}}$ fraction of the optimal total expected revenue, where B is a constant that is independent of the scaling rate κ. Thus, the relative optimality gap of the tailored variant of the rollout policy is $O(1 - \frac{1}{\sqrt{\kappa}})$ as the inventories of the products and the number of time periods scale up linearly at the same rate κ. These two performance guarantees do not generalize to arbitrary usage duration distributions.

5. Extensions

We give extensions to the case in which we have multiple customer types, we make pricing decisions instead of assortment offer decisions, and we can solve the assortment optimization problems only approximately. We show that our half-approximate performance guarantee continues to hold when we have multiple customer types and when we make pricing decisions. Furthermore, we show that if we can solve the assortment optimization problems approximately, then our performance guarantees hold with appropriate modifications to reflect the solution accuracy in the assortment problems. Some of these extensions are used in our computational experiments.

5.1. Heterogeneous Customer Types

We have m customer types indexed by $\mathcal{M} = \{1, 2, \ldots, m\}$. At time period $t \in \mathcal{T}$, a customer of type j arrives with probability $p^{t,j}$, where we have $\sum_{j\in\mathcal{M}} p^{t,j} = 1$, so that each time period has exactly one customer arrival. We observe the type of each arriving customer. Each customer type has its own choice model, reward structure, assortment constraints, and usage duration. If we offer the subset S of products, then a customer of type j arriving at time period t chooses product i with probability $\phi_i^{t,j}(S)$. Note that if we do not observe the type of each arriving customer, then we can continue using the model in Section 2, where the choice probability $\phi_i^t(S)$ is obtained by mixing the choice models corresponding to different customer types. If a customer of type j selects product i at time period t, then she pays a one-time upfront fee of $r_i^{t,j}$. Furthermore, if she rents this product during time period t, then she pays a per-period rental fee of $\pi_i^{t,j}$. The usage duration of product i by a customer of type j is given by the random variable $\mathsf{Duration}_i^j$. We let $\rho_{i,\ell}^j$ be the hazard rate of the usage duration of product i for a customer of type j, which is defined by $\rho_{i,\ell}^j = \Pr\{\mathsf{Duration}_i^j = \ell + 1 \mid \mathsf{Duration}_i^j > \ell\}$. Lastly, the assortments offered to customers of different types have different feasibility requirements. We use $\mathcal{F}^j$ to denote the set of feasible assortments that can be offered to customers of type j.

We can extend all of our results to the case with heterogeneous customer types. We will focus on the essentials in this section; the details are included in Online Appendix G. To capture the state of the system, because each customer type has its own reward structure and usage duration, we need to keep track of the number of units that are currently in use by each customer type. We use $q_{i,0}$ to denote the number of units of product i on-hand. For $\ell \geq 1$, we use $q_{i,\ell}^j$ to denote the number of units of product i that have been used for exactly ℓ time periods by a customer of type j. Therefore, we can describe the state of the system by using $\boldsymbol{q} = (q_{i,0}, q_{i,\ell}^j : i \in \mathcal{N}, j \in \mathcal{M}, \ell \geq 1)$. Using $\boldsymbol{q}$ as the state variable, we can give a dynamic-programming formulation of the problem that resembles the one in (2). In this case, we use value function approximations of the form

$$\hat{J}^t(\boldsymbol{q}) = \sum_{i\in\mathcal{N}} \hat{\theta}_i^t\, q_{i,0} + \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{M}} \sum_{\ell=1}^{\infty} \hat{v}_{i,\ell}^{t,j}\, q_{i,\ell}^j,$$

where $\hat{\theta}_i^t$ captures the marginal value of a unit of product i on-hand at time period t and $\hat{v}_{i,\ell}^{t,j}$ captures the marginal value of a unit of product i that has been in use for ℓ periods by a customer of type j at time period t. We propose computing $\hat{\theta}_i^t$ and $\hat{v}_{i,\ell}^{t,j}$ recursively as follows.

- **Initialization:** Set $\hat{\theta}_i^{T+1} = 0$ and $\hat{v}_{i,\ell}^{T+1,j} = 0$ for all $i \in \mathcal{N}, j \in \mathcal{M}, \ell \geq 1$.
- **Recursion:** For $t = T, T-1, \ldots, 1$, we compute $\hat{\theta}_i^t$ and $\hat{v}_{i,\ell}^{t,j}$ by using $\{\hat{\theta}_i^{t+1} : i \in \mathcal{N}\}$ and $\{\hat{v}_{i,\ell}^{t+1,j} : i \in \mathcal{N}, j \in \mathcal{M}, \ell \geq 1\}$ as follows. For each $j \in \mathcal{M}$, let $\hat{A}^{t,j} \in \mathcal{F}^j$ be such that

$$\hat{A}^{t,j} = \arg\max_{S\in\mathcal{F}^j} \sum_{i\in\mathcal{N}} \phi_i^{t,j}(S) \left[r_i^{t,j} + \pi_i^{t,j} - \left(1 - \rho_{i,0}^j\right)\left(\hat{\theta}_i^{t+1} - \hat{v}_{i,1}^{t+1,j}\right) \right].$$

Once $\hat{A}^{t,j}$ is computed for all $j \in \mathcal{M}$, for each $i \in \mathcal{N}$ and $j \in \mathcal{M}$, let

$$\hat{\theta}_i^t = \hat{\theta}_i^{t+1} + \frac{1}{C_i} \sum_{j\in\mathcal{M}} p^{t,j} \phi_i^{t,j}\left(\hat{A}^{t,j}\right) \left[r_i^{t,j} + \pi_i^{t,j} - \left(1 - \rho_{i,0}^j\right)\left(\hat{\theta}_i^{t+1} - \hat{v}_{i,1}^{t+1,j}\right) \right]$$
$$\hat{v}_{i,\ell}^{t,j} = \pi_i^{t,j} + \rho_{i,\ell}^j\, \hat{\theta}_i^{t+1} + \left(1 - \rho_{i,\ell}^j\right) \hat{v}_{i,\ell+1}^{t+1,j} \qquad \forall\, \ell = 1, 2, \ldots. \tag{12}$$

The above discussion completes the specification of the approximate value function $\hat{J}^t$. The computation of the parameters $\{\hat{\theta}_i^t : i \in \mathcal{N}, t \in \mathcal{T}\}$ and $\{\hat{v}_{i,\ell}^{t,j} : i \in \mathcal{N}, j \in \mathcal{M}, \ell \geq 1, t \in \mathcal{T}\}$ is similar to our approach in Section 3.1. Also, the intuition for the specification of the parameters above is similar to the one discussed in Section 3.1. Using an argument similar to the one in the previous two sections, we can show that the greedy policy with respect to the value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$ obtains at least 50% of the optimal total expected revenue. We can also perform rollout on a static policy to obtain a policy that takes the inventory levels of the products into consideration, while ensuring that we still obtain at least 50% of the optimal total expected revenue. We describe both of these results in Online Appendix G.

The use of heterogeneous customer types also allows us to model the case where the usage duration is revealed before offering an assortment. In our problem formulation, we observe the type of a customer before offering an assortment. Also, each customer type can have its own usage duration distribution. Thus, by associating different deterministic usage durations with different customer types, noting that we observe the type of a customer before offering an assortment, we can model the case where the usage duration is revealed before we offer an assortment.

5.2. Price Optimization with Discrete Prices

So far in the paper, we have assumed that the upfront and per-period rental fees for the products are fixed, and we decide on the assortment of products to make available to the customers. It is not difficult to adopt our results to the case in which we decide the upfront and per-period rental fees for the products and the customers choose based on the prices we charge.

In particular, we create multiple copies of each product i, where the different copies correspond to charging different prices for product i. We call each copy of a product a virtual product. Let $\mathcal{H}$ denote the set of possible copies of each product. We write $(i,h) \in \mathcal{N} \times \mathcal{H}$ to denote copy h of product i. Thus, the pairs $\{(i,h) : i \in \mathcal{N}, h \in \mathcal{H}\}$ are the set of all virtual products that we can offer to the customers. Offering virtual product (i,h) means that we offer product i at the price level corresponding to copy h of this product. In this case, the question becomes that of choosing an assortment of virtual products to offer at each time period to maximize the total expected revenue. As we can offer a product at no more than one price level, among all virtual copies of a particular product, we can offer at most one virtual copy. Thus, the set of possible assortments of virtual products that we can offer at each time period is given by $\mathcal{F} = \{S \subseteq \mathcal{N} \times \mathcal{H} : |S \cap (\{i\} \times \mathcal{H})| \le 1 \forall i \in \mathcal{N}\}$. Using $r^t_{i,h}$ to denote the upfront fee at time period t when we charge the price level corresponding to copy h for product i, and $\pi^t_{i,h}$ to denote the per-period fee at time period t when we charge the price level corresponding to copy h of product i, we can follow the same outline in the previous two sections to come up with a policy that obtains at least 50% of the optimal total expected revenue. The only difference is that we treat the virtual products $\mathcal{N} \times \mathcal{H}$ as the products.

5.3. Solving the Assortment Optimization Problem Approximately

The maximization problem in (5) is a combinatorial optimization problem. Under many choice models, we can solve this problem tractably, but it is not possible to solve this problem tractably under every choice model. In this section, we discuss how we can adapt our approach in principle to the case where we have a fully polynomial-time approximation scheme (FPTAS) for problem (5). For any $\epsilon > 0$, the FPTAS returns a $1/(1+\epsilon)$-approximate solution to problem (5), and the running time to do so is polynomial in n and $1/\epsilon$. It turns out that we can leverage the FPTAS to obtain a $1/(2(1+\epsilon))$-approximate policy, and the running time to obtain and execute the approximate policy is polynomial in n, $1/\epsilon$ and T. In particular, assume that we have an FPTAS such that for any $\epsilon > 0$, the FPTAS finds an assortment $\hat{A}^t$ satisfying

$$(1+\epsilon) \sum_{i \in \mathcal{N}} \phi^t_i\left(\hat{A}^t\right)\left[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\right]$$
$$\ge \max_{S \in \mathcal{F}} \sum_{i \in \mathcal{N}} \phi^t_i(S)\left[r^t_i + \pi^t_i - (1-\rho_{i,0})\left(\hat{v}^{t+1}_{i,0} - \hat{v}^{t+1}_{i,1}\right)\right],$$

in running time that is polynomial in n and $1/\epsilon$. In the next theorem, we show how to leverage this FPTAS to find a $1/(2(1+\epsilon))$-approximate policy. The proof is in Online Appendix H.

Theorem 5.1 (Policies Through Approximate Solutions). *Assume that for any $\epsilon > 0$, we can find a $1/(1+\epsilon)$-approximate solution to problem (5) in running time that is polynomial in n and $1/\epsilon$. Then, we can construct value-function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$ such that the greedy policy with respect to these value function approximations is a $1/(2(1+\epsilon))$-approximate policy and the running time to obtain and execute the greedy policy is polynomial in n, $1/\epsilon$, and T.*

A quick inspection of the proof of Theorem 5.1 shows that if the running time to obtain a $1/(1+\epsilon)$-approximate solution to problem (5) is $O(f(n,\frac{1}{\epsilon}))$ for some function f and the running time to compute the probabilities $\{\phi^t_i(S) : i \in \mathcal{N}\}$ for a fixed subset S and time period t is $O(g(n))$ for some function g, then the running time to obtain a $1/(2(1+\epsilon))$-approximate policy is $O(T \times f(n,\frac{4T}{\epsilon}) + T \times g(n) + T^2 n)$. Therefore, if we have an FPTAS for problem (5) so that $f(n,\frac{1}{\epsilon})$ is polynomial in $\frac{1}{\epsilon}$, then $f(n,\frac{4T}{\epsilon})$ is also a polynomial in n, $1/\epsilon$ and T, which corresponds to the case discussed in the theorem above. On the other hand, if we have only a polynomial-time approximation scheme for problem (5) so that $f(n,\frac{1}{\epsilon})$ is polynomial in n but exponential in $\frac{1}{\epsilon}$, then $f(n,\frac{4T}{\epsilon})$ is polynomial in n but exponential in $\frac{1}{\epsilon}$ and T.

6. Computational Experiments

We provide computational experiments to test the performance of our policies. In Section 6.1, we give an approach to obtain an upper bound on the optimal total expected revenue, which is useful for assessing the optimality gaps of our policies. In Sections 6.2 and 6.3, we give our computational results on retail assortment management and pricing parking spaces in the city of Seattle.

6.1. Upper Bound on the Optimal Total Expected Revenue

To compute an upper bound on the optimal total expected revenue, we formulate a linear program, in which the choices of the customers and the transition dynamics take on their expected values. We use the decision variables $(z^t(A) : A \in \mathcal{F}, t \in \mathcal{T})$ and $(q^t_{i,\ell} : i \in \mathcal{N}, \ell \ge 0, t \in \mathcal{T})$, where $z^t(A)$ is the frequency with which we offer assortment A at time period t and $q^t_{i,\ell}$ is the expected number of units of product i that have been in use for exactly ℓ time periods at time period t. To construct the constraints in our linear program, noting the dynamic-programming formulation in (2), if the state of the system at the beginning of time period t is $\boldsymbol{q}^t = (q^t_{i,\ell} : i \in \mathcal{N}, \ell \ge 0)$ and the customer arriving at this time period chooses product i, then the state of the system at the beginning of the next time period is given by the random variable $\mathsf{Z}(\rho_{i,0})\, \boldsymbol{X}(\boldsymbol{q}^t) + (1-\mathsf{Z}(\rho_{i,0}))\,(\boldsymbol{X}(\boldsymbol{q}^t) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})$, where $\mathsf{Z}(\rho)$ is a Bernoulli random variable with parameter ρ. If the customer does not choose any of the products, then the state of the system is $\boldsymbol{X}(\boldsymbol{q}^t)$. Furthermore, if we offer the assortment A at time period t with frequency $z^t(A)$, then the probability that a customer chooses product i is $\sum_{A\in\mathcal{F}} \phi_i^t(A) z^t(A)$. In this case, if the state of the system at the beginning of time period t is $\boldsymbol{q}^t$ and we offer assortment A with frequency $z^t(A)$, then the expected state of the system at the beginning of the next time period is given by $\sum_{i\in\mathcal{N}}\{\sum_{A\in\mathcal{F}} \phi_i^t(A) z^t(A)\}$ $\mathbb{E}\{\mathsf{Z}(\rho_{i,0})\,\boldsymbol{X}(\boldsymbol{q}^t) + (1-\mathsf{Z}(\rho_{i,0}))\,(\boldsymbol{X}(\boldsymbol{q}^t) - \boldsymbol{e}_{i,0} + \boldsymbol{e}_{i,1})\} + \{1 - \sum_{i\in\mathcal{N}}\{\sum_{A\in\mathcal{F}} \phi_i^t(A)\, z^t(A)\}\}\,\mathbb{E}\{\boldsymbol{X}(\boldsymbol{q}^t)\}$. Thus, using the fact that $\mathbb{E}\{\mathsf{Z}(\rho_{i,0})\} = \rho_{i,0}$, by arranging the terms, the expected state at the beginning of the next time period is given by $\mathbb{E}\{\boldsymbol{X}(\boldsymbol{q}^t)\} - \sum_{i\in\mathcal{N}}\{\sum_{A\in\mathcal{F}} \phi_i^t(A)\, z^t(A)\} \times (1-\rho_{i,0})\cdot (\boldsymbol{e}_{i,0} - \boldsymbol{e}_{i,1})$. By (1), $\mathbb{E}\{X_{i,0}(\boldsymbol{q}^t)\} = q_{i,0}^t + \sum_{s=1}^{\infty} \rho_{i,s}\, q_{i,s}^t$, $\mathbb{E}\{X_{i,1}\cdot(\boldsymbol{q}^t)\} = 0$ and $\mathbb{E}\{X_{i,\ell}(\boldsymbol{q}^t)\} = q_{i,\ell-1}^t - \rho_{i,\ell-1}\, q_{i,\ell-1}^t$ for $\ell \ge 2$, which implies that the expected next state $\mathbb{E}\{\boldsymbol{X}(\boldsymbol{q}^t)\}$ in the last expression is linear in the decision variables $\boldsymbol{q}^t = (q_{i,\ell}^t : i\in\mathcal{N}, \ell \ge 0)$. To obtain an upper bound on the optimal total expected revenue in our dynamic assortment problem, we use the linear program

$$
\begin{aligned}
\max\ & \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{N}}\sum_{A\in\mathcal{F}} \left(r_i^t + \pi_i^t\right) \phi_i^t(A)\, z^t(A) + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{N}} \pi_i^t \sum_{\ell=1}^{\infty} q_{i,\ell}^t \\
\text{s.t. } & \boldsymbol{q}^{t+1} = \mathbb{E}\left\{\boldsymbol{X}\left(\boldsymbol{q}^t\right)\right\} - \sum_{i\in\mathcal{N}} \left\{\sum_{A\in\mathcal{F}} \phi_i^t(A)\, z^t(A)\right\} \\
& \qquad (1-\rho_{i,0})\,(\boldsymbol{e}_{i,0} - \boldsymbol{e}_{i,1}) \quad \forall\, t\in\mathcal{T}\setminus\{T\} \\
& \boldsymbol{q}^1 = \sum_{i\in\mathcal{N}} C_i\, \boldsymbol{e}_{i,0} \\
& \sum_{A\in\mathcal{F}} z^t(A) = 1 \quad \forall\, t\in\mathcal{T} \\
& z^t(A) \ge 0 \quad \forall\, A\in\mathcal{F}, t\in\mathcal{T}, q_{i,\ell}^t \ge 0 \quad \forall\, i\in\mathcal{N}, \\
& \qquad\qquad \ell \ge 0, t\in\mathcal{T}.
\end{aligned} \tag{13}
$$

From the discussion right before the above problem, the objective function and the constraints are linear in $(z^t(A) : A\in\mathcal{F}, t\in\mathcal{T})$ and $(q_{i,\ell}^t : i\in\mathcal{N}, \ell\ge 0, t\in\mathcal{T})$. Therefore, the problem above is indeed a linear program. Because $\sum_{A\in\mathcal{F}} \phi_i^t(A)\, z^t(A)$ is the expected number of customers that choose product i at time period t, and $\sum_{\ell=1}^{\infty} q_{i,\ell}^t$ is the expected number of units of product i that are in use at time period t, the objective function computes the total expected revenue over the selling horizon. The first constraint keeps track of the expected numbers of products with different durations of use. The second constraint initializes the state of the system. The third constraint ensures that we offer an assortment at each time period, but this assortment can be empty. By the same argument in Section 2, because the products are all available on-hand at the beginning of the selling horizon, we have $q_{i,\ell}^t = 0$ for all $\ell \ge T+1$ in a feasible solution to the linear program above. Thus, we do not need to define the decision variable $q_{i,\ell}^t$ for $\ell \ge T+1$, which indicates that the numbers of decision variables and constraints are finite. In the next proposition, we show that the optimal objective value of the linear program above is an upper bound on the optimal total expected revenue in our dynamic assortment problem. The proof follows from a standard argument in the revenue management literature. We defer the proof to Online Appendix I.

Proposition 6.1. *Letting Z^* be the optimal objective value of problem (13), we have $Z^* \ge J^1(\sum_{i\in\mathcal{N}} C_i\, \boldsymbol{e}_{i,0})$.*

In problem (13), we have one decision variable $z^t(A)$ for each assortment $A\in\mathcal{F}$. Therefore, the number of decision variables increases exponentially with the number of products. Nevertheless, we can solve problem (13) by using column generation. In particular, noting that $\boldsymbol{q}^{t+1}$ in the first constraint in problem (13) corresponds to the vector $\boldsymbol{q}^{t+1} = (q_{i,\ell}^t : i\in\mathcal{N}, \ell\ge 0)$, we use $\boldsymbol{\alpha} = (\alpha_{i,\ell}^{t+1} : i\in\mathcal{N}, \ell\ge 0, t\in\mathcal{T}\setminus\{T\})$ to denote the dual variables associated with the first constraint. Similarly, noting that $\boldsymbol{q}^1$ in the second constraint in problem (13) corresponds to the vector $\boldsymbol{q}^1 = (q_{i,\ell}^1 : i\in\mathcal{N}, \ell\ge 0)$, we use $\boldsymbol{\delta} = (\delta_{i,\ell} : i\in\mathcal{N}, \ell\ge 0)$ to denote the dual variables associated with the second constraint. Also, we use $\gamma = (\gamma^t : t\in\mathcal{T})$ to denote the dual variables associated with the third constraint. In this case, the constraint associated with the decision variable $z^t(A)$ in the dual of problem (13) is $\sum_{i\in\mathcal{N}} \phi_i^t\cdot(A)(1-\rho_{i,0})(\alpha_{i,0}^{t+1} - \alpha_{i,1}^{t+1}) + \gamma^t \ge \sum_{i\in\mathcal{N}} \phi_i^t(A)\,(r_i^t + \pi_i^t)$. If we solve problem (13) with only a subset of the decision variables $(z^t(A) : A\in\mathcal{F}, t\in\mathcal{T})$ to obtain the dual solution $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\gamma}})$, then we can find which of the decision variables $(z^t(A) : A\in\mathcal{F}, t\in\mathcal{T})$ has the largest reduced cost by solving the problem

$$
\begin{aligned}
& \max_{A\in\mathcal{F}} \left\{ \sum_{i\in\mathcal{N}} \phi_i^t(A)\left(r_i^t + \pi_i^t\right) - \sum_{i\in\mathcal{N}} \phi_i^t(A)\,(1-\rho_{i,0}) \right. \\
& \qquad \left. \left(\hat\alpha_{i,0}^{t+1} - \hat\alpha_{i,1}^{t+1}\right) \right\} \\
& \quad = \max_{A\in\mathcal{F}} \sum_{i\in\mathcal{N}} \phi_i^t(A)\left[r_i^t + \pi_i^t - (1-\rho_{i,0})\left(\hat\alpha_{i,0}^{t+1} - \hat\alpha_{i,1}^{t+1}\right)\right],
\end{aligned}
$$

for all $t\in\mathcal{T}$. In the problem above, we follow the convention that $\alpha_{i,0}^{T+1} = \alpha_{i,1}^{T+1} = 0$ for all $i\in\mathcal{N}$. The problem above is known as the column generation subproblem. The column generation subproblem above has the same structure as the maximization problem in (3). As discussed at the end of Section 2, this problem is tractable under a variety of choice models. Also, if the customers choose according to the multinomial logit model and there are no constraints on the assortments that we can offer, then we can build on the work of Gallego et al. (2015) to give an equivalent formulation for problem (13), whose numbers of decision variables and constraints increase linearly with the number of products. Therefore, we can directly solve the equivalent formulation without resorting to column generation. We discuss the equivalent formulation in Online Appendix J.

We formulate problem (13) under the assumption that there is a single customer type, and we make assortment decisions. We can formulate analogues of problem (13) when we have multiple customer types and we make pricing decisions, which reflect the extensions we provided.

6.2. Dynamic Assortment Management

In our first set of computational experiments, the products are not reusable. We have access to a set of products with limited inventories. Customers arrive over time. Based on the remaining inventories of the products and the number of time periods left in the selling horizon, we offer an assortment to each arriving customer. The customer either purchases a product within the assortment or leaves without making a purchase. The purchased product is not returned, so the usage durations are infinite. Our goal is to find a policy to decide which assortment of products to offer to each customer so that we maximize the total expected revenue over the selling horizon.

Experimental Setup. In our test problems, we have six products indexed by $\mathcal{N} = \{1, \ldots, 6\}$ and six customer types indexed by $\mathcal{M} = \{1, \ldots, 6\}$. In Section 5.1, we discussed how to extend our model to the case with multiple customer types. Recalling that $\pi_i^{t,j}$ is the per-period rental fee that a customer of type j pays for product i at time period t, because the customers purchase the products outright, we set $\pi_i^{t,j} = 0$. The one-time upfront fee $r_i^{t,j}$ that a customer of type j pays for product i at time period t does not depend on the time period or the customer type. Thus, we use r_i to denote the upfront fee for product i.

To determine the upfront fees, we generate r_i from the uniform distribution over $[10, 25]$. After generating the upfront fees for all of the products, we reorder them so that $r_1 \geq r_2 \geq \ldots \geq r_n$. Thus, the first product has the largest upfront fee and the last product has the smallest upfront fee. The customers choose among the products according to the multinomial logit model. A customer of type j associates the preference weight v_i^j with product i and the preference weight v_0^j with the no-purchase option. If we offer the assortment S, then a customer of type j arriving at time period t chooses product $i \in S$ with probability $\phi_i^{t,j}(S) = v_i^j/(v_0^j + \sum_{\ell \in S} v_\ell^j)$. Note that the choice probabilities do not depend on the time period. To come up with the preference weights, we set the consideration set of customer type j as $\mathcal{C}_j = \{1, \ldots, j\}$. If $i \in \mathcal{C}_j$, then we generate v_i^j from the uniform distribution over $[0.9, 1.1]$, whereas if $i \notin \mathcal{C}_j$, then we set $v_i^j = 0$. Thus, a customer is interested in purchasing only the products in her consideration set. Among the products in her consideration set, she is somewhat indifferent. We calibrate the preference weight of the no-purchase option so that if we offer all products, then a customer leaves without a purchase with probability 0.1. Therefore, we calibrate v_0^j to satisfy $v_0^j/(v_0^j + \sum_{\ell \in \mathcal{C}_j} v_\ell^j) = 0.1$. Golrezaei et al. (2014) use a similar multinomial logit model with consideration sets in their computational experiments. Note that a customer of type n has the largest consideration set $\mathcal{C}_n = \{1, \ldots, n\}$, whereas a customer of type 1 has the smallest consideration set $\mathcal{C}_1 = \{1\}$. Therefore, customers of type n are the least choosy, whereas customers of type 1 are the most choosy.

In our test problems, the more choosy customers tend to arrive later in the selling horizon so that we need to carefully protect inventory for them. In particular, we choose equally spaced time periods $\tau^n \leq \tau^{n-1} \leq \ldots \leq \tau^1$ over the selling horizon. The probability $p^{t,j}$ that a customer of type j arrives at time period t is proportional to $e^{-\kappa|t-\tau^j|}$, where κ is a parameter that we vary. That is, we have $p^{t,j} = e^{-\kappa|t-\tau^j|} / \sum_{k \in \mathcal{M}} e^{-\kappa|t-\tau^k|}$. So, the arrival probability for a customer of type j peaks at around time period τ^j. Because $\tau^n \leq \tau^{n-1} \leq \ldots \leq \tau^1$, as $\kappa \to \infty$, we obtain an arrival process where customers of type n arrive first, followed by customers of type $n-1$ and so on. As $\kappa \to 0$, we have $p^{t,j} \to 1/|\mathcal{M}|$, in which case, different customer types arrive with equal probability at each time period. Thus, we control the arrival order for the customer types through the parameter κ. The selling horizon has $T = 300$ time periods. The initial inventory of product i is $C_i = 30/\alpha$, where α is another parameter that we vary to control the inventory scarcity.

Varying the parameters (α, κ) over $\{0.7, 0.8, 0.9, 1.0\} \times \{0, 0.01, 0.03\}$, we obtain 12 test problems in our experimental setup.

Benchmarks. In our computational experiments, we compare the performance of the following seven benchmark strategies.

Greedy Policy (GR). In this benchmark, we use the greedy policy with respect to the linear value function approximations $\{\hat{J}^t : t \in \mathcal{T}\}$, as discussed in Section 3.

Rollout Policy (RO). This benchmark is the policy obtained by applying rollout on the static policy, as discussed in Section 4.

Bid-Prices (BP). We use the classical bid-price policy in this benchmark. We solve the linear program in (13) to estimate the value of a unit of inventory for each product, called its bid-price. We offer the revenue-maximizing set of products at each time period, after adjusting the revenues from the products by their bid-prices; see section 5.2 in Zhang and Adelman (2009).

Offer Sets (OS). We solve the linear program in (13) to obtain an optimal solution $(\hat{z}^t(A) : A \in \mathcal{F}, t \in \mathcal{T})$ and $(\hat{q}^t_{i,\ell} : i \in \mathcal{N}, \ell \geq 0, t \in \mathcal{T})$. Because $\sum_{A \in \mathcal{F}} \hat{z}^t(A) = 1$, letting N^t be the set of products with on-hand inventory at time period t, we sample an assortment S with respect to the probabilities $(\hat{z}^t(A) : A \in \mathcal{F})$ and offer the assortment $S \cap N^t$ at time period t. Offering the assortment $S \cap N^t$ ensures that we only offer products that are currently available.

Decomposition (DC). This benchmark is the classical dynamic-programming decomposition method. The idea is to decompose the dynamic-programming formulation of the problem by the products and to obtain value function approximations by solving a separate dynamic program for each product; see section 6.2 in Liu and van Ryzin (2008). To our knowledge, this benchmark is one of the strongest heuristics in practice, but it does not have a performance guarantee.

Myopic Policy (MY). We can construct myopic policies by ignoring the future customer arrivals altogether. In particular, if we are at time period t with $q^t_{i,0}$ units of product i on-hand, then the assortment that we offer to a customer of type j is given by an optimal solution to the problem $\max_{S \in \mathcal{F}} \sum_{i \in \mathcal{N}} \mathbb{1}_{\{q^t_{i,0} \geq 1\}} \phi^{t,j}_i(S)\, r_i$. We use this benchmark to demonstrate the importance of considering the future customer arrivals when choosing an assortment to offer.

Inventory Balancing (IB). We implement the inventory-balancing policy in Golrezaei et al. (2014). Letting $\Psi : [0,1] \to [0,1]$ be an increasing function with $\Psi(0) = 0$, if we are at time period t with $q^t_{i,0}$ units of product i on-hand, then the assortment that we offer to a customer of type j is given by an optimal solution to $\max_{S \in \mathcal{F}} \sum_{i \in \mathcal{N}} \Psi(q^t_{i,0}/C_i)\, \phi^{t,j}_i(S)\, r_i$. Following Golrezaei et al. (2014), we use $\Psi(x) = \frac{e}{e-1}(1 - e^{-x})$. This policy has a half-approximation guarantee.

To further improve the performance of the benchmarks, we divide the selling horizon into three equal segments and recompute the policy parameters at the beginning of each segment. For GR, for example, if the remaining capacities of the products at the beginning of a segment are $(C'_i : i \in \mathcal{N})$ and the set of remaining time periods in the selling horizon is $\mathcal{T}' \subseteq \mathcal{T}$, then we apply the recursive computation at the beginning of Section 3.1 after replacing C_i with C'_i and $\mathcal{T}$ with $\mathcal{T}'$, which yields new value function approximations. We use the new value function approximations until we reach the next segment, at which point, we recompute the policy parameters. We use a similar approach to recompute the policy parameters for the other benchmarks, except for MY and IB. MY does not have any policy parameters to compute. The function Ψ in IB is fixed a priori.

Results. Table 1 shows our computational results. The first column in this table labels the test problems by using (α, κ), where α and κ are as discussed earlier in this section. The second column shows the upper bound on the optimal total expected revenue provided by the optimal objective value of problem (13). The third through ninth columns show the total expected revenues obtained by GR, RO, BP, OS, DC, MY, and IB,

Table 1. Computational Results for Dynamic Assortment Management

Parameters	Upper	Total expected revenue							% Gain of RO over					
(α, κ)	bound	GR	RO	BP	OS	DC	MY	IB	GR	BP	OS	DC	MY	IB
(0.7, 0)	4,364	4,229	4,292	4,188	4,234	4,292	3,746	4,046	1.5	2.4	1.4	0.0*	12.7	5.7
(0.7, 0.01)	3,955	3,753	3,893	3,707	3,828	3,861	3,091	3,432	3.6	4.8	1.7	0.8	20.6	11.8
(0.7, 0.03)	3,925	3,726	3,854	3,572	3,783	3,858	2,691	3,168	3.3	7.3	1.8	−0.1*	30.2	17.8
(0.8, 0)	4,026	3,924	3,959	3,865	3,907	3,951	3,510	3,763	0.9	2.4	1.3	0.2*	11.3	5.0
(0.8, 0.01)	3,937	3,790	3,873	3,668	3,820	3,882	3,051	3,454	2.1	5.3	1.4	−0.2	21.2	10.8
(0.8, 0.03)	3,934	3,801	3,882	3,526	3,810	3,901	3,151	3,318	2.1	9.2	1.9	−0.5	18.8	14.5
(0.9, 0)	3,112	3,022	3,041	2,956	3,012	3,027	2,733	2,904	0.6	2.8	1.0	0.5	10.1	4.5
(0.9, 0.01)	2,809	2,718	2,773	2,645	2,732	2,779	2,325	2,487	2.0	4.6	1.5	−0.2	16.2	10.3
(0.9, 0.03)	3,084	2,977	3,040	2,914	2,994	3,073	2,511	2,603	2.1	4.1	1.5	−1.1	17.4	14.4
(1.0, 0)	3,027	2,940	2,978	2,804	2,916	2,931	2,694	2,848	1.3	5.8	2.1	1.6	9.5	4.4
(1.0, 0.01)	2,483	2,415	2,467	2,389	2,425	2,463	2,070	2,212	2.1	3.2	1.7	0.2	16.1	10.3
(1.0, 0.03)	2,971	2,891	2,946	2,829	2,904	2,967	2,382	2,513	1.9	4.0	1.4	−0.7	19.1	14.7
Average									2.0	4.7	1.5	0.0	16.9	10.4

Note. BP, bid-prices; DC, decomposition; GR, greedy policy; IB, inventory balancing; MY, myopic policy; OS, offer sets; RO, rollout policy.

which are estimated by simulating each benchmark over 1,000 sample paths. The remaining columns show the percent gaps between the total expected revenues obtained by RO and every other benchmark. The performance gaps except for those indicated with a star are statistically significant at the 95% level.

Our computational results indicate that RO performs quite well. By our use of separable and nonlinear value function approximations, RO noticeably improves the performance of GR, which uses linear value function approximations. Compared with BP and OS, which are based on the linear program in (13), RO provides average performance improvements of 4.7% and 1.5%, respectively. RO and DC are competitive, but to our knowledge, DC does not have a theoretical performance guarantee. Ignoring the future customer arrivals may result in inferior decisions, as indicated by the 16.9% average performance gap between RO and MY. The multiplicative revenue modifier $\Psi(q^t_{i,0}/C_i)$ used by IB does not depend on the future customer arrivals either. As a result, the average performance gap between RO and IB is 10.4%. When we compare RO with MY and IB, the performance gaps are particularly noticeable when κ is large; the more choosy customers in that case tend to arrive later, and we need to carefully protect inventory for these customers.

For GR, it takes 1.8 seconds on average to simulate its performance over one sample path. This computation time includes the time to recompute the policy parameters three times over the selling horizon and to solve problem (7) to find an assortment to offer at each time period. The same average computation time per sample path for RO is 4.1 seconds. The average computation times per sample path for BP, OS, and DC are 147.3, 149.7, and 240.6 seconds, respectively. The average computation times per sample path for MY and IB are 0.8 and 0.7 seconds, respectively. Thus, beside its favorable revenues, RO has quite fast computation times. In Online Appendix K, we give the details of all computation times. Golrezaei et al. (2014) discuss possible variants of IB. We experimented with these variants, but they did not provide qualitatively different results for our test problems. In Online Appendix L, we give our computational results on the variants of IB. In this section, the usage durations were infinite. In Online Appendix M, we test the performance of our policies under geometrically distributed usage durations with different means.

6.3. Street Parking Pricing in the City of Seattle

In our second set of computational experiments, we focus on the problem of dynamically pricing street-parking spaces. We treat the parking spaces within close proximity to each other as one product. After having been used by a driver for a certain duration of time, a parking space can be used by another driver, so the parking spaces are reusable products. The dynamics of the problem are as follows. When a driver arrives into the system with an intention to park in a certain region, as a function of the remaining parking-space inventory in the nearby regions, we decide on the prices to charge for the parking spaces in different regions. The driver is informed about the prices in real time, possibly through a smartphone application. The driver either parks at a particular parking space or decides to leave the system. If the driver parks, then the parking space generates revenue for a random usage duration. Our goal is to find a policy for deciding on the parking spaces to offer and their prices so that the total expected revenue is maximized.

Data. For brevity of discussion, we describe the essential elements of the data that we use, the approach that we use to augment the data for compliance with our modeling assumptions and the methodology that we use to estimate the model parameters. We defer the details to Online Appendix N. We build on the data provided by the Open Data Program in the city of Seattle; see City of Seattle (2017). Seattle uses parking rates that are dependent on the location and the time of day. Through the Open Data Program, we have transaction data on the use of the street parking spaces during 20 weekdays of June 2017. Each transaction record shows a parking event, documenting the start time, duration, and location of the parking event, along with the rate paid. We focus on 40 blocks in the downtown area between the hours of 11 a.m. and 4 p.m. We partition this area into 11-block clusters, each including approximately 4 blocks arranged in a 2-by-2 configuration. We refer to each 2-by-2 block cluster as a locale.

The street parking spaces in each locale correspond to a different product in our model. Thus, we have $n = 11$ products. To comply with our modeling assumptions, we augment the data from the Open Data Program as follows. We assume that each driver arrives into the system with the intention to park at a particular locale. The intended locale of a driver determines the type of the driver. In Section 5.1, we discussed the extension to multiple customer types. Because the intended locale of a driver determines her type, there are $m = 11$ customer types. In the data, we have access to the locale at which a driver actually parked, but we do not have access to the intended locale of a driver. For each driver, we randomly sample one of the five locales that are closest to the locale where she actually parked. We set the intended locale of the driver as this sampled locale. Once we augment the data in this way, each transaction record gives the start time, duration, intended locale, actual parked locale, and per-hour rate for each parking event. (The intended locale of a driver corresponds to the type of the customer, and the locale where a driver actually parks corresponds to the product the customer has chosen. If we had set the intended locale of a driver as the locale where she actually parked, then customers of a certain type would always be choosing the same product.) Because we augment the data from the Open Data Program, we caution the reader against comparing our results with the real operations in the city of Seattle.

The set of feasible locales we can offer to a driver are the five locales that are closest to her intended one. As a function of the remaining parking space inventories in these locales, we decide on the prices to charge for these locales. In Section 5.2, we discussed the extension of our model to the case in which we make pricing decisions. The driver either decides to park in one of these locales or leaves the system. If the driver parks, then we generate a certain revenue, depending on the parking duration and the charged price. Although we have discussed the extensions of our model to multiple customer types and to pricing decisions separately, it is not difficult to combine these extensions and to come up with a variant of our model that makes pricing decisions under multiple customer types. It is also not difficult to extend the linear program in (13) to the case in which we make pricing decisions under multiple customer types.

Experimental Setup. As discussed in Section 5.2, when making pricing decisions, we create multiple copies of each product, whereby the different copies correspond to charging different prices for the product. As $\mathcal{N}$ corresponds to the set of possible parking locales, using $\mathcal{H}$ to denote the set of possible prices that we can charge for a parking space, offering product copy $(i,h) \in \mathcal{N} \times \mathcal{H}$ represents charging price level h for locale i. We use $\pi_{i,h}$ to denote the per-period fee when we charge the price level h for locale i. We assume that the choices of the drivers are governed by the multinomial logit model. So, if we offer the assortment $S \subseteq \mathcal{N} \times \mathcal{H}$ of locale and price combinations to a driver arriving at time period t with intended locale j, then she chooses to park in locale i with probability

$$\phi_i^{t,j}(S) = \frac{e^{\alpha^j + \beta\,\pi_{i,h}}}{1 + \sum_{(\ell,g)\in S} e^{\alpha^j + \beta\,\pi_{\ell,g}}}$$

as long as $(i,h) \in S$. The parameter β, which captures the price sensitivity of the drivers, is assumed to be constant over all drivers.

Throughout the paper so far, we have assumed that there is one customer arrival at each time period. This assumption is not appropriate here because the arrival rate of the drivers varies during the day, but extending our model to the case in which there is at most one customer arrival at each time period is straightforward. We scale the time so that each time period in our model corresponds to a time interval of 30 seconds. A time interval of 30 seconds is short enough to ensure that there is at most one driver arrival in the region of our focus. We use $p^{t,j}$ to denote the probability that a driver with intended locale j arrives at time period t. We estimate the parameters β, $(\alpha^j : j \in \mathcal{M})$ and $(p^{t,j} : t \in \mathcal{T}, j \in \mathcal{M})$ by using maximum likelihood.

We model the parking duration in locale i as $1 + \mathsf{NegBin}(s_i, \eta_i)$, where $\mathsf{NegBin}(s_i, \eta_i)$ is a negative binomial random variable with parameters $s_i \in \mathbb{Z}_{++}$ and $\eta_i \in [0,1]$. As discussed in Section 4.3, if s_i is small, then we can perform rollout on the static policy in a tractable fashion. For each locale i, a negative binomial distribution with the parameter $s_i = 2$ provides a sensible fit.

Ultimately, in our experimental setup, we vary the length of the selling horizon over two values, 11 a.m.–2 p.m. and 11 a.m.–4 p.m. To obtain problems with different congestion levels, we scale the arrival rates with three different factors, 2.5, 3.0, and 3.5. Also, we vary the number of parking spaces over two values, 55 and 79. This experimental setup yields 12 parameter combinations for our test problems. From the rates used by the city of Seattle, the possible rates that we can charge are within the menu of \$2, \$4, and \$6 per hour.

Benchmarks. We use the benchmarks greedy policy, rollout policy, and offer sets, which are discussed in Section 6.2. We make the necessary modifications in these benchmarks to ensure that we can handle multiple customer types and we choose the prices of the offered products. The performances of bid-prices and myopic policy were not competitive. Decomposition and inventory balancing do not extend to reusable products. Thus, we drop these four benchmarks. We also add the following benchmark.

Fixed Price (FP). Here, we charge one fixed price for all locales at all time periods. We test the performance of the rates \$2, \$4, and \$6 per hour, which is the price menu used by the other benchmarks. We select the best constant price. This benchmark is not sophisticated, but it serves as a baseline. In all test problems, the rate \$4 per hour provided the best performance.

Results. Table 2 shows our computational results. The first column in this table labels the test problems by using $(\mathcal{T}, \sigma, C)$, where $\mathcal{T} \in$ {11 a.m.–2p.m., 11 a.m.–4 p.m.} is the selling horizon, $\sigma \in \{2.5, 3.0, 3.5\}$ is the multiplier for the arrival rates, and $C \in \{55, 79\}$ is the total number of parking spaces. The organization of the rest of the table closely mirrors that of Table 1. All of the performance gaps in Table 2 are statistically significant. To get a feel for the congestion in our test problems, noting that $\mathbb{E}\{\mathsf{Duration}_i\}$ is the expected

Table 2. Computational Results for Street Parking Pricing in the City of Seattle

Parameters		Total expected revenue				% Gain of RO over		
$(\mathcal{T}, \sigma, C)$	Upper bound	RO	GR	OS	FP	GR	OS	FP
(11 a.m.–2 p.m., 2.5, 79)	344	320	329	325	321	2.7	1.2	2.4
(11 a.m.–2 p.m., 3.0, 79)	410	375	385	379	375	2.6	1.6	2.6
(11 a.m.–2 p.m., 3.5, 79)	474	429	439	430	423	2.3	2.1	3.6
(11 a.m.–2 p.m., 2.5, 55)	338	301	306	300	296	1.6	2.0	3.3
(11 a.m.–2 p.m., 3.0, 55)	397	348	353	345	336	1.4	2.3	4.8
(11 a.m.–2 p.m., 3.5, 55)	452	390	396	385	370	1.5	2.8	6.6
(11 a.m.–4 p.m., 2.5, 79)	631	575	591	582	575	2.7	1.5	2.7
(11 a.m.–4 p.m., 3.0, 79)	749	673	689	674	667	2.3	2.2	3.2
(11 a.m.–4 p.m., 3.5, 79)	860	766	781	761	747	1.9	2.6	4.4
(11 a.m.–4 p.m., 2.5, 55)	613	534	543	528	520	1.7	2.8	4.2
(11 a.m.–4 p.m., 3.0, 55)	717	614	624	609	587	1.6	2.4	5.9
(11 a.m.–4 p.m., 3.5, 55)	812	686	696	679	644	1.4	2.4	7.5
Average						2.0	2.1	4.3

Note. FP, fixed price; GR, greedy policy; OS, offer sets; RO, rollout policy.

parking duration in locale i, we can estimate the number of times that we can turn over a parking space in locale i as $T/\mathbb{E}\{\text{Duration}_i\}$. The total expected demand for parking is $\sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{M}} p^{t,j}$. Thus, with C_i parking spaces available in locale i, the ratio between the total expected demand and the total available capacity is $\frac{\sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{M}} p^{t,j}}{\sum_{i\in\mathcal{N}} C_i T/\mathbb{E}\{\text{Duration}_i\}}$. For the test problems having the smallest demand and the largest capacity with $\sigma = 2.5$ and $C = 79$, this ratio is 0.72, whereas for the test problems having the largest demand and the smallest capacity with $\sigma = 3.5$ and $C = 55$, this ratio is 1.61.

Our results indicate that RO is consistently the strongest benchmark, providing average performance improvements of 2.0%, 2.1%, and 4.3% over GR, OS, and FP, respectively. Comparing RO with OS and FP, the performance gaps tend to be larger when σ is larger and C is smaller so that the system is more congested and the expected demand exceeds the available capacity by larger margins. For our test problems, depending on the length of the selling horizon, the time to compute the value functions $\{\hat{J}^t : t \in \mathcal{T}\}$ for GR ranges from 362 to 672 seconds. The time to compute the value functions $\{V_i^t : i \in \mathcal{N}, t \in \mathcal{T}\}$ for RO ranges from 1,550 to 17,201 seconds. For OS, the time to solve the linear program in (13) ranges from 894 to 6,535 seconds. A few preliminary runs indicated that the performance did not noticeably improve for any of the benchmarks when we recomputed the policy parameters. Because the run times were relatively long, we did not recompute the policy parameters. Overall, the computation times for RO are significantly longer, but by using nonlinear value function approximations, RO can provide significant revenue improvements.

7. Conclusions

We studied dynamic assortment problems with reusable products and provided policies with half-approximate performance guarantees. A natural question that arises is what features of the rewards and transition dynamics make our half-approximation guarantees go through. In Online Appendix O, we give conditions on the rewards and transition dynamics that allow us to obtain our half-approximation guarantees. Our conditions on the transition dynamics, in particular, only require the expected transition dynamics to be linear in the state, along with a certain decoupling property between the effects of the actions and the current state on the transition dynamics. Considering future research, our rollout approach decomposes the problem by the products, which is reminiscent of dynamic-programming decomposition techniques in revenue management. To our knowledge, existing decomposition techniques do not provide any performance guarantees. An exciting research area is to construct decomposition techniques with performance guarantees for other revenue management problems. We were able to give an improved performance guarantee for a tailored variant of our rollout policy under infinite usage durations. Our efforts to extend the tailored variant to arbitrary usage duration distributions were not yet fruitful. It would be interesting to give stronger performance guarantees for our rollout approach under arbitrary usage duration distributions.

Acknowledgments

The authors thank the area editor, Professor Yinyu Ye, the associate editor, and three anonymous referees for their useful comments, which improved the paper in many ways.

References

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

Aouad A, Levi R, Segev D (2019) Approximation algorithms for dynamic assortment optimization models. *Math. Oper. Res.* 44(2): 487–511.

Aouad A, Levi R, Segev D (2018) Greedy-like algorithms for dynamic assortment planning under multinomial logit preferences. *Oper. Res.* 66(5):1321–1345.

Bertsekas DP, Tsitsiklis JN (1996) *Neuro-Dynamic Programming* (Athena Scientific, Belmont, MA).

Chen Y, Levi R, Shi C (2017) Revenue management of reusable resources with advanced reservations. *Production Oper. Management.* 26(5):836–859.

City of Seattle (2017) Seattle parking transactions. Accessed August 1, 2017, https://data.seattle.gov/Transportation/Seattle-Parking-Transactions/updn-y53g.

Dai J, Ding W, Kleywegt AJ, Wang X, Zhang Y (2014) Choice based revenue management for parallel flights. Technical report, Georgia Tech, Atlanta, GA.

Davis JM, Gallego G, Topaloglu H (2013) Assortment planning under the multinomial logit model with totally unimodular constraint structures. Technical report, Cornell University, Ithaca, NY.

Davis JM, Gallego G, Topaloglu H (2014) Assortment optimization under variants of the nested logit model. *Oper. Res.* 62(2): 250–273.

Desire A, Goyal V, Zhang J (2016) Near-optimal algorithms for capacity constrained assortment optimization. Technical report, Columbia University, New York.

Feldman JB, Topaloglu H (2015) Technical note: Capacity constraints across nests in assortment optimization under the nested logit model. *Oper. Res.* 63(4):812–822.

Gallego G, Topaloglu H (2014) Constrained assortment optimization for the nested logit model. *Management Sci.* 60(10):2583–2601.

Gallego G, van Ryzin G (1994) Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Sci.* 40(8):999–1020.

Gallego G, Ratliff R, Shebalov S (2015) A general attraction model and sales-based linear programming formulation for network revenue management under customer choice. *Oper. Res.* 63(1): 212–232.

Gallego G, Iyengar G, Phillips R, Dubey A (2004) Managing flexible products on a network. CORC Technical Report TR-2004-01, Columbia University, New York.

Gallego G, Li A, Truong VA, Wang X (2016) Online bipartite matching. Working paper, Columbia University, New York.

Golrezaei N, Nazerzadeh H, Rusmevichientong P (2014) Real-time optimization of personalized assortments. *Management Sci.* 60(6): 1532–1551.

Goyal V, Levi R, Segev D (2016) Near-optimal algorithms for the assortment planning problem under dynamic substitution and stochastic demand. *Oper. Res.* 64(1):219–235.

Honhon D, Gaur V, Seshadri S (2010) Assortment planning and inventory decisions under stockout-based substitution. *Oper. Res.* 58(5):1364–1379.

Lei Y, Jasin S (2016) Real-time dynamic pricing for revenue management with reusable resources and deterministic service time requirements. Technical report, University of Michigan, Ann Arbor.

Levi R, Radovanovic A (2010) Provably near-optimal LP-based policies for revenue management in systems with reusable resources. *Oper. Res.* 58(2):503–507.

Li G, Rusmevichientong P, Topaloglu H (2015) The *d*-level nested logit model: Assortment and price optimization problems. *Oper. Res.* 63(2):325–342.

Liu Q, van Ryzin GJ (2008) On the choice-based linear programming model for network revenue management. *Manufacturing Service Oper. Management.* 10(2):288–310.

Mahajan S, van Ryzin G (2001) Stocking retail assortments under dynamic customer substitution. *Oper. Res.* 49(3):334–351.

Owen Z, Simchi-Levi D (2017) Price and assortment optimization for reusable resources. Technical report, Massachusetts Institute of Technology, Cambridge.

Ozer O, Phillips R, eds. (2012) *The Oxford Handbook of Pricing Management* (Oxford University Press, Oxford, UK).

Rusmevichientong P, Shmoys D, Tong C, Topaloglu H (2014) Assortment optimization under the multinomial logit model with random choice parameters. *Production Oper. Management.* 23(11): 2023–2039.

Stein C, Truong VA, Wang X (2019) Advance reservations with heterogeneous customers. *Management Sci.*, ePub ahead of print October 15, https://doi.org/10.1287/mnsc.2019.3364.

Talluri K, van Ryzin GJ (2004) Revenue management under a general discrete choice model of consumer behavior. *Management Sci.* 50(1):15–33.

Talluri KT, van Ryzin GJ (2005) *The Theory and Practice of Revenue Management* (Kluwer Academic Publishers, Boston).

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 (2013) Joint stocking and product offer decisions under the multinomial logit model. *Production Oper. Management.* 22(5): 1182–1199.

van Ryzin G, Mahajan S (1999) On the relationship between inventory costs and variety benefits in retail assortments. *Management Sci.* 45(11):1496–1509.

Vossen TWM, Zhang D (2015) Reductions of approximate linear programs for network revenue management. *Oper. Res.* 63(6): 1352–1371.

Wang X, Truong VA, Bank D (2016) Online advance admission scheduling for services, with customer preferences. Working paper, Columbia University, New York.

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

Zhang D, Cooper WL (2005) Revenue management for parallel flights with customer choice behavior. *Oper. Res.* 53(3):415–431.

Zhang H, Rusmevichientong P, Topaloglu H (2017) Assortment optimization under the paired combinatorial logit model. Technical report, Cornell University, Ithaca, NY.