Technical Note: Capacitated Assortment Optimization under the Multinomial Logit Model with Nested Consideration Sets

Jacob Feldman
Olin Business School, Washington University,
St. Louis, MO 63130, USA
jbfeldman@wustl.edu

Huseyin Topaloglu
School of Operations Research and Information Engineering, Cornell Tech,
New York, NY 10011, USA
topaloglu@orie.cornell.edu

May 10, 2017

Abstract

We study capacitated assortment problems when customers choose under the multinomial logit model with nested consideration sets. In this choice model, there are multiple customer types and a customer of a particular type is interested in purchasing only a particular subset of products. We use the term consideration set to refer to the subset of products that a customer of a particular type is interested in purchasing. The consideration sets of customers of different types are nested in the sense that the consideration set of one customer type is included in the consideration set of another. The choice process for customers of different types is governed by the same multinomial logit model except for the fact that customers of different types have different consideration sets. Each product, if offered to the customers, occupies a certain amount of space. The sale of each product generates a certain amount of revenue. Given that the customers choose among the offered products according to the multinomial logit model with nested consideration sets, the goal of the assortment problem is to find a set of products to offer to maximize the expected revenue obtained from a customer, while making sure that the total space consumption of the offered products does not exceed a certain limit. We show that this assortment problem is NP-hard, even when there is no limit on the total space consumption of the offered products. Motivated by this complexity result, we give a fully polynomial time approximation scheme for the problem.

1 Introduction

Incorporating customer choice behavior into revenue management models has been seeing considerable attention. Traditional revenue management models represent the demand for a product by using an exogenous random variable, but this way of representing the demand may be inadequate when there are multiple products that can serve the needs of a customer and the customers choose and substitute among the offered products. When the customers choose and substitute among the offered products, the demand for a particular product depends on what other products are offered, creating complex interactions between the demands for the different products. There are numerous field studies that demonstrate the benefits from taking the customer choice process into consideration when making operational decisions. For example, Chong et al. (2001), Kok and Fisher (2007) and Misra (2008) focus on customers purchasing groceries in different food categories. These papers indicate that there is significant potential for revenue improvement by explicitly taking the choice process of the customers into consideration, when deciding which assortment of products

to make available. Similarly, Vulcano et al. (2010) focus on customers purchasing airline tickets and demonstrate that there is significant potential for revenue improvement when one explicitly accounts for the choice process of the customers among the different itineraries. When taking the choice process of the customers into consideration, a critical tradeoff is that a sophisticated choice model may capture the choice behavior of the customers faithfully, but solving operational problems under a sophisticated choice model may be intractable.

In this paper, we study capacitated assortment problems when customers choose among the offered products according to the multinomial logit model with nested consideration sets. In our assortment problem, a firm has access to a set of products among which it picks an assortment of products to offer to its customers. Each product, if offered, occupies a certain amount of space. The sale of each product generates a certain amount of revenue. Customers choose among the offered assortment of products according to a particular choice model. The goal of the firm is to pick an assortment of products to offer to maximize the expected revenue obtained from a customer, while making sure that the total space consumption of the offered products does not exceed a certain limit. We capture the choice process of the customers by using the multinomial logit model with nested consideration sets. In this choice model, there are multiple customer types. A customer of a particular type is interested in purchasing only a particular subset of products. We use the term consideration set to refer to the subset of products that a customer of a particular type is interested in purchasing. The consideration sets of customers of different types are nested, so that we can index the customer types such that the consideration set of a customer type with a smaller index is included in the consideration set of a customer type with a larger index. The choice process for customers of different types is governed by the same multinomial logit model except for the fact that customers of different types have different consideration sets. A customer observes which of the products in her consideration set are actually included in the offered assortment and she makes a choice only among those products according to the multinomial logit model.

Main Contributions. We show that the problem of finding the assortment of products that maximizes the expected revenue obtained from a customer is NP-hard. This complexity result holds even when there is no limit on the total space consumption of the offered products. Motivated by our complexity result, we give a fully polynomial time approximation scheme (FPTAS) for the assortment problem. This FPTAS appears to be the first to address assortment problems with consideration sets, while having a running time that is polynomial in the number of customer

types. Our FPTAS is based on a dynamic programming formulation of the assortment problem. In particular, since the consideration sets of different customer types are nested, we index the customer types such that the first customer type has the smallest consideration set and the last customer type has the largest consideration set. In our dynamic program, the decision epochs correspond to the customer types in the order of increasing consideration sets. In each decision epoch, we decide whether we offer the product that is in the consideration set of the current customer type, but not in the consideration sets of the previous customer types. To develop an FPTAS, we discretize the state space of the dynamic program by using a geometric grid. Ultimately, we show that if there are n products among which the firm picks an assortment, then we obtain a solution that provides a $1-\epsilon$ fraction of the optimal expected revenue in $O(n^4 \log(nR_{\max})\ \log(nV_{\max}) \log(nR_{\max} V_{\max}/\lambda_{\min}) \,/\, \epsilon^3)$ operations, where $R_{\max}$ is the largest value for the product of the revenue and the preference weight of a product, $V_{\max}$ is the largest value for the preference weight of a product and $\lambda_{\min}$ is the smallest value for the probability of arrival for a particular customer type.

It turns out that if we do not have a limit on the total space consumption of the offered products, then we can solve our dynamic program more efficiently. We show that the running time of our FPTAS given in the previous paragraph reduces to $O(n^3 \log(nR_{\max})\ \log(nV_{\max}) \,/\, \epsilon^2)$ operations, when we do not have a limit on the total space consumption of the offered products.

Multinomial Logit Model with Nested Consideration Sets. In our choice model, the consideration sets of customers of different types are nested. In particular, assuming that there are m customer types and using N_k to denote the consideration set of customers of type k, we index the customer types $\{1,\ldots,m\}$ such that $N_1 \subseteq \ldots \subseteq N_m$. Assuming that there are n products, we index the products $\{1,\ldots,n\}$ such that products $\{1,\ldots,|N_1|\}$ correspond to the products in N_1, products $\{|N_1|+1,\ldots,|N_2|\}$ correspond to the products in $N_2 \setminus N_1$ and so on. So, without loss of generality, we can assume that the consideration set of customers of a particular type is of the form $\{1,\ldots,j\}$ for some $j = 1,\ldots,n$. In this case, since the customer types differ in their consideration sets, the number of customer types is at most n. Without loss of generality, we can assume that the number of customer types is n, if necessary, by defining additional dummy customer types.

Consideration sets of the form $\{1,\ldots,j\}$ for some $j = 1,\ldots,n$ become useful when different customers drop products from consideration based on cutoff values for a particular attribute, such as price or quality. For example, ordering the products such that product 1 has the lowest price and product n has the highest price, customers may form their consideration sets by focusing

on the products within their budget. A customer with a consideration set $\{1, \ldots, j\}$ focuses on products whose prices do not exceed the price of product j. Similarly, ordering the products such that product 1 has the highest quality and product n has the lowest quality, a customer with a consideration set $\{1, \ldots, j\}$ focuses on products whose quality is no worse than the quality of product j. Goyal et al. (2016) also use such nested consideration sets.

Another feature of our choice model is that the choices of the customers of different types are governed by the same multinomial logit model, except for the fact that customers of different types may have different consideration sets and associate different mean utilities with the no purchase option. This choice model naturally arises when the mean utility that a customer of a particular type associates with a particular product is a *separable function* of the features of the product and the features of the customer type. Under the multinomial logit model, a customer of type k associates the mean utility μ_{jk} with product j. If the set of offered products that are in the consideration set of customer type k is given by S, then a customer of type k purchases product j with probability $e^{\mu_{jk}}/(e^{\mu_{0k}} + \sum_{i \in S} e^{\mu_{ik}})$, where μ_{0k} is the mean utility that a customer of type k associates with the no purchase option. If μ_{jk} is a separable function of the features of the product and the features of the customer type so that $\mu_{jk} = \eta_j + \sigma_k$ for some η_j and σ_k, then the last probability becomes $e^{\eta_j + \sigma_k}/(e^{\mu_{0k}} + \sum_{i \in S} e^{\eta_i + \sigma_k}) = e^{\eta_j}/(e^{\mu_{0k} - \sigma_k} + \sum_{i \in S} e^{\eta_i})$, which is the purchase probability of product j when customers of all types associate the same mean utility η_j with product j, but customers of type k associate the mean utility $\mu_{0k} - \sigma_k$ with the no purchase option.

To give a concrete example for our choice model, we consider the choice process of customers when they shop for canned coffee from a particular store. Different coffee options are characterized by their prices and qualities. We use r_j to denote the price and q_j to denote the quality of coffee option j. Different customer types are characterized by their daily coffee consumption amounts and shopping frequencies at the particular store. We use a_k to denote the daily coffee consumption amount and f_k to denote the shopping frequency of customers of type k. One possible model for the mean utility μ_{jk} that a customer of type k associates with coffee option j is to let $\mu_{jk} = \beta^1 r_j + \beta^2 q_j + \beta^3 a_k + \beta^4 f_k$, where the parameters $\{\beta^1, \ldots, \beta^4\}$ are estimated from the past purchase data. Identifying $\beta^1 r_j + \beta^2 q_j$ with σ_j and $\beta^3 a_k + \beta^4 f_k$ with η_k, we obtain the choice model in the previous paragraph. In addition, if different customers drop coffee options from consideration based on different cutoff values for the price, then we obtain the multinomial logit model with nested consideration sets. Since the customer types are characterized by their daily coffee consumption

amounts and shopping frequencies, as well as cutoff values for the price, there can be multiple customer types with the same consideration set, but this does not create a complication for our approach. An advantage of this model is that once we estimate the parameters $\{\beta^1, \ldots, \beta^4\}$, given any coffee option with certain price and quality and given any customer type with certain daily coffee consumption amount and shopping frequency, including those that are not in the past purchase data, we can immediately estimate the mean utility that this particular customer attaches to this particular coffee option. A shortcoming of this model is that this model captures the fact that the mean utility that a customer attaches to a coffee option depends on its price and quality, but this dependence is, intuitively speaking, captured in terms of the reaction of an "average" customer, since the parameters β^1 and β^2 do not depend on the type of a customer. One option is to use parameters of the form β^1_k and β^2_k, capturing the reaction of customers of type k to price and quality, but we are not able to give a tractable approach for the assortment problem in this case.

Literature Review. There is recent work on models where customers form consideration sets. Aouad et al. (2016) work with consider-then-choose choice models, where each customer has a consideration set and a ranking of the products in her consideration set. She purchases the most preferred available product in her consideration set. The authors study the assortment problem for a variety of possible consideration set and ranking structures. Jagabathula and Vulcano (2015) develop a choice model, where the consideration set of a customer includes only the product purchased during the last visit and the products in promotion, if there are any. They focus on estimating the parameters of their choice model. Jagabathula and Rusmevichientong (2015) use a choice model where a customer considers only the products whose prices are below a certain threshold. The authors focus on parameter estimation and pricing problems. Sahin and Wang (2014) develop a choice model that incorporates the product search cost so that the set of products that a customer considers purchasing can be different from what the firm offers. Dai et al. (2014) develop a revenue management model, where a customer may not even consider purchasing some of the offered itineraries due to personal restrictions on, for example, time of departure.

In a mixture of multinomial logit models, there are multiple customer types and customers of different types choose according to different multinomial logit models. The multinomial logit models for the different customer types may have completely different parameters and consideration sets. McFadden and Train (2000) show that a broad class of choice models can be approximated arbitrarily accurately by using a mixture of multinomial logit models. Our choice model is a special

case of a mixture of multinomial logit models, where the consideration sets are nested and customers of different types choose according to the same multinomial logit model except for the fact that their consideration sets are different. Talluri and van Ryzin (2004) and Gallego et al. (2004) show that assortment problems under the multinomial logit model with a single customer type can be solved efficiently. Bront et al. (2009), Desir and Goyal (2014) and Rusmevichientong et al. (2014) give approximation schemes and heuristics for assortment problems under a mixture of multinomial logit models. The running times of their approximation schemes increase exponentially with the number of customer types, whereas our running times increase polynomially.

Network revenue management problems focus on dynamically controlling the availability of products when there are multiple resources and the sale of a product consumes the capacity of a combination of resources. Gallego et al. (2004) give a deterministic linear programming approximation for network revenue management problems, where the number of decision variables increases exponentially with the number of products. Thus, the deterministic approximation is commonly solved by using column generation. The column generation subproblem has the same structure as our assortment problem when customers choose under the multinomial logit model with nested consideration sets. Gallego et al. (2016) discuss that if we can solve the column generation subproblem with a certain optimality gap, then we can also solve the deterministic approximation with a certain optimality gap. Talluri (2011) focuses on the deterministic approximation when the consideration sets of different customer types are disjoint or have small overlaps. Meissner et al. (2012) study a tractable relaxation of the deterministic approximation and add the so called product cuts to tighten it. Stauss and Talluri (2016) use a graph to represent the overlap between the consideration sets of different customer types and focus on solving the deterministic approximation when this graph is a tree. Kunnumkal and Talluri (2014) use a relaxation of the dynamic programming formulation of the network revenue management problem, which yields a tractable approach when the consideration set of each customer type is small.

Organization. In Section 2, we give a formulation for our assortment problem and show that our assortment problem is NP-hard. In Section 3, we formulate our assortment problem as a dynamic program and give an approximation to this dynamic program. In Section 4, we show how to use the approximate dynamic program to estimate the optimal expected revenue with a certain relative gap. In Section 5, we use our analysis of the approximate dynamic program to develop our FPTAS. In Section 6, we give a numerical study for our FPTAS. In Section 7, we conclude.

2 Assortment Problem and Computational Complexity

In our assortment problem, there are n products indexed by $\{1,\ldots,n\}$. Associated with product j, we have a revenue r_j and a space consumption c_j. The limit on the total space consumption of the offered products is C. We use the vector $x = (x_1,\ldots,x_n) \in \{0,1\}^n$ to capture the set of products offered to customers, where $x_j = 1$ if product j is offered and $x_j = 0$ if product j is not offered. There are n customer types indexed by $\{1,\ldots,n\}$. A customer arriving into the system is of type k with probability λ_k. A customer of type k considers purchasing only the products in the set $\{1,\ldots,k\}$. We refer to the set $N_k = \{1,\ldots,k\}$ as the consideration set of a customer of type k. A customer of type k chooses among the products in her consideration set according to the multinomial logit model. In particular, using v_j to denote the preference weight of product j and v_0 to denote the preference weight of the no purchase option, if the set of products offered to the customers is given by the vector x, then a customer of type k chooses product $j \in N_k$ with probability $P_{jk}(x) = v_j\, x_j/(v_0 + \sum_{i\in N_k} v_i\, x_i)$. Thus, if the set of products offered to the customers is given by the vector x, then we obtain an expected revenue of $\sum_{k=1}^n \lambda_k \sum_{j\in N_k} r_j\, P_{jk}(x)$ from a customer. Our goal is to find the set of products to offer to maximize the expected revenue obtained from each customer, while making sure that the total space consumption of the offered products does not exceed the limit on the total space consumption, yielding the problem

$$z^* = \max_{\substack{x\in\{0,1\}^n:\\ \sum_{j=1}^n c_j\, x_j \le C}} \left\{\sum_{k=1}^n \lambda_k \left\{\sum_{j\in N_k} r_j\, P_{jk}(x)\right\}\right\} = \max_{\substack{x\in\{0,1\}^n:\\ \sum_{j=1}^n c_j\, x_j \le C}} \left\{\sum_{k=1}^n \lambda_k \left\{\frac{\sum_{j=1}^k r_j\, v_j\, x_j}{v_0 + \sum_{j=1}^k v_j\, x_j}\right\}\right\}. \quad (1)$$

In problem (1), we can allow the preference weight v_0 of the no purchase option to depend on the customer type. In particular, if a customer of type k associates the preference weight v_{0k} with the no purchase option, then we can simply replace v_0 in problem (1) with v_{0k}. All of our results continue to hold under this extension with essentially no modifications and we obtain an FPTAS with the same running time.

Computational Complexity. In problem (1), the consideration sets are nested and customers of different types associate the same preference weight with a particular product, as long as this product is in their consideration sets. Rusmevichientong et al. (2014) consider assortment problems under a mixture of multinomial logit models, where consideration sets are arbitrary and customers of different types associate arbitrary preference weights with a particular product. The

authors show that the feasibility version of their assortment problem is NP-complete even if there are only two customer types. In the remainder of this section, we show that the feasibility version of problem (1) is also NP-complete, despite the fact this problem has the special structure where the consideration sets are nested and customers of different types associate the same preference weight with a particular product. This complexity result holds even when there is no limit on the total space consumption of the offered products. To discuss the computational complexity of problem (1), we focus on the following feasibility version of the problem.

Assortment Feasibility. We are given an expected revenue threshold K. The assortment feasibility problem asks whether there exists a vector $x \in \{0,1\}^n$ that provides an expected revenue of K or more in problem (1).

In the next theorem, we show that the feasibility version of problem (1) is NP-complete. We defer the proof to Appendix A. The proof of this theorem uses a reduction from the partition problem, which is a well-known NP-complete problem; see Garey and Johnson (1979).

Theorem 1 *The assortment feasibility problem is NP-complete.*

The computational complexity result in Theorem 1 motivates us to look for an FPTAS for problem (1). In the next section, we give a dynamic programming formulation for problem (1). This dynamic program forms the starting point for our FPTAS.

3 Dynamic Programming Formulation

To develop an FPTAS for problem (1), we follow three steps. First, we formulate problem (1) as a dynamic program, but there are exponentially many possible values for the state variable in this dynamic programming formulation. So, we give an approximation to the dynamic programming formulation, where there are polynomially many possible values for the state variable. Second, noting that z^* is the optimal expected revenue in problem (1), we show that we can use the value functions in the approximate dynamic program to obtain an estimate $\tilde{z}$ of the optimal expected revenue that satisfies $\tilde{z} \geq (1-\epsilon)\, z^*$ for an appropriate choice of $\epsilon \in (0,1)$. Third, we follow the optimal state and action trajectory in the approximate dynamic program to obtain a feasible solution to problem (1). Also, using Rev to denote the expected revenue corresponding to this feasible solution, we show that $\textsc{Rev} \geq \tilde{z}$. In this case, we obtain $\textsc{Rev} \geq \tilde{z} \geq (1-\epsilon)\, z^*$, which

bounds the relative gap between the expected revenue from the solution obtained by using the approximate dynamic program and the optimal expected revenue. In this section, we focus on the first step. In the next two sections, we focus on the second and third steps.

To formulate problem (1) as a dynamic program, assume that we have already decided which of the products in $\{1, \ldots, j-1\}$ are offered to the customers and these decisions are given by $(x_1, \ldots, x_{j-1}) \in \{0,1\}^{j-1}$, where $x_k = 1$ if product k is offered and $x_k = 0$ if product k is not offered. We use P_j to denote $\sum_{k=1}^{j-1} r_k \, v_k \, x_k$ and S_j to denote $\sum_{k=1}^{j-1} v_k \, x_k$. In this case, the expected revenue from a customer of type j is given by $\lambda_j \sum_{k=1}^{j} r_k \, v_k \, x_k / (v_0 + \sum_{k=1}^{j} v_k \, x_k) = \lambda_j \, (P_j + r_j \, v_j \, x_j)/(v_0 + S_j + v_j \, x_j)$. Therefore, once we decide whether product j is offered, we can compute the expected revenue from a customer of type j as a function of only P_j, S_j and x_j. Furthermore, we can compute $P_{j+1} = \sum_{k=1}^{j} r_k \, v_k \, x_k$ and $S_{j+1} = \sum_{k=1}^{j} v_k \, x_k$ as $P_{j+1} = P_j + r_j \, v_j \, x_j$ and $S_{j+1} = S_j + v_j \, x_j$, which are also functions of only P_j, S_j and x_j. So, given that the decisions for the products in $\{1, \ldots, j-1\}$ satisfy $P_j = \sum_{k=1}^{j-1} r_k \, v_k \, x_k$ and $S_j = \sum_{k=1}^{j-1} v_k \, x_k$ and we want to generate an expected revenue of R_j or more from customers of type $\{j, \ldots, n\}$, we let $V_j(P_j, S_j, R_j)$ be the minimum total space consumption of the products in $\{j, \ldots, n\}$ to do so. We can compute $\{V_j(\cdot, \cdot, \cdot) : j = 1, \ldots, n\}$ by using the dynamic program

$$V_j(P_j \, , S_j, R_j) = \min_{x_j \in \{0,1\}} \left\{ c_j \, x_j + V_{j+1}\Big(P_j + r_j \, v_j \, x_j \, , \, S_j + v_j \, x_j \, , \, R_j - \lambda_j \, \tfrac{P_j + r_j \, v_j \, x_j}{v_0 + S_j + v_j \, x_j}\Big) \right\}, \qquad (2)$$

with the boundary condition that $V_{n+1}(\cdot, \cdot, R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $V_{n+1}(\cdot, \cdot, R_{n+1}) = \infty$ if $R_{n+1} > 0$. The dynamic program above follows from the following reasoning. Given that the decisions for the products in $\{1, \ldots, j-1\}$ satisfy $P_j = \sum_{k=1}^{j-1} r_k \, v_k \, x_k$ and $S_j = \sum_{k=1}^{j-1} v_k \, x_k$, as mentioned above, the values of $P_{j+1} = \sum_{k=1}^{j} r_k \, v_k \, x_k$ and $S_{j+1} = \sum_{k=1}^{j} v_k \, x_k$ can be computed as $P_{j+1} = P_j + r_j \, v_j \, x_j$ and $S_{j+1} = S_j + v_j \, x_j$. Furthermore, if we want to generate an expected revenue of R_j or more from customers of type $\{j, \ldots, n\}$, then after making the decision for product j, the expected revenue that we want to generate from customers of type $\{j+1, \ldots, n\}$ is at least $R_j - \lambda_j \, \frac{P_j + r_j \, v_j \, x_j}{v_0 + S_j + v_j \, x_j}$. The boundary condition follows from the fact that if we make the decisions for all of the products and we still have a strictly positive expected revenue that we want to generate, then there is no set of products that provides the expected revenue that we want to generate. We refer to $V_j(\cdot, \cdot, \cdot)$ as the value function and (P_j, S_j, R_j) as the state variable for decision epoch j. To compute the value functions $\{V_j(\cdot, \cdot, \cdot) : j = 1, \ldots, n\}$, we move over the decision epochs in reverse order. For all $j = n, \ldots, 1$, given that we know the value function $V_{j+1}(\cdot, \cdot, \cdot)$, we compute the value

function $V_j(\cdot,\cdot,\cdot)$. Note that $V_1(0,0,R_1)$ is the minimum total space consumption of the products in $\{1,\ldots,n\}$ given that we want to generate an expected revenue of R_1 or more from customers of type $\{1,\ldots,n\}$. Thus, the optimal objective value of problem (1) is given by

$$z^* = \max_{z\in\Re_+}\Big\{z : V_1(0,0,z) \leq C\Big\}, \tag{3}$$

which is the largest expected revenue that we can generate from customers of type $\{1,\ldots,n\}$ while ensuring that the total space consumption of the products in $\{1,\ldots,n\}$ does not exceed C. It is important to emphasize that the dynamic program in (2) does not provide a tractable solution approach for problem (1). In particular, since there are exponentially many possible sets of products that we can offer to the customers, there are exponentially many possible values for the state variable (P_j, S_j, R_j) as well, which implies that computing the value function $V_j(P_j, S_j, R_j)$ for all possible values for the state variable (P_j, S_j, R_j) is intractable. In the remainder of this section, we give an approximate version of the dynamic program in (2), which is based on discretizing the state space in the original dynamic program. The approximate dynamic program forms the basis for the FPTAS that we develop for problem (1).

To approximate the dynamic program in (2), we choose a value of $\rho > 0$ and focus on the grid points in the domain $\mathsf{Dom}^\rho = \{0\} \cup \{(1+\rho)^k : k = \ldots,-1,0,1,\ldots\}$, which is a geometric grid of size $1+\rho$ augmented by an additional point at zero. The specific choice of ρ is determined later, but we hint that the ultimate choice of $\rho = \epsilon/6n$ will yield a performance guarantee of $1-\epsilon$ for $\epsilon \in (0,1)$. We define the round down operator $\lfloor\cdot\rfloor$ that rounds its argument down to the closest value in Dom^ρ. In other words, we have $\lfloor x\rfloor = \max\{y \in \mathsf{Dom}^\rho : y \leq x\}$. Similarly, we define the round up operator $\lceil\cdot\rceil$ that rounds its argument up to the closest value in Dom^ρ, which is given by $\lceil x\rceil = \min\{y \in \mathsf{Dom}^\rho : y \geq x\}$. The results of the round up and round down operators $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ depend on the value of ρ, but for notational brevity, we do not make this dependence explicit. We consider an approximate version of the dynamic program in (2) given by

$$\Theta_j(P_j\,, S_j, R_j) = \min_{x_j\in\{0,1\}}\left\{c_j\,x_j + \Theta_{j+1}\Big(\lfloor P_j + r_j\,v_j\,x_j\rfloor\,, \lceil S_j + v_j\,x_j\rceil\,, \lceil R_j - \lambda_j\,\tfrac{P_j + r_j\,v_j\,x_j}{v_0 + S_j + v_j\,x_j}\rceil\Big)\right\}, \tag{4}$$

with the boundary condition that $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$ if $R_{n+1} > 0$. On the right side above, we observe that we round the first component of the state variable down, whereas we round the second and third components of the state variable up, which

ultimately allows us to bound the gap between the value functions $V_j(\cdot,\cdot,\cdot)$ and $\Theta_j(\cdot,\cdot,\cdot)$ in (2) and (4). Noting that the results of the round up and down operators depend on the value of ρ, the value functions $\{\Theta_j(\cdot,\cdot,\cdot) : j = 1,\ldots,n\}$ depend on the value of ρ as well, but for notational brevity, we do not make this dependence explicit. We can bound each component of the state variable (P_j, S_j, R_j) in the dynamic program in (4). In particular, without loss of generality, we assume that $r_j \geq 1$ and $v_j \geq 1$ for all $j = 1,\ldots,n$. If either one of these conditions does not hold, then we can scale up all revenues or all preference weights by the same factor without changing the optimal solution to problem (1). For notational brevity, we let $R_{\max} = \max\{r_j\, v_j : j = 1,\ldots,n\}$ and $V_{\max} = \max\{v_j : j = 0,\ldots,n\}$. In the next lemma, we give upper bounds on each component of the state variable (P_j, S_j, R_j). The proof of this lemma uses a simple induction over the decision epochs and we defer the proof to Appendix B.

Lemma 2 *For any $(x_1,\ldots,x_n) \in \{0,1\}^n$ and $R_1 \in \Re_+$, assume that $\{(P_j, S_j, R_j) : j = 1,\ldots,n\}$ are computed as $P_{j+1} = \lfloor P_j + r_j\, v_j\, x_j \rfloor$, $S_{j+1} = \lceil S_j + v_j\, x_j \rceil$ and $R_{j+1} = \lceil R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil$ with $P_1 = 0$ and $S_1 = 0$. Then, letting $\gamma = (\exp(n\rho) - 1)/\rho$, we have $P_j \leq n R_{\max}$, $S_j \leq \gamma\, V_{\max}$ and $R_j \leq R_{n+1} + (\lambda_j + \ldots + \lambda_n)\, R_{\max}$ for all $j = 1,\ldots,n$.*

We can use Lemma 2 to give upper and lower bounds on each component of the state variable (P_j, S_j, R_j) as follows. Since $r_j \geq 1$ and $v_j \geq 1$ for all $j = 1,\ldots,n$, the smallest nonzero values of P_j and S_j are one. Also, by Lemma 2, we have $P_j \leq n R_{\max}$ and $S_j \leq \gamma\, V_{\max}$. Therefore, we can assume that $P_j \in \{0\} \cup [1, n R_{\max}]$ and $S_j \in \{0\} \cup [1, \gamma\, V_{\max}]$ for all $j = 1,\ldots,n$. Also, if $R_j > R_{\max}$, then by Lemma 2, we have $R_{n+1} \geq R_j - (\lambda_j + \ldots + \lambda_n)\, R_{\max} > (1 - (\lambda_j + \ldots + \lambda_n))\, R_{\max} \geq 0$. In this case, noting the boundary condition of the dynamic program in (4), if $R_j > R_{\max}$, then we immediately have $\Theta_j(\cdot,\cdot,R_j) = \infty$. In other words, there is no need to compute $\Theta_j(\cdot,\cdot,R_j)$ explicitly for $R_j > R_{\max}$. On the other hand, we let $\lambda_{\min} = \min\{\lambda_j : j = 1,\ldots,n\}$ for notational brevity. By the discussion in this paragraph, we have $P_j \in \{0\} \cup [1, n R_{\max}]$ and $S_j \in \{0\} \cup [1, \gamma\, V_{\max}]$. In this case, noting that $r_j \geq 1$ and $v_j \geq 1$ for all $j = 1,\ldots,n$, if $\lambda_j\, (P_j + r_j\, v_j\, x_j)/(v_0 + S_j + v_j\, x_j)$ is nonzero, then it satisfies $\lambda_j\, (P_j + r_j\, v_j\, x_j)/(v_0 + S_j + v_j\, x_j) \geq \lambda_{\min}/((2+\gamma)\, V_{\max})$. That is, the quantity $\lambda_j\, (P_j + r_j\, v_j\, x_j)/(v_0 + S_j + v_j\, x_j)$ is either zero or is at least $\lambda_{\min}/((2+\gamma)\, V_{\max})$. This observation implies that if the expected revenue that we want to generate from customer types $\{j,\ldots,n\}$ is above zero but below $\lambda_{\min}/((2+\gamma)\, V_{\max})$, then we can increase the expected revenue that we want to generate up to $\lambda_{\min}/((2+\gamma)\, V_{\max})$, since offering any of the products to any of

the customer types immediately yields an expected revenue of at least $\lambda_{\min}/((2+\gamma)\,V_{\max})$. In this case, we can assume that $R_j \in \{0\} \cup [\lambda_{\min}/((2+\gamma)\,V_{\max}), R_{\max}]$. On the right side of (4), we need $\Theta_{j+1}(P_{j+1}, S_{j+1}, R_{j+1})$ for the values of $(P_{j+1}, S_{j+1}, R_{j+1}) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$. So, noting the bounds for the different components of the state variable (P_j, S_j, R_j), we need to store the value of $\Theta_j(P_j, S_j, S_j)$ only for $(P_j, S_j, R_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\,1, \lceil nR_{\max} \rceil\,]$, $S_j \in \{0\} \cup [\,1, \lceil \gamma\, V_{\max} \rceil\,]$ and $R_j \in \{0\} \cup [\,\lfloor \lambda_{\min}/((2+\gamma)\,V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$.

4 Estimating the Optimal Expected Revenue

Noting that z^* is the optimal expected revenue in problem (1), in this section, we show that we can use the approximate dynamic program in (4) to obtain an estimate $\tilde{z}$ of the optimal expected revenue that satisfies $\tilde{z} \geq (1-\epsilon)\, z^*$ for an appropriate choice of $\epsilon \in (0,1)$. In the next lemma, we give elementary properties of the value functions computed through the dynamic program in (4). The first part of the lemma shows that if we decrease P_j, increase S_j or increase R_j, then $\Theta_j(P_j, S_j, R_j)$ increases. The second part of the lemma shows how we can counterbalance a decrease in P_j and an increase in S_j by a decrease in R_j to ensure that the value function $\Theta_j(P_j, S_j, R_j)$ does not increase. In particular, if we decrease P_j by at most a factor of $1+\rho$ and increase S_j by at most a factor of $1+\rho$, then we can decrease R_j by at least a factor of $(1+\rho)^2$ to ensure that $\Theta_j(P_j, S_j, R_j)$ does not increase. We defer the proof of this lemma to Appendix C.

Lemma 3 (a) *Assume that* (P_j, S_j, R_j) *and* $(\hat{P}_j, \hat{S}_j, \hat{R}_j)$ *satisfy* $\hat{P}_j \leq P_j$, $\hat{S}_j \geq S_j$ *and* $\hat{R}_j \geq R_j$. *Then, we have* $\Theta_j(\hat{P}_j\,, \hat{S}_j, \hat{R}_j) \geq \Theta_j(P_j\,, S_j, R_j)$ *for all* $j = 1, \ldots, n$.

(b) *Assume that* (P_j, S_j, R_j) *and* $(\hat{P}_j, \hat{S}_j, \hat{R}_j)$ *satisfy* $\hat{P}_j \geq P_j/(1+\rho)$, $\hat{S}_j \leq (1+\rho)\, S_j$ *and* $\hat{R}_j \leq R_j/(1+\rho)^2$. *Then, we have* $\Theta_j(\hat{P}_j\,, \hat{S}_j, \hat{R}_j) \leq \Theta_j(P_j\,, S_j, R_j)$ *for all* $j = 1, \ldots, n$.

In the next proposition, we build on Lemma 3 to show that we can decrease the third component of the state variable (P_j, S_j, R_j) by a factor of $(1+\rho)^{3\,(n-j)}$ to obtain a lower bound on the value function $V_j(\cdot,\cdot,\cdot)$ by using the value function $\Theta_j(\cdot,\cdot,\cdot)$. This proposition is the critical ingredient in the development of our FPTAS and it ultimately allows us to obtain an estimate $\tilde{z}$ of the optimal expected revenue that satisfies $\tilde{z} \geq z^*/(1+\rho)^{3n}$.

Proposition 4 *For all* $j = 1, \ldots, n$, *we have* $V_j(P_j, S_j, R_j) \geq \Theta_j(P_j, S_j, R_j\,/\,(1+\rho)^{3(n-j)})$.

Proof. We show the result by using induction over the decision epochs in reverse order. For the decision epoch $n+1$, by the boundary condition of the dynamic program in (2), we have $V_{n+1}(\cdot,\cdot,R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $V_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$ if $R_{n+1} > 0$. Similarly, by the boundary condition of the dynamic program in (4), we have $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$ if $R_{n+1} > 0$. We have $R_{n+1} \leq 0$ if and only if $(1+\rho)^3 R_{n+1} \leq 0$. So, we obtain $V_{n+1}(\cdot,\cdot,R_{n+1}) = \Theta_{n+1}(\cdot,\cdot,(1+\rho)^3 R_{n+1})$ and the result holds for decision epoch $n+1$. Assuming that the result holds for decision epoch $j+1$, we show that the result holds for decision epoch j. For notational brevity, we let $\beta = 1/(1+\rho)^{3(n-j)}$. We fix an arbitrary value for the state variable (P_j, S_j, R_j) and $x_j \in \{0,1\}$. For the fixed values of (P_j, S_j, R_j) and $x_j \in \{0,1\}$, we let $P_{j+1} = P_j + r_j\, v_j\, x_j$, $S_{j+1} = S_j + v_j\, x_j$, $R_{j+1} = R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}$ and $R^{\beta}_{j+1} = \beta\, R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}$. By the induction assumption, we have

$$V_{j+1}(P_{j+1}, S_{j+1}, R_{j+1}) \geq \Theta_{j+1}(P_{j+1}, S_{j+1}, R_{j+1}/(1+\rho)^{3(n-j-1)}). \tag{5}$$

Furthermore, we observe that we can lower bound $R_{j+1}/(1+\rho)^{3(n-j-1)}$ by $(1+\rho)^3 R^{\beta}_{j+1}$, which follows from the chain of inequalities

$$\begin{aligned}
\frac{1}{(1+\rho)^{3(n-j-1)}} R_{j+1} = (1+\rho)^3\, \beta\, R_{j+1} &= (1+\rho)^3\, \beta \left\{ R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \right\} \\
&\geq (1+\rho)^3 \left\{ \beta\, R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \right\} = (1+\rho)^3\, R^{\beta}_{j+1}.
\end{aligned}$$

Using the first part of Lemma 3 for decision epoch $j+1$, $\Theta_{j+1}(\cdot,\cdot,R_{j+1})$ is increasing in R_{j+1}, in which case, noting the inequality above, we obtain

$$\Theta_{j+1}(P_{j+1}, S_{j+1}, R_{j+1}/(1+\rho)^{3(n-j-1)}) \geq \Theta_{j+1}(P_{j+1}, S_{j+1}, (1+\rho)^3\, R^{\beta}_{j+1}). \tag{6}$$

Since $\lfloor P_{j+1} \rfloor \geq P_{j+1}/(1+\rho)$, $\lceil S_{j+1} \rceil \leq (1+\rho)\, S_{j+1}$ and $(1+\rho)\, R^{\beta}_{j+1} = (1+\rho)^3\, R^{\beta}_{j+1}/(1+\rho)^2$ by the definition of $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$, using the second part of Lemma 3 for decision epoch $j+1$, we get

$$\Theta_{j+1}(P_{j+1}, S_{j+1}, (1+\rho)^3\, R^{\beta}_{j+1}) \geq \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, (1+\rho)\, R^{\beta}_{j+1}). \tag{7}$$

For the moment, assume that $R^{\beta}_{j+1} > 0$. Since $(1+\rho)\, R^{\beta}_{j+1} \geq \lceil R^{\beta}_{j+1} \rceil$ and $\Theta_{j+1}(\cdot,\cdot,R_{j+1})$ is increasing in R_{j+1} by the first part of Lemma 3, we get $\Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, (1+\rho)\, R^{\beta}_{j+1}) \geq$

$\Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R^{\beta}_{j+1} \rceil)$. Next, assume that $R^{\beta}_{j+1} \leq 0$. The third component of the state variable in the dynamic program in (4) can only decrease as the decision epochs progress and it is the sign of this component that affects the boundary condition. Thus, if $\hat{R}_{j+1}$ and $\tilde{R}_{j+1}$ both satisfy $\hat{R}_{j+1} \leq 0$ and $\tilde{R}_{j+1} \leq 0$, then we have $\Theta_{j+1}(\cdot, \cdot, \hat{R}_{j+1}) = 0 = \Theta_{j+1}(\cdot, \cdot, \tilde{R}_{j+1})$. So, if $(1+\rho)\, R^{\beta}_{j+1} \leq 0$, then we have $\lceil R^{\beta}_{j+1} \rceil \leq 0$ as well and we obtain $\Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, (1+\rho)\, R^{\beta}_{j+1}) = 0 = \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R^{\beta}_{j+1} \rceil)$. Thus, under both assumptions, we obtain

$$\Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, (1+\rho)\, R^{\beta}_{j+1}) \geq \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R^{\beta}_{j+1} \rceil). \tag{8}$$

Noting the definitions of P_{j+1}, S_{j+1}, R_{j+1} and R^{β}_{j+1} and combining the inequalities in (5), (6), (7) and (8), it follows that

$$\begin{aligned} &V_{j+1}\Big(P_j + r_j\, v_j\, x_j\,,\, S_j + v_j\, x_j\,,\, R_j - \lambda_j \tfrac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}\Big) = V_{j+1}(P_{j+1}, S_{j+1}, R_{j+1}) \\ &\quad \geq \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R^{\beta}_{j+1} \rceil) = \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor\,, \lceil S_j + v_j\, x_j \rceil\,, \lceil \beta\, R_j - \lambda_j \tfrac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big). \end{aligned}$$

Since our choice of $x_j \in \{0,1\}$ is arbitrary, the inequality above holds for any $x_j \in \{0,1\}$. Adding $c_j\, x_j$ to both sides of the inequality above and taking the minimum over $x_j \in \{0,1\}$, we get

$$\begin{aligned} V_j(P_j\,, S_j, R_j) &= \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + V_{j+1}\Big(P_j + r_j\, v_j\, x_j\,,\, S_j + v_j\, x_j\,,\, R_j - \lambda_j \tfrac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}\Big) \right\} \\ &\geq \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor\,, \lceil S_j + v_j\, x_j \rceil\,, \lceil \beta\, R_j - \lambda_j \tfrac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big) \right\} = \Theta_j(P_j, S_j, \beta\, R_j), \end{aligned}$$

where the two equalities above use (2) and (4). Since $\beta = 1/(1+\rho)^{3(n-j)}$, the chain of inequalities above yields $V_j(P_j\,, S_j, R_j) \geq \Theta_j(P_j, S_j, R_j/(1+\rho)^{3(n-j)})$, which is the desired result. □

In the next corollary, we build on Proposition 4 to show that we can use $\{\Theta_j(\cdot,\cdot,\cdot) : j = 1, \ldots, n\}$ to obtain an estimate $\tilde{z}$ of the optimal expected revenue that satisfies $\tilde{z} \geq z^*/(1+\rho)^{3n}$.

Corollary 5 *Let* $\tilde{z} = \max\{z \in \mathsf{Dom}^{\rho} : \Theta_1(0,0,z) \leq C,\ z \in [\,\lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]\}$ *and* z^* *be the optimal objective value of problem* (1)*. Then, we have* $\tilde{z} \geq z^*/(1+\rho)^{3n}$.

Proof. First, assume that $\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor \geq \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor$. It is simple to check that the optimal objective value of problem (1) satisfies $z^* \leq R_{\max}$ so that $\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor \leq z^* \leq \lceil R_{\max} \rceil$. Note that z^* is also given by the optimal objective value of problem (3). Thus,

we have $V_1(0,0,z^*) \leq C$. So, it follows that $C \geq V_1(0,0,z^*) \geq \Theta_1(0,0,z^*/(1+\rho)^{3(n-1)}) \geq \Theta_1(0,0,\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor)$, where the second inequality is by Proposition 4 and the third inequality is by the first part of Lemma 3. This discussion shows that $\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor$ is a feasible solution to the problem in the corollary providing an objective value of $\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor$. Since $\tilde{z}$ is the optimal objective value of this problem, we get $\tilde{z} \geq \lfloor z^*/(1+\rho)^{3(n-1)} \rfloor \geq z^*/(1+\rho)^{3n}$, where the second inequality is by the fact that $\lfloor (1+\rho)\, x \rfloor \geq x$ for $x \in \Re_+$. Second, assume that $\lfloor z^*/(1+\rho)^{3(n-1)} \rfloor < \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor$. Since $\tilde{z}$ is a feasible solution to the problem in the corollary, we have $\tilde{z} \geq \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor > \lfloor z^*/(1+\rho)^{3(n-1)} \rfloor \geq z^*/(1+\rho)^{3n}$. $\square$

Corollary 5 does not yet yield a solution to problem (1) with a performance guarantee, since due to the round up and down operators in the dynamic program in (4), the estimate $\tilde{z}$ of the optimal expected revenue may not correspond to the expected revenue from a solution.

5 Fully Polynomial Time Approximation Scheme

In this section, we show that we can follow the optimal state and action trajectory in the dynamic program in (4) to obtain a feasible solution to problem (1). Also, the expected revenue from this solution is no worse than $\tilde{z}$ given in Corollary 5. Using these results, we give our FPTAS. The next algorithm follows the optimal state and action trajectory in the dynamic program in (4).

Step 0. Using the dynamic program in (4), for all $j = 1, \ldots, n$, compute $\Theta_j(P_j, S_j, R_j)$ for all $(P_j, S_j, R_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\, 1, \lceil n R_{\max} \rceil\,]$, $S_j \in \{0\} \cup [1, \lceil \gamma\, V_{\max} \rceil\,]$ and $R_j \in \{0\} \cup [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$. Set $j = 1$, $\tilde{P}_1 = 0$, $\tilde{S}_1 = 0$ and $\tilde{R}_1 = \max\{z \in \mathsf{Dom}^\rho : \Theta_1(0,0,z) \leq C,\ z \in [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]\}$.

Step 1. Given the values of $\tilde{P}_j$, $\tilde{S}_j$ and $\tilde{R}_j$, to decide whether to offer product j, let $\tilde{x}_j$ be an optimal solution to the problem

$$\min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor \tilde{P}_j + r_j\, v_j\, x_j \rfloor\, , \lceil \tilde{S}_j + v_j\, x_j \rceil\, , \lceil \tilde{R}_j - \lambda_j\, \tfrac{\tilde{P}_j + r_j\, v_j\, x_j}{v_0 + \tilde{S}_j + v_j\, x_j} \rceil \Big) \right\}. \tag{9}$$

Set $\tilde{P}_{j+1} = \lfloor \tilde{P}_j + r_j\, v_j\, \tilde{x}_j \rfloor$, $\tilde{S}_{j+1} = \lceil \tilde{S}_j + v_j\, \tilde{x}_j \rceil$ and $\tilde{R}_{j+1} = \lceil \tilde{R}_j - \lambda_j\, \frac{\tilde{P}_j + r_j\, v_j\, \tilde{x}_j}{v_0 + \tilde{S}_j + v_j\, \tilde{x}_j} \rceil$.

Step 2. Increase j by 1. If $j \leq n$, then go to Step 1. Otherwise, stop and return $(\tilde{x}_1, \ldots, \tilde{x}_n)$.

Since the results of the round up and down operators depend on the value of ρ, the output of the algorithm above depends on the value of ρ as well and we refer to this algorithm

as the APPRX(ρ) algorithm, where APPRX stands for approximation and ρ emphasizes its dependence on ρ. The APPRX(ρ) algorithm follows the optimal state and action trajectory in the dynamic program in (4) starting from the initial state $(\tilde{P}_1, \tilde{S}_1, \tilde{R}_1)$, where $\tilde{P}_1 = 0$, $\tilde{S}_1 = 0$ and $\tilde{R}_1 = \max\{z \in \mathsf{Dom}^\rho : \Theta_1(0,0,z) \leq C,\ z \in [\,\lfloor \lambda_{\min}/((2+\gamma)\,V_{\max})\rfloor, \lceil R_{\max}\rceil\,]\}$. The product offer decisions from the algorithm are $(\tilde{x}_1, \ldots, \tilde{x}_n)$. Our goal is to show that these product offer decisions are feasible to problem (1) and they provide a performance guarantee. In the next lemma, we show that the product offer decisions from the APPRX(ρ) algorithm are feasible to problem (1).

Lemma 6 *Assume that $(\tilde{x}_1, \ldots, \tilde{x}_n)$ are generated by the* APPRX(ρ) *algorithm. Then, we have $\sum_{j=1}^n c_j\,\tilde{x}_j \leq C$.*

Proof. We have $\tilde{R}_1 = \max\{z \in \mathsf{Dom}^\rho : \Theta_1(0,0,z) \leq C,\ z \in [\,\lfloor \lambda_{\min}/((2+\gamma)\,V_{\max})\rfloor, \lceil R_{\max}\rceil\,]\}$ in Step 0 of the APPRX(ρ) algorithm, which implies that $\Theta_1(0,0,\tilde{R}_1) \leq C$. In this case, since $\tilde{P}_1 = 0$ and $\tilde{S}_1 = 0$ in the APPRX(ρ) algorithm, we obtain $\Theta_1(\tilde{P}_1, \tilde{S}_1, \tilde{R}_1) \leq C$. Noting the definitions of $\tilde{P}_{j+1}$, $\tilde{S}_{j+1}$ and $\tilde{R}_{j+1}$ in Step 1 of the APPRX(ρ) algorithm and using the fact that $\tilde{x}_j$ is the optimal solution to problem (9), if we compare problems (4) and (9), then it follows that $\Theta_j(\tilde{P}_j, \tilde{S}_j, \tilde{R}_j) = c_j\,\tilde{x}_j + \Theta_{j+1}(\tilde{P}_{j+1}, \tilde{S}_{j+1}, \tilde{R}_{j+1})$ for all $j = 1, \ldots, n$. Adding these equalities over all $j = 1, \ldots, n$, we obtain $\Theta_1(\tilde{R}_1, \tilde{S}_1, \tilde{R}_1) = \sum_{j=1}^n c_j\,\tilde{x}_j + \Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1})$. Thus, noting that $\Theta_1(\tilde{P}_1, \tilde{S}_1, \tilde{R}_1) \leq C$, we have $\sum_{j=1}^n c_j\,\tilde{x}_j + \Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) \leq C$. By the boundary condition of the dynamic program in (4), $\Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1})$ is either zero or infinity. If $\Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) = \infty$, then we get a contradiction to the inequality $\sum_{j=1}^n c_j\,\tilde{x}_j + \Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) \leq C$. Therefore, we must have $\Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) = 0$. In this case, since we have $\sum_{j=1}^n c_j\,\tilde{x}_j + \Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) \leq C$, we get $\sum_{j=1}^n c_j\,\tilde{x}_j \leq C$. □

In the next lemma, we show that the product offer decisions from the APPRX(ρ) algorithm provide a performance guarantee for problem (1). A large part of the proof of this lemma shows that the expected revenue from the product offer decisions generated by the APPRX(ρ) algorithm is no worse than the optimal expected revenue estimate $\tilde{R}_1 = \max\{z \in \mathsf{Dom}^\rho : \Theta_1(0,0,z) \leq C,\ z \in [\,\lfloor \lambda_{\min}/((2+\gamma)\,V_{\max})\rfloor, \lceil R_{\max}\rceil\,]\}$.

Lemma 7 *Let* REV *be the expected revenue from the product offer decisions generated by the* APPRX(ρ) *algorithm and z^* be the optimal objective value of problem* (1). *Then, we have* REV $\geq z^*/(1+\rho)^{3n}$.

Proof. Assume that $(\tilde{P}_1, \ldots, \tilde{P}_{n+1})$, $(\tilde{S}_1, \ldots, \tilde{S}_{n+1})$ and $(\tilde{R}_1, \ldots, \tilde{R}_{n+1})$ are generated by the APPRX(ρ) algorithm. Noting the definition of $\tilde{R}_1$ in Step 0 of APPRX(ρ) algorithm, by Corollary 5, we have $\tilde{R}_1 \geq z^*/(1+\rho)^{3n}$. We let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be the product offer decisions generated by the APPRX(ρ) algorithm. By the expected revenue expression in (1), we have REV $= \sum_{j=1}^{n} \lambda_j \frac{\sum_{k=1}^{j} r_k\, v_k\, \tilde{x}_k}{v_0 + \sum_{k=1}^{j} v_k\, \tilde{x}_k}$. In the rest of the proof, we show that REV $\geq \tilde{R}_1$, in which case, the desired result follows by the fact that REV $\geq \tilde{R}_1 \geq z^*/(1+\rho)^{3n}$. Since $\tilde{P}_{k+1} = \lfloor \tilde{P}_k + r_k\, v_k\, \tilde{x}_k \rfloor$ in Step 1 of the APPRX(ρ) algorithm, we have $\tilde{P}_{k+1} \leq \tilde{P}_k + r_k\, v_k\, \tilde{x}_k$. Adding this inequality over all $k = 1, \ldots, j-1$ and noting that $\tilde{P}_1 = 0$, we get $\tilde{P}_j \leq \sum_{k=1}^{j-1} r_k\, v_k\, \tilde{x}_k$. A similar argument yields $\tilde{S}_j \geq \sum_{k=1}^{j-1} v_k\, \tilde{x}_k$ as well. Since $\tilde{R}_{j+1} = \lceil \tilde{R}_j - \lambda_j \frac{\tilde{P}_j + r_j\, v_j\, \tilde{x}_j}{v_0 + \tilde{S}_j + v_j\, \tilde{x}_j} \rceil$ in Step 1 of the APPRX(ρ) algorithm, we get

$$\tilde{R}_{j+1} \geq \tilde{R}_j - \lambda_j \frac{\tilde{P}_j + r_j\, v_j\, \tilde{x}_j}{v_0 + \tilde{S}_j + v_j\, \tilde{x}_j} \geq \tilde{R}_j - \lambda_j \frac{\sum_{k=1}^{j} r_k\, v_k\, \tilde{x}_k}{v_0 + \sum_{k=1}^{j} v_k\, \tilde{x}_k},$$

where the second inequality is by the fact that $\tilde{P}_j \leq \sum_{k=1}^{j-1} r_k\, v_k\, \tilde{x}_k$ and $\tilde{S}_j \geq \sum_{k=1}^{j-1} v_k\, \tilde{x}_k$ for all $j = 1, \ldots, n$. Adding the chain of inequalities above over all $j = 1, \ldots, n$, we get $\tilde{R}_{n+1} \geq \tilde{R}_1 - \sum_{j=1}^{n} \lambda_j \frac{\sum_{k=1}^{j} r_k\, v_k\, \tilde{x}_k}{v_0 + \sum_{k=1}^{j} v_k\, \tilde{x}_k} = \tilde{R}_1 -$ REV. By using the same argument at the end of the proof of Lemma 6, we have $\Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) = 0$, but $\Theta_{n+1}(\tilde{P}_{n+1}, \tilde{S}_{n+1}, \tilde{R}_{n+1}) = 0$ if and only if $\tilde{R}_{n+1} \leq 0$. Thus, we must have $\tilde{R}_{n+1} \leq 0$. In this case, we obtain $0 \geq \tilde{R}_{n+1} \geq \tilde{R}_1 -$ REV. □

Next, we focus on the number of operations required to run the APPRX(ρ) algorithm. The number of operations required to compute the value functions $\{\Theta_j(\cdot,\cdot,\cdot) : j = 1, \ldots, n\}$ in Step 0 of the APPRX(ρ) algorithm dominates the number of operations required to run the other steps. In Step 0 of the APPRX(ρ) algorithm, we need to compute $\Theta_j(P_j, S_j, R_j)$ for all $j = 1, \ldots, n$ and $(P_j, S_j, R_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\,1, \lceil n R_{\max} \rceil\,]$, $S_j \in \{0\} \cup [\,1, \lceil \gamma\, V_{\max} \rceil\,]$ and $R_j \in \{0\} \cup [\,\lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$. By the definition of Dom^ρ, there are

$$O\left(\frac{\log(n R_{\max})}{\log(1+\rho)} \times \frac{\log(\gamma\, V_{\max})}{\log(1+\rho)} \times \frac{\log(\frac{(1+\gamma)\, R_{\max}\, V_{\max}}{\lambda_{\min}})}{\log(1+\rho)}\right) = O\left(\frac{\log(n R_{\max}) \log(\gamma\, V_{\max}) \log(\frac{(1+\gamma)\, R_{\max}\, V_{\max}}{\lambda_{\min}})}{\rho^3}\right)$$

possible values of $(P_j, S_j, R_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\,1, \lceil n R_{\max} \rceil\,]$, $S_j \in \{0\} \cup [\,1, \lceil \gamma\, V_{\max} \rceil\,]$ and $R_j \in \{0\} \cup [\,\lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$. Thus, we need to compute $\Theta_j(P_j, S_j, R_j)$ for $O(\log(n R_{\max}) \log(\gamma\, V_{\max}) \log((1+\gamma)\, R_{\max}\, V_{\max}/\lambda_{\min})/\rho^3)$ different values of (P_j, S_j, R_j). We compute the value functions $\{\Theta_j(\cdot,\cdot,\cdot) : j = 1, \ldots, n\}$ by moving over the decision epochs in reverse order in the dynamic program in (4) and computing $\Theta_j(P_j, S_j R_j)$ for a certain value of (P_j, S_j, R_j) takes $O(1)$ operations. So, since there are n decision epochs, we can run

Step 0 of the APPRX(ρ) algorithm in $O(n \log(nR_{\max}) \log(\gamma V_{\max}) \log((1+\gamma) R_{\max} V_{\max}/\lambda_{\min})/\rho^3)$ operations. In the next theorem, we use this observation to give an FPTAS for problem (1).

Theorem 8 *Letting z^* be the optimal objective value of problem (1), there exists an algorithm such that for any $\epsilon \in (0,1)$, the algorithm runs in $O(n^4 \log(nR_{\max}) \log(nV_{\max}) \log(nR_{\max} V_{\max}/\lambda_{\min})/\epsilon^3)$ operations and generates product offer decisions that are feasible to problem (1), providing an expected revenue of at least $(1-\epsilon) z^*$.*

Proof. We run the APPRX(ρ) algorithm with $\rho = \epsilon/(6n)$ and let REV be the expected revenue from the product offer decisions generated by this algorithm. By Lemma 6, the product offer decisions are feasible to problem (1). By Lemma 7, we get $z^* \leq (1+\epsilon/(6n))^{3n}$ REV. Since $(1+x/n)^n \leq \exp(x) \leq (1+2x)$ for $x \in (0, 1/2)$, we get $(1+\epsilon/(6n))^{3n} \leq \exp(\epsilon/2) \leq 1+\epsilon$ for $\epsilon \in (0,1)$. In this case, we obtain REV $\geq z^*/(1+\epsilon/(6n))^{3n} \geq z^*/(1+\epsilon) \geq (1-\epsilon) z^*$. So, the expected revenue from the product offer decisions is at least $(1-\epsilon) z^*$. If we set $\rho = \epsilon/(6n)$, then we have $n\rho \leq 1/6$ so that $\gamma = (\exp(n\rho) - 1)/\rho \leq (1 + 2n\rho - 1)/\rho = 2n$ by the inequalities at the beginning of the proof. By the discussion right before the theorem, we can run the APPRX(ρ) algorithm in $O(n \log(nR_{\max}) \log(\gamma V_{\max}) \log((1+\gamma) R_{\max} V_{\max}/\lambda_{\min})/\rho^3)$ operations. Replacing ρ with $\epsilon/6n$ and γ with the upper bound of $2n$ in the last expression yields the desired running time. □

If we do not have a limit on the total space consumption of the offered products, then we can solve the dynamic program in (4) more efficiently, in which case, the running time of our FPTAS reduces to $O(n^3 \log(nR_{\max}) \log(nV_{\max}) / \epsilon^2)$ operations. In Appendix D, we analyze the running time of our FPTAS for the uncapacitated version of the assortment problem.

6 Numerical Study

In our numerical study, we test the performance of our FPTAS on a large number of problem instances. We generate each problem instance as follows. We generate the revenue r_j of product j from the uniform distribution over $[1, \bar{R}]$, where $\bar{R}$ is a parameter that we vary. We reindex $(r_1, \ldots, r_n)$ such that $r_1 \leq \ldots \leq r_n$. Thus, noting that the consideration set of a customer of type k is $\{1, \ldots, k\}$, a customer of type k is interested in the k products with the lowest prices. To come up with the arrival probability λ_k of customer type k, we generate β_j from the uniform distribution over $[0,1]$ for all $j = 1, \ldots, n$ and set $\lambda_k = \beta_k / \sum_{j=1}^n \beta_j$. The preference weight of the no purchase

option is $v_0 = 10$. We use two approaches to generate the preference weights of the products. In the first approach, we simply generate the preference weight v_j of product j from the uniform distribution over $[1, 50]$. In the second approach, we generate the preference weight v_j of product j from the uniform distribution over $[1, 50]$ and reindex $(v_1, \ldots, v_n)$ such that $v_1 \geq \ldots \geq v_n$. In the first approach, referred to as NR, the preference weight of a product does not have any relationship with the price of the product. In the second approach, referred to as PD, the products with higher prices have lower preference weights. In all of our problem instances, the number of products is $n = 36$ and there is no limit on the total space consumption of the offered products.

Using PR $\in \{\text{NR}, \text{PD}\}$ to denote the approach used to generate the preference weights, we vary $(\bar{R}, \text{PR})$ over $\{5, 25, 50\} \times \{\text{NR}, \text{PD}\}$, yielding six combinations. In each combination, we generate 100 problem instances by using the approach above. For each problem instance, we run the $\textsc{Apprx}(\rho)$ algorithm to get a solution providing at least 1/2, 3/4 or 9/10 of the optimal expected revenue. By the proof of Theorem 8, we can set $\rho = \epsilon/(6n)$ to get a solution providing at least a $1 - \epsilon$ fraction of the optimal expected revenue. In Appendix E, we also give a dynamic program to get an upper bound on the optimal expected revenue. We compare the expected revenues from the solutions obtained by the $\textsc{Apprx}(\rho)$ algorithm with the optimal expected revenue upper bounds.

We give our numerical results in Table 1. On the left side of the table, we show the combination $(\bar{R}, \text{PR})$. In the rest of the table, there are three blocks of five columns. The three blocks focus on the cases where we use the performance guarantees of 1/2, 3/4 and 9/10. The first, second, third and fourth columns in each block respectively show the average, maximum, 90th percentile and 75th percentile of the percent gaps between the upper bound on the optimal expected revenue and the expected revenue from the solution obtained by the $\textsc{Apprx}(\rho)$ algorithm, where these summary statistics are computed over the 100 problem instances in a particular combination of $(\bar{R}, \text{PR})$. So, for problem instance ℓ, letting $\textsc{UpBn}^\ell$ be the upper bound on the optimal expected revenue and $\textsc{Rev}^\ell$ be the expected revenue from the solution obtained by the $\textsc{Apprx}(\rho)$ algorithm, the first, second, third and fourth columns respectively show the average, maximum, 90th percentile and 75th percentile of the data $\{100 \times \frac{\textsc{UpBn}^\ell - \textsc{Rev}^\ell}{\textsc{UpBn}^\ell} : \ell = 1, \ldots, 100\}$. The fifth column shows the average CPU seconds for the $\textsc{Apprx}(\rho)$ algorithm over the 100 problem instances.

The results in Table 1 indicate that the practical performance of the $\textsc{Apprx}(\rho)$ algorithm can be quite strong. The maximum optimality gap is 3.89% when we use a performance guarantee of 1/2 and 2.16% when we use a performance guarantee of 9/10. The average CPU second is 5.28 with

Combin.	Perf. Guar. of $1-\epsilon = 1/2$					Perf. Guar. of $1-\epsilon = 3/4$					Perf. Guar. of $1-\epsilon = 9/10$				
$(\bar{R}, \text{PR})$	% Gap with Upp. Bnd				CPU	% Gap with Upp. Bnd				CPU	% Gap with Upp. Bnd				CPU
	Avg.	Max.	90th	75th	Sec.	Avg.	Max.	90th	75th	Sec.	Avg.	Max.	90th	75th	Sec.
$(5, \text{NR})$	1.86	2.91	2.42	2.13	4.47	1.40	1.86	1.65	1.55	19.91	1.25	1.74	1.45	1.37	152.22
$(5, \text{PD})$	2.14	3.89	3.16	2.61	5.01	1.52	2.83	1.89	1.70	22.27	1.32	1.84	1.55	1.47	170.50
$(25, \text{NR})$	1.55	2.82	2.13	1.80	5.11	1.12	1.78	1.42	1.27	23.12	1.01	1.50	1.24	1.11	179.53
$(25, \text{PD})$	1.65	3.23	2.43	2.04	5.68	1.22	2.10	1.55	1.43	25.39	1.14	1.63	1.47	1.26	196.59
$(50, \text{NR})$	1.50	2.65	2.18	1.72	5.44	1.08	1.71	1.37	1.26	24.78	0.97	1.42	1.20	1.06	198.34
$(50, \text{PD})$	1.56	3.60	2.29	1.82	5.99	1.15	1.84	1.53	1.34	27.21	1.08	2.16	1.42	1.18	217.57

Table 1: Performance of the APPRX(ρ) algorithm with different performance guarantees.

a performance guarantee of 1/2 and 185.79 with a performance guarantee of 9/10. We report only the average CPU seconds, but the CPU seconds vary among the problem instances in a particular $(\bar{R}, \text{PR})$ combination by no more than 30%. If have 72 products, then the average CPU seconds is 42.80 with a performance guarantee of 1/2 and 1575.08 with a performance guarantee 9/10, but we do not report detailed results for 72 products. As expected from the discussion after Theorem 8, doubling the number of products increases the CPU seconds by about a factor of eight.

Meissner et al. (2012) give a linear programming formulation for assortment problems under a mixture of multinomial logit models. This formulation relaxes the assortment problem by assuming that we can offer different subsets of products to different customer types, but uses the so called product cuts to tighten the relaxation. This formulation also provides an upper bound on the optimal expected revenue. In Appendix F, we give a detailed comparison of our approach with the linear programming formulation of Meissner et al. (2012). Our numerical results indicate that our approach compares quite favorably with the linear programming formulation. To give a feel for our numerical results, the largest percent gap between our upper bound on the optimal expected revenue and the expected revenue from the solution obtained by our approach is 2.36%. The same percent gap for the linear programming formulation can be as high as 19.07%.

7 Conclusions and Future Research

One research direction is to work with other consideration set structures. In our FPTAS, customers of different types associate the same preference weight with a particular product, as long as this product is in their consideration sets. Also, they have nested consideration sets. In Appendix G, we show that if customers of different types associate the same preference weight with a particular product, but they have arbitrary consideration sets, then the assortment problem is NP-hard to approximate within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$. To show this result, we build on Desir and

Goyal (2014) and Aouad, Farias, Levi and Segev (2015), but these papers do not immediately imply our result. Desir and Goyal (2014) use a mixture of multinomial logit models, but customers of two different types may associate different preference weights with a particular product when this product is in the consideration sets of both customer types. Aouad, Farias, Levi and Segev (2015) use a choice model where customers of a certain type rank the products according to a certain preference order and purchase their most preferred available product. The complexity result in Appendix G indicates that the case where customers of different types associate the same preference weight with a particular product but they have arbitrary consideration sets is unlikely to provide adequate structure to give an FPTAS for the corresponding assortment problem.

To impose some structure on the consideration sets, we can work with the so called interval consideration sets of the form $\{k, \ldots, j\}$ with $k \leq j$. Aouad, Levi and Segev (2015) study dynamic assortment problems where multiple customers are served with limited inventories. In their choice model, each customer of a particular type ranks the products according to a preference order of the form $(k, \ldots, j)$ with $k \leq j$, where product i is preferred over product $i+1$. She purchases her most preferred available product. Letting n be the number of products and p be the ratio between the largest and smallest product revenues, the authors group the customer types into $O(\log n)$ or $O(\log \log p)$ groups. The dynamic assortment problem that focuses on each group ends up being tractable to approximate. Picking the best of the solutions from the dynamic assortment problems for different groups, the authors get a solution whose expected revenue deviates from the optimal by at most a factor of $O(\log n)$ or $O(\log \log p)$. In Appendix H, we show that if we have interval consideration sets but no limit on the space consumption, then we can use the idea in Aouad, Levi and Segev (2015) to group the customer types into $O(\log n)$ or $O(\log \log p)$ groups, in which case, we can approximate the assortment problem that focuses on each group by using nested consideration sets. So, we can use our FPTAS to approximate the assortment problem for each group. Running our FPTAS $O(n)$ or $O(\log p)$ times, we get a solution whose expected revenue deviates from the optimal by at most a factor of $O(\log n)$ or $O(\log \log p)$. This approach does not provide an expected revenue arbitrarily close to the optimal. Thus, giving an FPTAS for interval consideration sets is one research direction. Another one is to address the capacitated problem.

Lastly, the running time of our FPTAS depends on $R_{\max}$, $V_{\max}$ and $\lambda_{\min}$. An interesting question is whether one can come up with a strongly polynomial time algorithm, whose running time does not depend on these quantities.

Acknowledgements. We thank the associate editor and two anonymous referees whose comments improved our paper substantially and unified our results.

References

Aouad, A., Farias, V. and Levi, R. (2016), Assortment optimization under consider-then-choose choice models, Technical report, MIT, Massachusetts, MA.

Aouad, A., Farias, V., Levi, R. and Segev, D. (2015), The approximability of assortment optimization under ranking preferences, Technical report, MIT, Cambridge, MA.

Aouad, A., Levi, R. and Segev, D. (2015), Approximation algorithms for dynamic assortment optimization models, Technical report, MIT, Massachusetts, MA.

Bront, J. J. M., Diaz, I. M. and Vulcano, G. (2009), 'A column generation algorithm for choice-based network revenue management', *Oper. Res.* **57**(3), 769–784.

Chong, J.-K., Ho, T.-H. and Tang, C. S. (2001), 'A modeling framework for category assortment planning', *M&SOM* **3**(3), 191–210.

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

Desir, A. and Goyal, V. (2014), Near-optimal algorithms for capacity constrained assortment optimization, Technical report, Columbia University, New York, NY.

Gallego, G., Iyengar, G., Phillips, R. and Dubey, A. (2004), Managing flexible products on a network, Technical report, Columbia University, New York, NY.

Gallego, G., Li, A., Truong, V.-A. and Wang, X. (2016), Online personalized resource allocation with customer choice, Technical report, Columbia University, New York, NY.

Garey, M. and Johnson, D. (1979), *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, NY.

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

Hastad, J. (1996), Clique is hard to approximate within $n^{1-\epsilon}$, *in* '*Proceedings of the 37th Annual Symposium on Foundations of Computer Science*', Burlington, VT, pp. 627–636.

Jagabathula, S. and Rusmevichientong, P. (2015), A nonparametric joint assortment and price choice model, Technical report, New York University, New York, NY.

Jagabathula, S. and Vulcano, G. (2015), A model to estimate individual preferences using panel data, Technical report, New York University, New York, NY.

Kok, A. G. and Fisher, M. L. (2007), 'Demand estimation and assortment optimization under substitution: Methodology and application', *Oper. Res.* **55**(6), 1001–10021.

Kunnumkal, S. and Talluri, K. (2014), On the tractability of the piecewise-linear approximation for general discrete-choice network revenue management, Technical report, Universitat Pompeu Fabra, Barcelona, Spain.

McFadden, D. and Train, K. (2000), 'Mixed MNL models for discrete response', *Journal of Applied Economics* **15**, 447–470.

Meissner, J., Strauss, A. and Talluri, K. (2012), 'An enhanced concave program relaxation for choice network revenue management', *Production and Operations Management* **22**(1), 71–87.

Misra, K. (2008), Understanding retail assortments in competitive markets, Technical report, Northwestern University, Evanston, IL.

Rusmevichientong, P., Shmoys, D. B., Tong, C. and Topaloglu, H. (2014), 'Assortment optimization under the multinomial logit model with random choice parameters', *POM* **23**(11), 2023–2039.

Sahin, O. and Wang, R. (2014), The impact of consumer search cost on assortment planning and pricing, Technical report, Johns Hopkins University, Baltimore, MD.

Stauss, A. K. and Talluri, K. (2016), Tractable consideration set structures for network revenue management, Technical report, Imperial College, London, UK.

Talluri, K. (2011), A randomized concave programming method for choice network revenue management, Technical report, Universitat Pompeu Fabra, Barcelona, Spain.

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

Vulcano, G., van Ryzin, G. and Chaar, W. (2010), 'OM practice – Choice-based revenue management: An empirical study of estimation and optimization', *M&SOM* **12**(3), 371–392.

A Appendix: Proof of Theorem 1

We use a reduction from the partition problem. In the partition problem, we are given n items indexed by $\{1,\ldots,n\}$. There is an integer weight w_j associated with item j and we have $\sum_{j=1}^{n} w_j = 2L$ for some integer L. The partition problem asks whether there exists a vector $x \in \{0,1\}^n$ such that $\sum_{j=1}^{n} w_j\, x_j = L$. Using the instance of the partition problem, we define an instance of the assortment feasibility problem as follows. There are $n+1$ products. The revenues of the products are $r_j = 2$ for all $j = 1,\ldots,n$ and $r_{n+1} = 7$. The preference weights of the products are $v_j = w_j$ for all $j = 1,\ldots,n$ and $v_{n+1} = 4L$. The preference weight of the no purchase option is $v_0 = L$. The probabilities of observing different customer types are $\lambda_j = 0$ for all $j = 1,\ldots,n-1$, $\lambda_n = \lambda_{n+1} = 1/2$. The expected revenue threshold in the assortment feasibility problem is $K = 3$. We show that there exists a set of products that provides an expected revenue of $K = 3$ or more in the assortment feasibility problem if and only if there exists a vector $x \in \{0,1\}^n$ such that $\sum_{j=1}^{n} w_j\, x_j = L$.

Product $n+1$ has the largest revenue among the $n+1$ products. Thus, it is simple to show that offering this product always increases the expected revenue from any set of products, in which case, we always offer product $n+1$ in the assortment feasibility problem and the only question for the assortment feasibility problem is to choose a set of products among the products $\{1,\ldots,n\}$ to obtain an expected revenue of $K = 3$ or more. We use the vector $x = (x_1,\ldots,x_n) = \{0,1\}^n$ to capture which of the products $\{1,\ldots,n\}$ we offer, where $x_j = 1$ if and only if product j is offered. Since we always offer product $n+1$, noting the expected revenue expression in (1), as a function of $x \in \{0,1\}^n$, the expected revenue in the assortment feasibility problem is

$$\frac{1}{2}\left\{\frac{\sum_{j=1}^{n} 2\, w_j\, x_j}{L + \sum_{j=1}^{n} w_j\, x_j}\right\} + \frac{1}{2}\left\{\frac{\sum_{j=1}^{n} 2\, w_j\, x_j + 28L}{L + \sum_{j=1}^{n} w_j\, x_j + 4L}\right\} = \frac{\sum_{j=1}^{n} w_j\, x_j}{L + \sum_{j=1}^{n} w_j\, x_j} + \frac{\sum_{j=1}^{n} w_j\, x_j + 14L}{5L + \sum_{j=1}^{n} w_j\, x_j}.$$

So, letting $w(x) = \sum_{j=1}^{n} w_j\, x_j$ for notational brevity, there exists a set of products that provides an expected revenue of $K = 3$ or more if and only if there exists a vector $x \in \{0,1\}^n$ such that $\frac{w(x)}{L+w(x)} + \frac{w(x)+14L}{5L+w(x)} \geq 3$. We write this inequality as $w(x)\,(5L + w(x)) + (w(x) + 14L)\,(L + w(x)) \geq 3\,(L + w(x))\,(5L + w(x))$, which is equivalent to $w(x)^2 - 2L\,w(x) + L^2 \leq 0$. Thus, there exists a set of products that provides an expected revenue of $K = 3$ or more if and only if there exists a vector $x \in \{0,1\}^n$ such that $w(x)^2 - 2L\,w(x) + L^2 \leq 0$, but the last inequality holds if and only if $w(x) = L$. Therefore, there exists a set of products that provides an expected revenue of $K = 3$ or more if and only if there exists a vector $x \in \{0,1\}^n$ such that $w(x) = L$.

B Appendix: Proof of Lemma 2

Considering the first inequality in the lemma, we have $P_{k+1} = \lfloor P_k + r_k\, v_k\, x_k \rfloor \leq P_k + r_k\, v_k\, x_k \leq P_k + R_{\max}\, x_k$. Adding this inequality over all $k = 1, \ldots, j-1$ and noting that $P_1 = 0$, we obtain $P_j \leq R_{\max} \sum_{k=1}^{j-1} x_k$ for all $j = 1, \ldots, n$. Therefore, we have $P_j \leq R_{\max} \sum_{k=1}^{j-1} x_k \leq n R_{\max}$ and the first inequality in the lemma follows. Considering the second inequality in the lemma, we use induction over the decision epochs in forward order to show that $S_j \leq \frac{(1+\rho)^j - (1+\rho)}{\rho}\, V_{\max}$ for all $j = 1, \ldots, n$. If $j = 1$, then the expression on the right side of the last inequality is equal to zero. Thus, since $S_1 = 0$, the result holds for the first decision epoch. Assuming that the result holds for decision epoch j, we show that the result holds for decision epoch $j+1$. We have

$$\begin{aligned} S_{j+1} = \lceil S_j + v_j\, x_j \rceil &\leq (1+\rho)\, (S_j + v_j\, x_j) \\ &\leq (1+\rho) \left\{ \frac{(1+\rho)^j - (1+\rho)}{\rho}\, V_{\max} + V_{\max} \right\} = \frac{(1+\rho)^{j+1} - (1+\rho)}{\rho}\, V_{\max}, \end{aligned}$$

where the second inequality follows from the induction argument. Thus, the result holds for decision epoch $j+1$ as well, completing the induction argument. In this case, we have $S_j \leq \frac{(1+\rho)^n - (1+\rho)}{\rho}\, V_{\max}$ for all $j = 1, \ldots, n$. Noting that $1 + \rho \leq \exp(\rho)$, we have $(1+\rho)^n \leq \exp(n\rho)$, so that

$$S_j \leq \frac{(1+\rho)^n - (1+\rho)}{\rho}\, V_{\max} \leq \frac{\exp(n\rho) - (1+\rho)}{\rho}\, V_{\max} \leq \frac{\exp(n\rho) - 1}{\rho}\, V_{\max} = \gamma\, V_{\max}.$$

Thus, the second inequality in the lemma holds. Considering the third inequality in the lemma, by the discussion at the beginning of the proof, we have $P_j \leq R_{\max} \sum_{k=1}^{j-1} x_k$ for all $j = 1, \ldots, n$. Also, since $v_k \geq 1$, we have $S_{k+1} = \lceil S_k + v_k\, x_k \rceil \geq S_k + x_k$. Adding this inequality over all $k = 1, \ldots, j-1$ and noting that $S_1 = 0$, we obtain $S_j \geq \sum_{k=1}^{j-1} x_k$. In this case, we have

$$\begin{aligned} R_{k+1} = \left\lceil R_k - \lambda_k\, \frac{P_k + r_k\, v_k\, x_k}{v_0 + S_k + v_k\, x_k} \right\rceil &\geq R_k - \lambda_k\, \frac{P_k + r_k\, v_k\, x_k}{v_0 + S_k + v_k\, x_k} \\ &\geq R_k - \lambda_k\, \frac{R_{\max} \sum_{j=1}^{k-1} x_j + R_{\max}\, x_k}{v_0 + \sum_{j=1}^{k-1} x_j + x_k} \geq R_k - \lambda_k\, R_{\max}, \end{aligned}$$

where the second inequality uses the fact that $r_k\, v_k \leq R_{\max}$ and $v_k \geq 1$. Adding the inequality above over all $k = j, \ldots, n$, we obtain $R_{n+1} \geq R_j - (\lambda_j + \ldots + \lambda_n)\, R_{\max}$. Thus, the third inequality in the lemma holds.

C Appendix: Proof of Lemma 3

We show both parts by using induction over the decision epochs in reverse order. To show the first part, by the boundary condition of the dynamic program in (4), we note that $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$ if $R_{n+1} > 0$. So, $\Theta_{n+1}(\cdot,\cdot,R_{n+1})$ is increasing in R_{n+1}. Thus, if $\hat{R}_{n+1} \geq R_{n+1}$, then we have $\Theta_{n+1}(\cdot,\cdot,\hat{R}_{n+1}) \geq \Theta_{n+1}(\cdot,\cdot,R_{n+1})$ and the result holds for decision epoch $n+1$. Assuming that the result holds for decision epoch $j+1$, we show that the result holds for decision epoch j. We fix an arbitrary $x_j \in \{0,1\}$. Considering (P_j, S_j, R_j) and $(\hat{P}_j, \hat{S}_j, \hat{R}_j)$ such that $\hat{P}_j \leq P_j$, $\hat{S}_j \geq S_j$ and $\hat{R}_j \geq R_j$, for the fixed value of $x_j \in \{0,1\}$, we let $P_{j+1} = P_j + r_j\, v_j\, x_j$, $S_{j+1} = S_j + v_j\, x_j$, $R_{j+1} = R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}$, $\hat{P}_{j+1} = \hat{P}_j + r_j\, v_j\, x_j$, $\hat{S}_{j+1} = \hat{S}_j + v_j\, x_j$, $\hat{R}_{j+1} = \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j}$. Noting that $\hat{P}_j \leq P_j$, we have $\hat{P}_{j+1} = \hat{P}_j + r_j\, v_j\, x_j \leq P_j + r_j\, v_j\, x_j = P_{j+1}$. So, since $\hat{P}_{j+1} \leq P_{j+1}$, we obtain $\lfloor \hat{P}_{j+1} \rfloor \leq \lfloor P_{j+1} \rfloor$. By using a similar argument, we obtain $\lceil \hat{S}_{j+1} \rceil \geq \lceil S_{j+1} \rceil$. Also, since $\hat{P}_j \leq P_j$, $\hat{S}_j \geq S_j$ and $\hat{R}_j \geq R_j$, we obtain

$$\hat{R}_{j+1} = \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \geq R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} = R_{j+1},$$

which implies that $\lceil \hat{R}_{j+1} \rceil \geq \lceil R_{j+1} \rceil$. Since we have $\lfloor \hat{P}_{j+1} \rfloor \leq \lfloor P_{j+1} \rfloor$, $\lceil \hat{S}_{j+1} \rceil \geq \lceil S_{j+1} \rceil$ and $\lceil \hat{R}_{j+1} \rceil \geq \lceil R_{j+1} \rceil$, by the induction assumption, it follows that $\Theta_{j+1}(\lfloor \hat{P}_{j+1} \rfloor, \lceil \hat{S}_{j+1} \rceil, \lceil \hat{R}_{j+1} \rceil) \geq \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R_{j+1} \rceil)$. Noting the definitions of P_{j+1}, S_{j+1}, R_{j+1}, $\hat{P}_{j+1}$, $\hat{S}_{j+1}$ and $\hat{R}_{j+1}$, we write the last inequality as $\Theta_{j+1}\Big(\lfloor \hat{P}_j + r_j\, v_j\, x_j \rfloor, \lceil \hat{S}_j + v_j\, x_j \rceil, \lceil \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \rceil\Big) \geq \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor, \lceil S_j + v_j\, x_j \rceil, \lceil R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big)$. Since our choice of $x_j \in \{0,1\}$ is arbitrary, the last inequality holds for any $x_j \in \{0,1\}$. In this case, adding $c_j\, x_j$ to both sides of the last inequality and taking the minimum over $x_j \in \{0,1\}$, we have

$$\begin{aligned}
\Theta_j(\hat{P}_j, \hat{S}_j \hat{R}_j) &= \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor \hat{P}_j + r_j\, v_j\, x_j \rfloor, \lceil \hat{S}_j + v_j\, x_j \rceil, \lceil \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \rceil\Big) \right\} \\
&\geq \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor, \lceil S_j + v_j\, x_j \rceil, \lceil R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big) \right\} = \Theta_j(P_j, S_j, R_j),
\end{aligned}$$

where the two equalities above use (4). The chain of inequalities above establishes the desired result. Next, we focus on the second part of the lemma.

To show the second part of the lemma, for the moment, assume that $R_{n+1} \leq 0$. In this case, having $\hat{R}_{n+1} \leq R_{n+1}/(1+\rho)^2$ and $R_{n+1} \leq 0$ implies that $\hat{R}_{n+1} \leq 0$. Thus, by the boundary

condition of the dynamic program in (4), we obtain $\Theta_{n+1}(\cdot,\cdot,\hat{R}_{n+1}) = 0 = \Theta_{n+1}(\cdot,\cdot,R_{n+1})$. Next, assume that $R_{n+1} > 0$. In this case, by the boundary condition of the dynamic program in (4), we have $\Theta_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$, but $\Theta_{n+1}(\cdot,\cdot,\hat{R}_{n+1}) \in \{0,\infty\}$. Therefore, we have $\Theta_{n+1}(\cdot,\cdot,\hat{R}_{n+1}) \leq \Theta_{n+1}(\cdot,\cdot,R_{n+1}) = \infty$. Thus, under both assumptions, we have $\Theta_{n+1}(\cdot,\cdot,\hat{R}_{n+1}) \leq \Theta_{n+1}(\cdot,\cdot,R_{n+1})$ and the result holds for decision epoch $n+1$. Assuming that the result holds for decision epoch $j+1$, we show that the result holds for decision epoch j. We fix an arbitrary $x_j \in \{0,1\}$. Considering (P_j, S_j, R_j) and $(\hat{P}_j, \hat{S}_j, \hat{R}_j)$ such that $\hat{P}_j \geq P_j/(1+\rho)$, $\hat{S}_j \leq (1+\rho)\, S_j$ and $\hat{R}_j \leq R_j/(1+\rho)^2$, for the fixed value of x_j, we let $P_{j+1} = P_j + r_j\, v_j\, x_j$, $S_{j+1} = S_j + v_j\, x_j$, $R_{j+1} = R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j}$, $\hat{P}_{j+1} = \hat{P}_j + r_j\, v_j\, x_j$, $\hat{S}_{j+1} = \hat{S}_j + v_j\, x_j$, $\hat{R}_{j+1} = \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j}$. Since $(1+\rho)\,\hat{P}_j \geq P_j$, we have $(1+\rho)\,\hat{P}_{j+1} = (1+\rho)\,(\hat{P}_j + r_j\, v_j\, x_j) \geq P_j + r_j\, v_j\, x_j = P_{j+1} \geq \lfloor P_{j+1} \rfloor$, yielding $\hat{P}_{j+1} \geq \lfloor P_{j+1} \rfloor/(1+\rho)$. Note that $\lfloor P_{j+1} \rfloor/(1+\rho) \in \mathsf{Dom}^\rho$. Thus, having $\hat{P}_{j+1} \geq \lfloor P_{j+1} \rfloor/(1+\rho)$ implies that $\lfloor \hat{P}_{j+1} \rfloor \geq \lfloor P_{j+1} \rfloor/(1+\rho)$. By using a similar argument, we can show that $\lceil \hat{S}_{j+1} \rceil \leq (1+\rho)\, \lceil S_{j+1} \rceil$. Also, since $\hat{P}_j \geq P_j/(1+\rho)$, $\hat{S}_j \leq (1+\rho)\, S_j$ and $\hat{R}_j \leq R_j/(1+\rho)^2$, we have

$$R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \geq (1+\rho)^2 \hat{R}_j - \lambda_j \frac{(1+\rho)\,\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \frac{\hat{S}_j}{1+\rho} + v_j\, x_j} \geq (1+\rho)^2 \left\{ \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \right\}.$$

Noting the definitions of R_{j+1} and $\hat{R}_{j+1}$, the inequality above yields $\hat{R}_{j+1} \leq R_{j+1}/(1+\rho)^2$. So, we obtain $\hat{R}_{j+1} \leq R_{j+1}/(1+\rho)^2 \leq \lceil R_{j+1} \rceil/(1+\rho)^2$. Since $\lceil R_{j+1} \rceil/(1+\rho)^2 \in \mathsf{Dom}^\rho$, having $\hat{R}_{j+1} \leq \lceil R_{j+1} \rceil/(1+\rho)^2$ implies that $\lceil \hat{R}_{j+1} \rceil \leq \lceil R_{j+1} \rceil/(1+\rho)^2$. By the discussion in this paragraph, we have $(1+\rho)\,\lfloor \hat{P}_{j+1} \rfloor \geq \lfloor P_{j+1} \rfloor$, $\lceil \hat{S}_{j+1} \rceil \leq (1+\rho)\, \lceil S_{j+1} \rceil$ and $\lceil \hat{R}_{j+1} \rceil \leq \lceil R_{j+1} \rceil/(1+\rho)^2$. In this case, the induction assumption implies that $\Theta_{j+1}(\lfloor \hat{P}_{j+1} \rfloor, \lceil \hat{S}_{j+1} \rceil, \lceil \hat{R}_{j+1} \rceil) \leq \Theta_{j+1}(\lfloor P_{j+1} \rfloor, \lceil S_{j+1} \rceil, \lceil R_{j+1} \rceil)$ and we write the last inequality as $\Theta_{j+1}\Big(\lfloor \hat{P}_j + r_j\, v_j\, x_j \rfloor, \lceil \hat{S}_j + v_j\, x_j \rceil, \lceil \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \rceil\Big) \leq \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor, \lceil S_j + v_j\, x_j \rceil, \lceil R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big)$. Noting that our choice of $x_j \in \{0,1\}$ is arbitrary, the last inequality holds for any $x_j \in \{0,1\}$. Adding $c_j\, x_j$ to both sides of the last inequality and taking the minimum over $x_j \in \{0,1\}$, we obtain

$$\begin{aligned}
\Theta_j(\hat{P}_j, \hat{S}_j \hat{R}_j) &= \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor \hat{P}_j + r_j\, v_j\, x_j \rfloor, \lceil \hat{S}_j + v_j\, x_j \rceil, \lceil \hat{R}_j - \lambda_j \frac{\hat{P}_j + r_j\, v_j\, x_j}{v_0 + \hat{S}_j + v_j\, x_j} \rceil\Big) \right\} \\
&\leq \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j\, x_j \rfloor, \lceil S_j + v_j\, x_j \rceil, \lceil R_j - \lambda_j \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rceil\Big) \right\} = \Theta_j(P_j, S_j, R_j),
\end{aligned}$$

which is the desired result.

D Appendix: Uncapacitated Assortment Problem

In this section, we consider the case where there is no limit on the total space consumption of the offered products. If there is no limit on the total space consumption of the offered products, then we can reduce the number of operations required for the APPRX(ρ) algorithm. This reduction in the number of operations for the APPRX(ρ) algorithm results in a more efficient FPTAS. In particular, if there is no limit on the total space consumption of the offered products, then we can assume that $c_j = 0$ for all $j = 1, \ldots, n$, so that the space consumptions of all of the products are zero. By the boundary condition of the dynamic program in (4), $\Theta_{n+1}(\cdot, \cdot, \cdot)$ takes the value zero or infinity. In this case, the critical observation is that if $c_j = 0$ for all $j = 1, \ldots, n$, then noting the dynamic program in (4), $\Theta_j(\cdot, \cdot, \cdot)$ takes the value zero or infinity for all $j = 1, \ldots, n$. In Step 0 of the APPRX(ρ) algorithm, we need to compute $\Theta_j(P_j, S_j, R_j)$ for the values of (P_j, S_j, R_j) in $\mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$. It turns out that for the values of R_j in Dom^ρ, we can give an efficient characterization of when $\Theta_j(P_j, S_j, R_j)$ takes the value zero or infinity. To give this efficient characterization, we compute the value functions $\{Z_j(\cdot, \cdot) : j = 1, \ldots, n\}$ through the dynamic program

$$Z_j(P_j, S_j) = \max_{x_j \in \{0,1\}} \left\{ \left\lfloor \lambda_j \frac{P_j + r_j \, v_j \, x_j}{v_0 + S_j + v_j \, x_j} + Z_{j+1}(\lfloor P_j + r_j \, v_j \, x_j \rfloor, \lceil S_j + v_j \, x_j \rceil) \right\rfloor \right\}, \tag{10}$$

with the boundary condition that $Z_{n+1}(\cdot, \cdot) = 0$. In the next lemma, we focus on the values of R_j in Dom^ρ. We show that if $R_j \leq Z_j(P_j, S_j)$, then we have $\Theta_j(P_j, S_j, R_j) = 0$, whereas if $R_j > Z_j(P_j, S_j)$, then we have $\Theta_j(P_j, S_j, R_j) = \infty$.

Lemma 9 *Assume that $c_j = 0$ for all $j = 1, \ldots, n$ and the value functions $\{\Theta_j(\cdot, \cdot, \cdot) : j = 1, \ldots, n\}$ and $\{Z_j(\cdot, \cdot) : j = 1, \ldots, n\}$ are computed through the dynamic programs in (4) and (10). For all $j = 1, \ldots, n$, if $R_j \in \mathsf{Dom}^\rho$, then we have*

$$\Theta_j(P_j, S_j, R_j) = \begin{cases} 0 & \text{if } R_j \leq Z_j(P_j, S_j) \\ \infty & \text{if } R_j > Z_j(P_j, S_j). \end{cases}$$

Proof. We show the result by using induction over the decision epochs in reverse order. By the boundary condition of the dynamic program in (10), we have $Z_{n+1}(\cdot, \cdot) = 0$. By the boundary condition of the dynamic program in (4), we have $\Theta_{n+1}(P_{n+1}, S_{n+1}, R_{n+1}) = 0$ if $R_{n+1} \leq 0 =$

$Z_{n+1}(P_{n+1}, S_{n+1})$, whereas we have $\Theta_{n+1}(P_{n+1}, S_{n+1}, R_{n+1}) = \infty$ if $R_{n+1} > 0 = Z_{n+1}(P_{n+1}, S_{n+1})$. So, the result holds for decision epoch $n+1$. Assuming that the result holds for decision epoch $j+1$, we show that the result holds for decision epoch j. Noting that $c_j = 0$, we write (4) as

$$\Theta_j(P_j, S_j, R_j) = \min \left\{ \Theta_{j+1}\Big(\lfloor P_j \rfloor , \lceil S_j \rceil , \lceil R_j - \lambda_j \tfrac{P_j}{v_0 + S_j} \rceil \Big), \right.$$
$$\left. \Theta_{j+1}\Big(\lfloor P_j + r_j\, v_j \rfloor , \lceil S_j + v_j \rceil , \lceil R_j - \lambda_j \tfrac{P_j + r_j\, v_j}{v_0 + S_j + v_j} \rceil \Big) \right\}. \quad (11)$$

Therefore, $\Theta_j(P_j, S_j, R_j)$ takes the value infinity if and only if both expressions inside the minimum operator above take the value infinity. Considering the first expression inside the minimum operator above, by the induction assumption, we have $\Theta_{j+1}\Big(\lfloor P_j \rfloor , \lceil S_j \rceil , \lceil R_j - \lambda_j \tfrac{P_j}{v_0 + S_j} \rceil \Big) = \infty$ if and only if $\lceil R_j - \lambda_j \tfrac{P_j}{v_0 + S_j} \rceil > Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil)$. A straightforward lemma, given as Lemma 10 right after the proof, shows that for $a, b \in \mathsf{Dom}^\rho$ and $\alpha \in \Re$, we have $\lceil a - \alpha \rceil > b$ if and only if $a > \lfloor b + \alpha \rfloor$. We have $R_j \in \mathsf{Dom}^\rho$. Noting the outer round down operator in the dynamic program in (10), we have $Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil) \in \mathsf{Dom}^\rho$ as well. In this case, by Lemma 10, we have $\lceil R_j - \lambda_j \tfrac{P_j}{v_0 + S_j} \rceil > Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil)$ if and only of $R_j > \Big\lfloor \lambda_j \tfrac{P_j}{v_0 + S_j} + Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil) \Big\rfloor$. In this case, it follows that the first expression inside the minimum operator in (11) takes the value infinity if and only if $R_j > \Big\lfloor \lambda_j \tfrac{P_j}{v_0 + S_j} + Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil) \Big\rfloor$. Considering the second expression inside the minimum operator in (11) and following a similar argument, it follows that the second expression inside the minimum operator in (11) takes the value infinity if and only if $R_j > \Big\lfloor \lambda_j \tfrac{P_j + r_j\, v_j}{v_0 + S_j + v_j} + Z_{j+1}(\lfloor P_j + r_j\, v_j \rfloor , \lceil S_j + v_j \rceil) \Big\rfloor$. Thus, letting

$$\zeta_j(P_j, S_j) = \max \left\{ \left\lfloor \lambda_j \frac{P_j}{v_0 + S_j} + Z_{j+1}(\lfloor P_j \rfloor , \lceil S_j \rceil) \right\rfloor , \right.$$
$$\left. \left\lfloor \lambda_j \frac{P_j + r_j\, v_j}{v_0 + S_j + v_j} + Z_{j+1}(\lfloor P_j + r_j\, v_j \rfloor , \lceil S_j + v_j \rceil) \right\rfloor \right\},$$

we have $\Theta_j(P_j, S_j, R_j) = \infty$ if and only if $R_j > \zeta_j(P_j, S_j)$. The maximization problem above is identical to the maximization problem in (10). In this case, we get $\zeta_j(P_j, S_j) = Z_j(P_j, S_j)$ and it follows that $\Theta_j(P_j, S_j, R_j) = \infty$ if and only if $R_j > Z_j(P_j, S_j)$. □

In the next lemma, we give an elementary property of the round up and down operators. This property is used in the proof of Lemma 9. The proof uses the fact that if $x \leq y$ and $y \in \mathsf{Dom}^\rho$, then we must have $\lceil x \rceil \leq y$.

Lemma 10 *For $a, b \in \mathsf{Dom}^\rho$ and $\alpha \in \Re$, we have $\lceil a - \alpha \rceil > b$ if and only if $a > \lfloor b + \alpha \rfloor$.*

Proof. First, we show that if $\lceil a-\alpha \rceil > b$, then we have $a > \lfloor b+\alpha \rfloor$. To get a contradiction, assume that $\lceil a-\alpha \rceil > b$ and $a \leq \lfloor b+\alpha \rfloor$. By the last inequality, we have $a \leq \lfloor b+\alpha \rfloor \leq b+\alpha$, which implies that $a - \alpha \leq b$. Since $b \in \mathsf{Dom}^\rho$, having $a - \alpha \leq b$ implies that $\lceil a - \alpha \rceil \leq b$, which contradicts the fact that $\lceil a - \alpha \rceil > b$. Second, we show that if $a > \lfloor b + \alpha \rfloor$, then we have $\lceil a - \alpha \rceil > b$. To get a contradiction, assume that $a > \lfloor b + \alpha \rfloor$ and $\lceil a - \alpha \rceil \leq b$. By the last inequality, we have $a - \alpha \leq \lceil a - \alpha \rceil \leq b$, which implies that $a \leq b + \alpha$. Since $a \in \mathsf{Dom}^\rho$, having $a \leq b + \alpha$ implies that $a \leq \lfloor b + \alpha \rfloor$, which contradicts the fact that $a > \lfloor b + \alpha \rfloor$. □

We recall that in Step 0 of the Apprx(ρ) algorithm, we need to compute $\Theta_j(P_j, S_j, R_j)$ for all $j = 1, \ldots, n$ and $(P_j, S_j, R_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\, 1, \lceil n R_{\max} \rceil \,]$, $S_j \in \{0\} \cup [\, 1, \lceil \gamma\, V_{\max} \rceil \,]$ and $R_j \in \{0\} \cup [\, \lfloor \lambda_{\min}/((2 + \gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil \,]$. By Lemma 9, if $c_j = 0$ for all $j = 1, \ldots, n$, then it is enough to compute $Z_j(P_j, S_j)$ for all $j = 1, \ldots, n$ and $(P_j, S_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\, 1, \lceil n R_{\max} \rceil \,]$ and $S_j \in \{0\} \cup [\, 1, \lceil \gamma\, V_{\max} \rceil \,]$. In this case, by comparing $Z_j(P_j, S_j)$ with R_j, we can come up with the value of $\Theta_j(P_j, S_j, R_j)$, as long as $R_j \in \mathsf{Dom}^\rho$. Noting the definition of Dom^ρ, there are

$$O\left(\frac{\log(n R_{\max})}{\log(1+\rho)} \times \frac{\log(\gamma\, V_{\max})}{\log(1+\rho)}\right) = O\left(\frac{\log(n R_{\max}) \log(\gamma\, V_{\max})}{\rho^2}\right)$$

possible values of $(P_j, S_j) \in \mathsf{Dom}^\rho \times \mathsf{Dom}^\rho$ such that $P_j \in \{0\} \cup [\, 1, \lceil n R_{\max} \rceil \,]$ and $S_j \in \{0\} \cup [\, 1, \lceil \gamma\, V_{\max} \rceil \,]$. Thus, we need to compute $Z_j(P_j, S_j)$ for $O(\log(n R_{\max}) \log(\gamma\, V_{\max})/\rho^2)$ different values of (P_j, S_j), in which case, since there are n decision epochs, we can run Step 0 of the Apprx(ρ) algorithm in $O(n \log(n R_{\max}) \log(\gamma\, V_{\max})/\rho^2)$ operations. The number of operations in other steps are dominated by the number of operations in Step 0. In the next theorem, we build on this computation time analysis to provide an FPTAS for problem (1) when there is no limit on the total space consumption of the offered products.

Theorem 11 *Assume that $c_j = 0$ for all $= 1, \ldots, n$. Letting z^* be the optimal objective value of problem (1), there exists an algorithm such that for any $\epsilon \in (0, 1)$, the algorithm runs in $O(n^3 \log(n R_{\max}) \log(n V_{\max})/\epsilon^2)$ operations and generates product offer decisions that provide an expected revenue of at least $(1 - \epsilon)\, z^*$.*

Proof. Noting the proof of Theorem 8, if we run the APPRX(ρ) algorithm with $\rho = \epsilon/(6n)$, then the resulting product offer decisions provide an expected revenue of at least $(1-\epsilon)\, z^*$. By the discussion right before the theorem, the APPRX(ρ) algorithm runs in $O(n \log(nR_{\max}) \log(\gamma\, V_{\max})/\rho^2)$ operations when we have $c_j = 0$ for all $j = 1, \ldots, n$. In the proof of Theorem 8, we also show that if $\rho = \epsilon/(6n)$, then we have $\gamma \leq 2n$. Replacing ρ with $\epsilon/(6n)$ and γ with the upper bound of $2n$ in the last running time expression, we obtain the desired running time. □

E Appendix: An Upper Bound on the Optimal Expected Revenue

We show how to obtain an upper bound on the optimal expected revenue, which is used in our numerical study. Since our numerical study focuses on the case where there is no limit on the total space consumption of the offered products, our upper bound also focuses on the same case. In the second half of this section, we discuss the case where there is a limit on the total space consumption. Our upper bound is based on solving a dynamic program similar to the one in (10). In particular, we compute the value functions $\{Y_j(\cdot,\cdot) : j = 1, \ldots, n\}$ through the dynamic program

$$Y_j(P_j, S_j) = \max_{x_j \in \{0,1\}} \left\{ \lambda_j \, \frac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} + Y_{j+1}(\lceil P_j + r_j\, v_j\, x_j \rceil\, , \lfloor S_j + v_j\, x_j \rfloor) \right\}, \tag{12}$$

with the boundary condition that $Y_{n+1}(\cdot,\cdot) = 0$. In the next lemma, we show that if there is no limit on the total space consumption of the offered products, then we can use the value functions $\{Y_j(\cdot,\cdot) : j = 1, \ldots, n\}$ to obtain an upper bound on the optimal expected revenue.

Lemma 12 *Assume that $c_j = 0$ for all $j = 1, \ldots, n$ and the value functions $\{Y_j(\cdot,\cdot) : j = 1, \ldots, n\}$ are computed through the dynamic program in* (12)*. Letting z^* be the optimal objective value of problem* (1)*, we have $Y_1(0,0) \geq z^*$.*

Proof. We use $x^* = (x_1^*, \ldots, x_n^*)$ to denote an optimal solution to problem (1) so that $z^* = \sum_{j=1}^n \lambda_j \frac{\sum_{k=1}^j r_k\, v_k\, x_k^*}{v_0 + \sum_{k=1}^j v_k\, x_k^*}$. We define $(\tilde{P}_1, \ldots, \tilde{P}_n)$ such that $\tilde{P}_1 = 0$ and $\tilde{P}_{j+1} = \lceil \tilde{P}_j + r_j\, v_j\, x_j^* \rceil$. In this case, we have $\tilde{P}_{j+1} \geq \tilde{P}_j + r_j\, v_j\, x_j^*$. Adding this inequality over all $j = 1, \ldots, k-1$ and noting that $\tilde{P}_1 = 0$, we obtain $\tilde{P}_k \geq \sum_{j=1}^{k-1} r_j\, v_j\, x_j^*$, where we follow the convention that a sum over an empty set is zero. Similarly, we define $(\tilde{S}_1, \ldots, \tilde{S}_n)$ such that $\tilde{S}_1 = 0$ and $\tilde{S}_{j+1} = \lfloor \tilde{S}_j + v_j\, x_j^* \rfloor$. By using the same argument, it follows that $\tilde{S}_k \leq \sum_{j=1}^{k-1} v_j\, x_j^*$. We observe that if we solve problem (12) with $P_j = \tilde{P}_j$ and $S_j = \tilde{S}_j$, then x_j^* is a feasible, but not necessarily an optimal solution. Therefore,

noting that the optimal objective value of problem (12) with $P_j = \tilde{P}_j$ and $S_j = \tilde{S}_j$ is given by $Y_j(\tilde{P}_j, \tilde{S}_j)$, we obtain

$$Y_j(\tilde{P}_j, \tilde{S}_j) \geq \lambda_j \frac{\tilde{P}_j + r_j\, v_j\, x_j^*}{v_0 + \tilde{S}_j + v_j\, x_j^*} + Y_{j+1}(\lceil \tilde{P}_j + r_j\, v_j\, x_j^* \rceil\, , \lfloor \tilde{S}_j + v_j\, x_j^* \rfloor)$$
$$\geq \lambda_j \frac{\sum_{k=1}^{j} r_k\, v_k\, x_k^*}{v_0 + \sum_{k=1}^{j} v_k\, x_k^*} + Y_{j+1}(\tilde{P}_{j+1}\, , \tilde{S}_{j+1}),$$

where the second inequality uses the fact that $\tilde{P}_j \geq \sum_{k=1}^{j-1} r_k\, v_k\, x_k^*$, $\tilde{S}_j \leq \sum_{k=1}^{j-1} v_k\, x_k^*$, $\tilde{P}_{j+1} = \lceil \tilde{P}_j + r_j\, v_j\, x_j^* \rceil$ and $\tilde{S}_{j+1} = \lfloor \tilde{S}_j + v_j\, x_j^* \rfloor$. Adding the inequality above over all $j = 1, \ldots, n$ and noting that $\tilde{P}_1 = 0$, $\tilde{S}_1 = 0$ and $Y_{n+1}(\cdot, \cdot) = 0$, we get $Y_1(0,0) \geq \sum_{j=1}^{n} \lambda_j \frac{\sum_{k=1}^{j} r_k\, v_k\, x_k^*}{v_0 + \sum_{k=1}^{j} v_k\, x_k^*} = z^*$. □

If there is a limit on the total space consumption, then we can use a dynamic program similar to the one in (4) to get an upper bound on the optimal expected revenue. We compute the value functions $\{\Psi_j(\cdot, \cdot, \cdot) : j = 1, \ldots, n\}$ through the dynamic program

$$\Psi_j(P_j\, , S_j, R_j) = \min_{x_j \in \{0,1\}} \left\{ c_j\, x_j + \Psi_{j+1}\Big(\lceil P_j + r_j\, v_j\, x_j \rceil\, , \lfloor S_j + v_j\, x_j \rfloor\, , \lfloor R_j - \lambda_j \tfrac{P_j + r_j\, v_j\, x_j}{v_0 + S_j + v_j\, x_j} \rfloor \Big) \right\}, \quad (13)$$

with the boundary condition that $\Psi_{n+1}(\cdot, \cdot, R_{n+1}) = 0$ if $R_{n+1} \leq 0$ and $\Psi_{n+1}(\cdot, \cdot, R_{n+1}) = \infty$ if $R_{n+1} > 0$. By using induction over the decision epochs in reverse order, we can show that $\Psi_j(P_j, S_j, R_j) \leq V_j(P_j, S_j, R_j)$ for all $j = 1, \ldots, n$. Also, similar to the first part of Lemma 3, we can show that $\Psi_j(P_j, S_j, R_j)$ is increasing in R_j. In this case, letting $\tilde{z} = \max\{z \in \mathsf{Dom}^\rho : \Psi_1(0,0,z) \leq C, \;\; z \in [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]\}$, $(1+\rho)\, \tilde{z}$ is an upper bound on the optimal expected revenue. To see this result, by (3), we have $C \geq V_1(0,0,z^*) \geq \Psi_1(0,0,z^*) \geq \Psi_1(0,0,\lfloor z^* \rfloor)$, where the second inequality uses the fact that $\Psi_j(P_j, S_j, R_j) \leq V_j(P_j, S_j, R_j)$ for all $j = 1, \ldots, n$ and the third inequality uses the fact that $\Psi_j(P_j, S_j, R_j)$ is decreasing in R_j. It is also simple to check that the optimal expected revenue satisfies $z^* \in [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$ so that $\lfloor z^* \rfloor \in [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]$. So, $\lfloor z^* \rfloor$ is a feasible solution to the problem $\max\{z \in \mathsf{Dom}^\rho : \Psi_1(0,0,z) \leq C, \;\; z \in [\, \lfloor \lambda_{\min}/((2+\gamma)\, V_{\max}) \rfloor, \lceil R_{\max} \rceil\,]\}$ providing an objective value of $\lfloor z^* \rfloor$, which implies that the optimal objective value $\tilde{z}$ of this problem is at least $\lfloor z^* \rfloor$. Thus, we have $(1+\rho)\, \tilde{z} \geq (1+\rho)\, \lfloor z^* \rfloor \geq z^*$.

When computing an upper bound on the optimal expected revenue in our numerical study in Section 6, we use the value of $\rho = \frac{1}{10}/(6n)$, which provides a performance guarantee of 9/10 in the

Apprx(ρ) algorithm. We emphasize that there is no practical need to bound the relative gap of the upper bound, since we already have an upper bound with a bounded relative gap with the optimal expected revenue. In particular, using Rev to denote the expected revenue from the product offer decisions generated by our FPTAS, by Theorem 8, we have $\text{Rev}/(1-\epsilon) \geq z^* \geq \text{Rev}$, which implies that $\text{Rev}/(1-\epsilon)$ already provides an upper bound on the optimal expected revenue with a relative gap of $1-\epsilon$. Also, we note that we use the dynamic program in (13) to obtain an upper bound on the optimal expected revenue but not to obtain a solution with a performance guarantee. Our approach for obtaining a solution with a performance guarantee is based on obtaining a lower bound on the optimal expected revenue with a certain relative gap and constructing a feasible solution that provides an expected revenue that is at least as large as the lower bound.

F Appendix: Numerical Study on Product Cuts

Meissner et al. (2012) give a linear programming formulation for assortment problems under a mixture of multinomial logit models. This formulation relaxes the assortment problem by assuming that we can offer different subsets of products to different customer types, but uses the so called product cuts to tighten the relaxation. In particular, the decision variables in the linear programming formulation correspond to the frequencies with which we offer different subsets of products to different customer types. The linear programming formulation assumes that we can offer different subsets of products to different customer types with different frequencies, but the so called m-product cuts ensure that any group of m products that are in the consideration sets of two customer types are offered to the two customer types with equal frequencies.

There are two potential sources of difficulty associated with the linear programming formulation of Meissner et al. (2012), when we use this linear programming formulation for our assortment problem. First, if there are ℓ products that are common to the consideration sets of two customer types, then there are $\binom{\ell}{m}$ different m-product cuts for these two customer types. Therefore, it is difficult to generate all of the product cuts when there are many products that are common to the consideration sets of two customer types. In our assortment problem, there are $n-1$ products that are common to the consideration sets of customer types $n-1$ and n. Thus, there are $\binom{n-1}{m}$ different m-product cuts for customers types $n-1$ and n, which implies that the number of product cuts for customer types $n-1$ and n is $\sum_{m=1}^{n-1}\binom{n-1}{m} = 2^{n-1}-1$. To get around this difficulty, we focus on only 1-product and 2-product cuts. Section 4.1.2 in Meissner et al. (2012) discusses similar difficulties

associated with the linear programming formulation when the number of products that are common to two consideration sets is large. Also, the numerical study in Meissner et al. (2012) indicates that using only 1-product and 2-product cuts is quite effective. Second, the decision variables in the linear programming formulation correspond to the frequency with which we offer different subsets of products to different customer types. In our assortment problem, we have k products in the consideration set of customers of type k, which implies that there are 2^k different subsets of products that we can offer to customers of type k. Therefore, we have $\sum_{k=1}^{n} 2^k = 2^{n+1} - 2$ decision variables in the linear programming formulation. We can try to deal with the large number of decision variables by using column generation, but the column generation subproblem is intractable when there are product cuts. To get around this difficulty, we focus on test problems with a relatively small number of products. In our numerical study, the number of products is $n = 18$.

We compare the quality of the solutions obtained by our approach and the linear programming formulation with comparable computational burden. The total running time for our approach is the sum of the running time for the $\textsc{Apprx}(\rho)$ algorithm to obtain a solution and the running time for the approach in Appendix E to obtain an upper bound on the optimal expected revenue. We solve the linear programming formulation with 1-product cuts. If the running time for the linear programming formulation is less than the total running time for our approach, then we find the 2-product cuts that are violated by the optimal solution to the linear programming formulation, add these 2-product cuts to the linear programming formulation and solve it. We continue adding the violated 2-product cuts until the total running time for the linear programming formulation reaches the total running time for our approach. In all of our problem instances, we stop adding 2-product cuts due to the running time limit before we run out of 2-product cuts to add.

Our numerical setup follows the one in Section 6. In particular, we vary $(\bar{R}, \mathrm{PR})$ over $\{5, 25, 50\} \times \{\mathrm{NR}, \mathrm{PD}\}$ to obtain six combinations. In each combination, we generate 100 problem instances by using the approach that we describe at the beginning of Section 6. The only difference is that we have $n = 18$ products, instead of $n = 36$, in our problem instances. For each problem instance ℓ, we compute four quantities. First, we use the $\textsc{Apprx}(\rho)$ algorithm to obtain a solution with a performance guarantee of 9/10. We use $\textsc{Rev}^{\ell}_{\textsc{Apprx}}$ to denote the expected revenue from the solution obtained by the $\textsc{Apprx}(\rho)$ algorithm for problem instance ℓ. Second, we use the approach in Appendix E to obtain an upper bound on the optimal expected revenue. We use $\textsc{UpBn}^{\ell}_{\textsc{Apprx}}$ to denote the upper bound on the optimal expected revenue for problem instance ℓ. Third, the optimal

Combin.	Apprx(ρ)				Product Cuts			
$(\bar{R}, \text{PR})$	Avg.	Max.	90th	75th	Avg.	Max.	90th	75th
$(5, \text{NR})$	1.19	1.86	1.51	1.40	3.68	14.22	8.34	5.55
$(5, \text{PD})$	1.27	2.36	1.64	1.50	6.13	19.07	12.36	8.72
$(25, \text{NR})$	1.07	1.65	1.47	1.26	3.36	15.15	8.33	5.15
$(25, \text{PD})$	1.19	2.03	1.59	1.35	4.84	18.21	12.02	7.40
$(50, \text{NR})$	0.99	1.63	1.41	1.19	3.26	16.50	8.96	4.82
$(50, \text{PD})$	1.09	1.84	1.41	1.26	4.98	18.63	12.6	8.14

Table 2: Optimality gaps obtained by the Apprx(ρ) algorithm and the product cuts.

objective value of the linear programming formulation of Meissner et al. (2012) provides an upper bound on the optimal expected revenue. We use $\textsc{UpBn}^{\ell}_{\text{LP}}$ to denote this upper bound for problem instance ℓ. Fourth, we check the expected revenue from each subset of products that is offered with a strictly positive frequency in the optimal solution to the linear programming formulation and pick the best subset. We use $\textsc{Rev}^{\ell}_{\text{LP}}$ to denote the expected revenue from the best subset for problem instance ℓ. Therefore, we can interpret $100 \times \frac{\textsc{UpBn}^{\ell}_{\textsc{Apprx}} - \textsc{Rev}^{\ell}_{\textsc{Apprx}}}{\textsc{UpBn}^{\ell}_{\textsc{Apprx}}}$ as the percent optimality gap that we can assess for problem instance ℓ by using our approach, whereas we can interpret $100 \times \frac{\textsc{UpBn}^{\ell}_{\text{LP}} - \textsc{Rev}^{\ell}_{\text{LP}}}{\textsc{UpBn}^{\ell}_{\text{LP}}}$ as the percent optimality gap that we can assess for problem instance ℓ by using the linear programming formulation.

We give our numerical results in Table 2. On the left side of the table, we show the combination $(\bar{R}, \text{PR})$. There are two blocks of four columns in the rest of the table. The first block focuses on our approach, whereas the second block focuses on the linear programming formulation. Considering the 100 problem instances that we generate in each combination of $(\bar{R}, \text{PR})$, the four columns in the first block respectively show the average, maximum, 90th percentile and 75th percentile of the data $\{100 \times \frac{\textsc{UpBn}^{\ell}_{\textsc{Apprx}} - \textsc{Rev}^{\ell}_{\textsc{Apprx}}}{\textsc{UpBn}^{\ell}_{\textsc{Apprx}}} : \ell = 1, \ldots, 100\}$. The four columns in the second block show the same summary statistics for the data $\{100 \times \frac{\textsc{UpBn}^{\ell}_{\text{LP}} - \textsc{Rev}^{\ell}_{\text{LP}}}{\textsc{UpBn}^{\ell}_{\text{LP}}} : \ell = 1, \ldots, 100\}$. The results in Table 2 indicate that our approach provides a significantly better understanding of the optimality gaps. By the second column in the first block, if we use the Apprx(ρ) algorithm to obtain a solution and the approach in Appendix E to obtain an upper bound, then we can conclude that the optimality gaps of the solutions that we obtain is no larger than 2.36%. By the second column in the second block, if we use the linear programming formulation, then we can only conclude that the optimality gaps of the solutions that we obtain are no larger than 19.07%.

The optimality gaps provided by our approach turn out to be significantly smaller than those provided by the linear programming formulation. A natural question is whether these smaller

Combin.	Exp Rev. from Soln.					Upper Bnds. on Opt. Exp. Rev.				
$(\bar{R}, \text{PR})$	Avg.	Max.	Min.	75th	25th	Avg.	Max.	Min.	75th	25th
$(5, \text{NR})$	2.49	10.80	-0.14	3.87	0.00	0.07	5.04	-1.86	0.87	-1.06
$(5, \text{PD})$	3.68	12.96	-0.14	6.24	0.46	1.40	8.23	-1.82	2.75	-0.35
$(25, \text{NR})$	2.18	12.38	-0.34	3.56	0.00	0.18	6.24	-1.63	0.74	-0.90
$(25, \text{PD})$	2.67	11.69	-0.29	4.00	0.00	1.16	9.43	-1.77	2.37	-0.94
$(50, \text{NR})$	2.16	14.08	-0.24	3.11	0.00	0.18	5.39	-1.49	0.82	-0.87
$(50, \text{PD})$	2.85	12.54	-0.41	5.09	0.00	1.23	9.15	-1.50	2.52	-0.67

Table 3: Expected revenues and the upper bounds obtained by the APPRX(ρ) algorithm and the product cuts.

optimality gaps are due to the fact that the expected revenues from the solutions obtained by our approach are larger or the upper bounds obtained by our approach are tighter. In Table 3, we compare the expected revenues from the solutions obtained by our approach and the linear programming formulation, as well as the upper bounds on the optimal expected revenue obtained by our approach and the linear programming formulation. On the left side of the table, we show the combination $(\bar{R}, \text{PR})$. There are two blocks of five columns in the rest of the table. The five columns in the first block show the average, maximum, minimum, 75th percentile and 25th percentile of the data $\{100 \times \frac{\text{REV}^{\ell}_{\text{APPRX}} - \text{REV}^{\ell}_{\text{LP}}}{\frac{1}{2}(\text{REV}^{\ell}_{\text{APPRX}} + \text{REV}^{\ell}_{\text{LP}})} : \ell = 1, \ldots, 100\}$, which compare the expected revenues from the solutions obtained by our approach and the linear programming formulation. The five columns in the second block show the same summary statistics for the data $\{100 \times \frac{\text{UPBN}^{\ell}_{\text{LP}} - \text{UPBN}^{\ell}_{\text{APPRX}}}{\frac{1}{2}(\text{UPBN}^{\ell}_{\text{LP}} + \text{UPBN}^{\ell}_{\text{APPRX}})} : \ell = 1, \ldots, 100\}$, which compare the upper bounds obtained by our approach and the linear programming formulation.

The results in Table 3 indicate that the expected revenues from the solutions obtained by our approach are significantly better than those from the solutions obtained by the linear programming formulation. Noting the second column in the first block, over all of our problem instances, the expected revenues from the solutions obtained by our approach improve those from the solutions obtained by the linear programming formulation by as much as 14.08%. In contrast, noting the third column in the first block, the expected revenues from the solutions obtained by the linear programming formulation can improve those from the solutions obtained by our approach by at most 0.41%. On average, the expected revenues from the solutions obtained by our approach improve those from the solutions obtained by the linear programming formulation by 2.67%. When we compare our approach and the linear programming formulation from the perspective of the upper bounds on the optimal expected revenue, our approach also provides some improvement over the linear programming formulation. On average, the upper bounds provided by our approach improve

the upper bounds provided by the linear programming formulation by 0.7%. Noting the second column in the second block, over all of our problem instances, the upper bounds provided by our approach can improve those provided by the linear programming formulation by as much as 9.43%, whereas noting the third column in the second block, the upper bounds provided by the linear programming formulation can improve those provided by our approach by at most 1.86%. Thus, the results in Table 3 indicate that our approach generally provides solutions with larger expected revenues as well as tighter upper bounds on the optimal expected revenue, when compared with the linear programming formulation with comparable computational burden.

G Appendix: Computational Complexity under Arbitrary Consideration Sets

We consider the complexity of our assortment problem when the consideration sets of different customer types are arbitrary, but customers of different types associate the same preference weight with a particular product, as long as this product is in their consideration sets. This version of the assortment problem, where the consideration sets of the different customer types are arbitrary, is substantially more difficult to solve, even without a limit on the total space consumption of the offered products. We give a brief formulation of the assortment problem with arbitrary consideration sets. There are n products indexed by $\{1,\ldots,n\}$. We use the vector $x = (x_1,\ldots,x_n) \in \{0,1\}^n$ to capture the set of products offered to customers, where $x_j = 1$ if and only if product j is offered. There are m customer types indexed by $\{1,\ldots,m\}$. A customer arriving into the system is of type k with probability λ_k. The consideration set of customers of type k is $N_k \subseteq \{1,\ldots,n\}$. The preference weight of product j is v_j and the preference weight of the no purchase option is v_0. If the set of products offered to customers is given by the vector x, then a customer of type k chooses product $j \in N_k$ with probability $P_{jk}(x) = v_j\, x_j/(v_0+\sum_{i\in N_k} v_i\, x_i)$. Thus, to maximize the expected revenue obtained from a customer, we solve the problem

$$z^* = \max_{x\in\{0,1\}^n} \left\{ \sum_{k=1}^{m} \lambda_k \left\{ \sum_{j\in N_k} r_j\, P_{jk}(x) \right\} \right\} = \max_{x\in\{0,1\}^n} \left\{ \sum_{k=1}^{m} \lambda_k \left\{ \frac{\sum_{j\in N_k} r_j\, v_j\, x_j}{v_0 + \sum_{j\in N_k} v_j\, x_j} \right\} \right\}. \tag{14}$$

Hastad (1996) shows that we cannot approximate the maximum independent set problem within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$ unless $P = NP$, where n is the number of vertices of the underlying graph in the maximum independent set problem. We give an approximation preserving reduction that reduces any maximum independent set problem to an instance of the assortment problem in

(14). Our reduction uses the work of Desir and Goyal (2014) and Aouad, Farias, Levi and Segev (2015), but we need some adjustments on their work. Desir and Goyal (2014) use a mixture of multinomial logit models, but the preference weight of a product may be n^2 or 1 when it appears in the consideration sets of different customer types. The preference weight of product j in problem (14) is fixed at v_j, when product j appears in the consideration sets of different customer types. Aouad, Farias, Levi and Segev (2015) do not use a mixture of multinomial logit models. In their work, each customer arrives with a ranking of the products in mind and chooses the most preferred available product. The maximum independent set problem is defined as follows.

Maximum Independent Set. We are given an undirected graph (N, E) with vertex set $N = \{1, \ldots, n\}$ and edge set E. We use the vector $y = (y_1, \ldots, y_n) \in \{0, 1\}^n$ to denote a set of vertices, where $y_j = 1$ if vertex j is included in the set and $y_j = 0$ if vertex j is not included in the set. A vector $y = (y_1, \ldots, y_n) \in \{0, 1\}^n$ is an independent set if $y_i + y_j \leq 1$ for all $(i, j) \in E$. The maximum independent set problem is $\max\{\sum_{j=1}^n y_j : y_i + y_j \leq 1 \ \ \forall (i, j) \in E\}$.

Therefore, a set of vertices is an independent set if any two vertices that are in the set are not connected by an edge. The maximum independent set problem finds an independent set with the largest cardinality. Given an instance of the maximum independent set problem over the undirected graph (N, E) with $N = \{1, \ldots, n\}$, we define an instance of the assortment problem in (14) as follows. There are n products and n customer types, both indexed by $N = \{1, \ldots, n\}$. The revenue of product j is $r_j = n^{2(j-1)}$. The preference weight of product j is $v_j = n^{2(n-j)}$. The preference weight of the no purchase option is $v_0 = 1$. The probability that a customer of type j arrives into the system is $\lambda_j = \alpha\, n^{-2(j-1)}$, where α is a constant that ensures that $\sum_{j=1}^n \lambda_j = 1$. The consideration set of customers of type j is given by $N_j = \{i \in N : i < j \text{ and } (i, j) \in E\} \cup \{j\}$. Thus, the consideration set of customers of type j includes each product i such that there is an edge between nodes i and j in the underlying graph for the maximum independent set problem and the index of product i is smaller than the index of product j. In addition, the consideration set of customers of type j always includes product j. We refer to this instance of the assortment problem as the induced assortment problem, emphasizing that this instance of the assortment problem is induced by the maximum independent set problem. In the next lemma, we show that we can use an independent set over the undirected graph (N, E) to construct a solution to the induced assortment problem such that the expected revenue from this solution has a certain lower bound. Throughout this section, we use $\textsc{Rev}(x)$ to denote the objective function of problem (14) as a function of $x \in \{0, 1\}^n$.

Lemma 13 *Assume that $\hat{y} \in \{0,1\}^n$ is an independent set over the undirected graph (N,E). Then, in polynomial time, we can construct a solution $\hat{x} \in \{0,1\}^n$ to the induced assortment problem that satisfies* $\textsc{Rev}(\hat{x}) \geq \alpha \sum_{j=1}^n \hat{y}_j/2$.

Proof. We let $\hat{x}_j = \hat{y}_j$ for all $j \in N$. Clearly, the solution $\hat{x} \in \{0,1\}^n$ is constructed from the independent set $\hat{y}$ in polynomial time. Consider a customer type $j \in N$. We claim that if $\hat{x}_j = 1$, then we have $\hat{x}_i = 0$ for all $i \in N_j \setminus \{j\}$. To get a contradiction, assume that $\hat{x}_j = 1$ and $\hat{x}_i = 1$ for some $i \in N_j \setminus \{j\}$. Since $i \in N_j \setminus \{j\}$, noting the definition of N_j, it follows that $(i,j) \in E$. In this case, we obtain $\hat{y}_j + \hat{y}_i = \hat{x}_j + \hat{x}_i = 2$ for some $(i,j) \in E$, which contradicts the fact that $\hat{y}$ is an independent set. Thus, the claim follows. In the objective function of problem (14), customers of type j make a contribution of $\lambda_j \sum_{i \in N_j} r_i \, v_i \, x_i / (v_0 + \sum_{i \in N_j} v_i \, x_i)$. By the claim, if $\hat{x}_j = 1$, then we have $\hat{x}_i = 0$ for all $i \in N_j \setminus \{j\}$. In this case, if $\hat{x}_j = 1$, then we have $\lambda_j \sum_{i \in N_j} r_i \, v_i \, x_i / (v_0 + \sum_{i \in N_j} v_i \, x_i) = \lambda_j \, r_j \, v_j / (v_0 + v_j)$. Therefore, we obtain

$$\begin{aligned}
\textsc{Rev}(\hat{x}) = \sum_{j=1}^n \lambda_j \left\{ \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{v_0 + \sum_{k \in N_j} v_k \, \hat{x}_k} \right\} &\geq \sum_{j=1}^n \lambda_j \, \frac{r_j \, v_j}{v_0 + v_j} \, \hat{x}_j \\
&= \sum_{j=1}^n \alpha \, n^{-2(j-1)} \, \frac{n^{2(j-1)} \, n^{2(n-j)}}{1 + n^{2(n-j)}} \, \hat{x}_j \geq \frac{\alpha}{2} \sum_{j=1}^n \hat{x}_j = \frac{\alpha}{2} \sum_{j=1}^n \hat{y}_j,
\end{aligned}$$

where the first inequality follows by ignoring the contribution from customers of type j such that $\hat{x}_j = 0$, the second inequality uses the fact that $2 \, n^{2(n-j)} \geq 1 + n^{2(n-j)}$ for all $j \in N$ and the third equality is by the fact that $\hat{x}_j = \hat{y}_j$ for all $j \in N$. □

In the next lemma, we show that we can use a solution to the induced assortment problem to construct an independent set over the undirected graph (N,E) such that the number of vertices in this independent set has a certain lower bound.

Lemma 14 *Assume that $\hat{x} \in \{0,1\}^n$ is a solution to the induced assortment problem. Then, in polynomial time, we can construct an independent set $\hat{y} \in \{0,1\}^n$ over the undirected graph (N,E) that satisfies $\sum_{j=1}^n \hat{y}_j \geq \max\{(\textsc{Rev}(\hat{x})/\alpha) - 2, 1\}$.*

Proof. Defining $M = \{j \in N : \hat{x}_j = 1 \text{ and } \hat{x}_k = 0 \ \forall \, k \in N_j \setminus \{j\}\}$, we let $\hat{y}_j = 1$ if $j \in M$ and $\hat{y}_j = 0$ if $j \notin M$. We claim that $\hat{y} \in \{0,1\}^n$ is an independent set. To get a contradiction, assume that $\hat{y}_i + \hat{y}_j = 2$ for some $(i,j) \in E$. Without loss of generality, we assume that $i < j$. Otherwise,

we can interchange the roles of i and j. Since $\hat{y}_i + \hat{y}_j = 2$, by the definition of $\hat{y}$, we must have $i \in M$ and $j \in M$. Furthermore, we have $(i,j) \in E$. Since $j \in M$, we have $\hat{x}_j = 1$ and $\hat{x}_k = 0$ for all $k \in N_j \setminus \{j\}$. Noting the definition of N_j, it follows that $\hat{x}_j = 1$ and $\hat{x}_k = 0$ for all $k \in N$ such that $k < j$ and $(k,j) \in E$. In this case, since we have $i < j$ and $(i,j) \in E$, we must have $\hat{x}_i = 0$, but having $\hat{x}_i = 0$ implies that $i \notin M$, which is a contradiction. Thus, the claim follows. Clearly, the independent set $\hat{y}$ is constructed from the solution $\hat{x}$ in polynomial time. First, we consider a customer type $j \in M$. Since $j \in M$, by the definition of M, it follows that $\hat{x}_j = 1$ and $\hat{x}_k = 0$ for all $k \in N_j \setminus \{j\}$. Therefore, we obtain the chain of inequalities

$$\lambda_j \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{v_0 + \sum_{k \in N_j} v_k \, \hat{x}_k} = \lambda_j \frac{r_j \, v_j}{v_0 + v_j} \leq \lambda_j \, r_j = \alpha \, n^{-2(j-1)} \, n^{2(j-1)} = \alpha. \tag{15}$$

Second, consider a customer type $j \notin M$. If $\hat{x}_j = 1$, then since $j \notin M$, noting the definition of M, we must have $\hat{x}_k = 1$ for some $k \in N_j \setminus \{j\}$, which implies that $\sum_{k \in N_j \setminus \{j\}} \hat{x}_k \geq 1 = \hat{x}_j$. Naturally, if $\hat{x}_j = 0$, then we have $\sum_{k \in N_j \setminus \{j\}} \hat{x}_k \geq \hat{x}_j$ trivially. Thus, we always have $\sum_{k \in N_j \setminus \{j\}} \hat{x}_k \geq \hat{x}_j$. Adding $\sum_{k \in N_j \setminus \{j\}} \hat{x}_k$ to both sides of this inequality, we obtain $2 \sum_{k \in N_j \setminus \{j\}} \hat{x}_k \geq \sum_{k \in N_j} \hat{x}_k$. Also, since $v_k = n^{2(n-k)}$, the preference weight of product k is decreasing in k. By the definition of N_j, if $k \in N_j \setminus \{j\}$, then we have $k \leq j - 1$. Therefore, we have $v_k \geq v_{j-1}$ for all $k \in N_j \setminus \{j\}$. In this case, we obtain the chain of inequalities

$$\begin{aligned} \lambda_j \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{v_0 + \sum_{k \in N_j} v_k \, \hat{x}_k} &\leq \lambda_j \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{\sum_{k \in N_j \setminus \{j\}} v_k \, \hat{x}_k} \leq \lambda_j \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{v_{j-1} \sum_{k \in N_j \setminus \{j\}} \hat{x}_k} \\ &= \alpha \, n^{-2(j-1)} \frac{\sum_{k \in N_j} n^{2(k-1)} \, n^{2(n-k)} \, \hat{x}_k}{n^{2(n-j+1)} \sum_{k \in N_j \setminus \{j\}} \hat{x}_k} = \alpha \frac{\sum_{k \in N_j} \hat{x}_k}{n^2 \sum_{k \in N_j \setminus \{j\}} \hat{x}_k} \leq \frac{2\alpha}{n^2}, \end{aligned} \tag{16}$$

where the last inequality follows from the fact that $2 \sum_{k \in N_j \setminus \{j\}} \hat{x}_k \geq \sum_{k \in N_j} \hat{x}_k$. Noting that the inequality in (15) holds for all $j \in M$, whereas the inequality in (16) holds for all $j \notin M$, the expected revenue from the solution $\hat{x}$ satisfies

$$\textsc{Rev}(\hat{x}) = \sum_{j=1}^{n} \lambda_j \frac{\sum_{k \in N_j} r_k \, v_k \, \hat{x}_k}{v_0 + \sum_{k \in N_j} v_k \, \hat{x}_k} \leq \sum_{j \in M} \alpha + \sum_{j \notin M} \frac{2\alpha}{n^2} \leq \alpha \, |M| + 2 \, \alpha = \alpha \sum_{j=1}^{n} \hat{y}_j + 2 \, \alpha, \tag{17}$$

where the second inequality follows from the fact that $\sum_{j \notin M} 2\alpha/n^2 = 2 \, \alpha \, |N \setminus M|/n^2 \leq 2 \, \alpha$ and the second equality follows from the fact that $\hat{y}_j = 1$ if $j \in M$ and $\hat{y}_j = 0$ if $j \notin M$. By (17), we get $\sum_{j=1}^{n} \hat{y}_j \geq (\textsc{Rev}(\hat{x})/\alpha) - 2$. If $M \neq \varnothing$, then noting the definition of $\hat{y}$, we have $\sum_{j=1}^{n} \hat{y}_j = |M| \geq 1$

as well. In this case, we get $\sum_{j=1}^{n} \hat{y}_j \geq \max\{(\textsc{Rev}(\hat{x})/\alpha) - 2, 1\}$. On the other hand, if $M = \varnothing$, then the same argument in the chain of inequalities in (17) implies that $\textsc{Rev}(\hat{x}) \leq 2\,\alpha$, yielding $(\textsc{Rev}(\hat{x})/\alpha) - 2 \leq 0$. We define $\hat{z} \in \{0,1\}^n$ such that $\hat{z}_1 = 1$ and $\hat{z}_j = 0$ for all $N \setminus \{1\}$. Clearly, $\hat{z} \in \{0,1\}^n$ is an independent set since it includes one vertex. Also, we have $\sum_{j=1}^{n} \hat{z}_j = 1$. Noting that $(\textsc{Rev}(\hat{x})/\alpha) - 2 \leq 0$, we obtain $\sum_{j=1}^{n} \hat{z}_j \geq \max\{(\textsc{Rev}(\hat{x})/\alpha) - 2, 1\}$. □

In the next theorem, we use Lemmas 13 and 14 to show that it is NP-hard to approximate the assortment problem in (14) within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$.

Theorem 15 *The assortment problem in* (14) *is NP-hard to approximate within a factor of* $O(n^{1-\epsilon})$ *for any* $\epsilon > 0$.

Proof. We show that a γ-approximation algorithm with $\gamma \geq 1$ for the assortment problem in (14) yields a 6γ-approximation algorithm for the maximum independent set problem. In this case, the result follows from the fact that the maximum independent set problem is NP-hard to approximate within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$. Assume that we have a γ-approximation algorithm for the assortment problem in (14). Consider an instance of the maximum independent set problem over the indirected graph (N, E). We let $x^* \in \{0,1\}^n$ be the optimal solution to the corresponding induced assortment problem. Using the γ-approximation algorithm for the assortment problem in (14), we obtain a γ-approximate solution to the induced assortment problem, denoted by $\hat{x} \in \{0,1\}^n$. Thus, we have $\gamma\,\textsc{Rev}(\hat{x}) \geq \textsc{Rev}(x^*)$. By Lemma 14, given the solution $\hat{x}$, we can construct an independent set $\hat{y} \in \{0,1\}^n$ in polynomial time such that $\sum_{j=1}^{n} \hat{y}_j \geq \max\{(\textsc{Rev}(\hat{x})/\alpha) - 2, 1\}$. The last inequality implies that $\sum_{j=1}^{n} \hat{y}_j + 2 \geq \textsc{Rev}(\hat{x})/\alpha$ and $\sum_{j=1}^{n} \hat{y}_j \geq 1$. Thus, we obtain $3 \sum_{j=1}^{n} \hat{y}_j \geq \sum_{j=1}^{n} \hat{y}_j + 2 \geq \textsc{Rev}(\hat{x})/\alpha$. Letting $y^* \in \{0,1\}^n$ be the optimal solution to the maximum independent set problem, by Lemma 13, given the independent set y^*, there exists a solution $\tilde{x} \in \{0,1\}^n$ to the induced assortment problem such that $\textsc{Rev}(\tilde{x}) \geq \alpha \sum_{j=1}^{n} y^*_j/2$. Therefore, we obtain

$$\sum_{j=1}^{n} y^*_j \leq \frac{2}{\alpha}\,\textsc{Rev}(\tilde{x}) \leq \frac{2}{\alpha}\textsc{Rev}(x^*) \leq \frac{2}{\alpha}\,\gamma\textsc{Rev}(\hat{x}) \leq 6\gamma \sum_{j=1}^{n} \hat{y}_j,$$

where the fourth inequality is by the fact that $3 \sum_{j=1}^{n} \hat{y}_j \geq \textsc{Rev}(\hat{x})/\alpha$. Thus, the independent set $\hat{y}$ is a 6γ-approximate solution to the maximum independent set problem. Also, the independent set $\hat{y}$ is obtained in polynomial time by using the solution $\hat{x}$, which is, in turn, obtained in polynomial time by using the γ-approximation algorithm for the assortment problem in (14). □

H Appendix: Interval Consideration Sets

We consider the assortment problem with so called interval consideration sets of the form $\{k, \ldots, j\}$ with $k < j$. Assuming that there is no limit on the total space consumption of the offered products, we verify that we can build on the ideas in Aouad, Levi and Segev (2015) to obtain solutions whose expected revenues deviate from the optimal expected revenue by a factor of $(1+\epsilon)\,O(\log n)$ or $(1+\epsilon)\,O(\log\log p)$, where n is the number of products and p is the ratio between the largest and smallest product revenues. To obtain these approximation factors, we need to run our FPTAS $O(n)$ or $O(\log p)$ times with a performance guarantee of $1-\epsilon/2$. Thus, choosing, say, $\epsilon = 1/2$ yields the approximation factors of $O(\log n)$ or $O(\log\log p)$. For economy of space, we focus on the approximation factor of $O(\log n)$. Nevertheless, using the development in this section and in Aouad, Levi and Segev (2015), one can obtain the approximation factor of $O(\log\log p)$ as well. We give a brief formulation of the assortment problem with interval consideration sets.

There are n products indexed by $N = \{1, \ldots, n\}$. There are m customer types indexed by $M = \{1, \ldots, m\}$. The revenue of product j is r_j. We use the set $S \subseteq N$, rather than the vector $x \in \{0,1\}^n$, to capture the set of products offered to customers. This change in notation allows us to handle interval consideration sets more naturally. A customer of type k arrives into the system with probability λ_k. We use $N_k = \{l_k, \ldots, u_k\}$ to denote the consideration set of customers of type k. If we offer the subset $S \subseteq N$ of products, then a customer of type k purchases product $j \in S$ with probability $P_{jk}(S)$. We have $P_{jk}(S) = 0$ when $j \notin S$. The choice probabilities are governed by the multinomial logit model so that $P_{jk}(S) = v_j/(v_0 + \sum_{i \in S \cap N_k} v_i)$ for all $j \in S$, where v_j is the preference weight of product j and v_0 is the preference weight of the no purchase option. In this case, if we offer the subset S of products, then we obtain an expected revenue of $\textsc{Rev}(S) = \sum_{k \in M} \lambda_k \sum_{j \in N_k} r_j\, P_{jk}(S)$. We want to solve the problem

$$z^* \;=\; \max_{S \subseteq N} \Big\{ \textsc{Rev}(S) \Big\} \;=\; \max_{S \subseteq N} \Bigg\{ \sum_{k \in M} \lambda_k \Bigg\{ \sum_{j \in N_k} r_j\, P_{jk}(S) \Bigg\} \Bigg\}. \tag{18}$$

We observe that the choice probabilities satisfy the property $P_{jk}(S) \geq P_{jk}(T)$ for all $j \in S$ whenever $S \subseteq T$, which becomes useful in our development.

Our approach is based on the following steps. We partition the set of customer types into $L = O(\log n)$ groups by using the approach in Aouad, Levi and Segev (2015). We formulate an

assortment problem that focuses on each group of customer types separately. We approximate these assortment problems by using nested consideration sets, instead of interval consideration sets, in which case, we can use our FPTAS to obtain approximate solutions to these problems. Picking the best of the approximate solutions from the assortment problems that focus on different groups of customer types, we obtain the desired performance guarantee.

We describe the grouping approach in Aouad, Levi and Segev (2015). We let $[a,b] = \{a,\ldots,b\}$ be the set of consecutive products from a to b with $a \leq b$. Throughout this section, we use $\lfloor x \rfloor$ to denote the largest integer that is no larger than x. We let $\mathsf{Mid}(a,b) = \lfloor (a+b)/2 \rfloor$ be the middle of the interval $[a,b]$. Also, we define the operator $\mathsf{Part}(a,b) = \{[a,\mathsf{Mid}(a,b)],\ [\mathsf{Mid}(a,b)+1,b]\}$ that partitions the interval $[a,b]$ into two. We recursively define the set of intervals as

$$I_1 = \Big\{[1,n]\Big\}, \quad I_2 = \Big\{\mathsf{Part}(1,n)\Big\} = \Big\{[1,\mathsf{Mid}(1,n)],\ [\mathsf{Mid}(1,n)+1,n]\Big\},\ldots,$$
$$I_\ell = \Big\{\mathsf{Part}(a,b) : [a,b] \in I_{\ell-1}\Big\},\ldots,I_L = \Big\{[j,j] : j \in N\Big\}.$$

Since we halve the intervals successively, $L = O(\log n)$. Let $J_\ell = \{\mathsf{Mid}(a,b) : [a,b] \in I_\ell\}$ be the set of middle points of the intervals in I_ℓ. We partition the set of customer types M as

$$G_1 = \Big\{k \in M : N_k \cap J_1 \neq \varnothing\Big\}, \quad G_2 = \Big\{k \in M \setminus G_1 : N_k \cap J_2 \neq \varnothing\Big\},\ldots,$$
$$G_\ell = \Big\{k \in M \setminus \cup_{j=1}^{\ell-1} G_j : N_k \cap J_\ell \neq \varnothing\Big\},\ldots,G_L = \Big\{k \in M \setminus \cup_{j=1}^{L-1} G_j : N_k \cap J_L \neq \varnothing\Big\}.$$

The group G_ℓ includes the customer types that are not in the groups $G_1,\ldots,G_{\ell-1}$ with consideration sets including a middle point of one of the intervals in I_ℓ. By definition $G_\ell \cap G_{\ell'} = \varnothing$ for $\ell \neq \ell'$. Also, we have $J_L = N$. So, if there are any customer types not included in $\cup_{j=1}^{L-1} G_j$, then these customer types are always included in G_L. In this case, it follows that the groups of customer types $G_1,\ldots,G_L$ partition the set of all customer types M.

Noting that $\cup_{\ell=1}^{L} G_\ell = M$, we write the objective function of problem (18) equivalently as $\sum_{\ell=1}^{L} \sum_{k \in G_\ell} \lambda_k \sum_{j \in N_k} r_j\, P_{jk}(S)$. Thus, we can use a relaxation of problem (18) given by

$$\tilde{z} = \sum_{\ell=1}^{L} \max_{S \subseteq N} \left\{ \sum_{k \in G_\ell} \lambda_k \left\{ \sum_{j \in N_k} r_j\, P_{jk}(S) \right\} \right\}, \tag{19}$$

which assumes that we can offer different subsets of products to different groups of customer types. So, we have $\tilde{z} \geq z^*$, where z^* is the optimal objective value of problem (18). For any

subset of customer types $G \subseteq M$, we use $\textsc{Rev}(S,G) = \sum_{k \in G} \lambda_k \sum_{j \in N_k} r_j P_{jk}(S)$ to denote the expected revenue obtained from the subset of customer types G when we offer the subset S of products. Therefore, the maximization problem on the right side of (19) is equivalent to $\max_{S \subseteq N} \textsc{Rev}(S, G_\ell)$. Also, comparing the definitions of $\textsc{Rev}(S)$ and $\textsc{Rev}(S,G)$, it follows that $\textsc{Rev}(S) \geq \textsc{Rev}(S,G)$ for all $S \subseteq N$ and $G \subseteq M$. We let S^*_ℓ be an optimal solution to the problem $\max_{S \subseteq N} \textsc{Rev}(S, G_\ell)$. For the moment, assume that it is possible to find a subset $\hat{S}_\ell$ of products for each $\ell = 1, \ldots, L$ that satisfies the inequality $2\,(1+\epsilon)\,\textsc{Rev}(\hat{S}_\ell, G_\ell) \geq \textsc{Rev}(S^*_\ell, G_\ell)$. In this case, noting that $\textsc{Rev}(S) \geq \textsc{Rev}(S,G)$, we obtain the chain of inequalities

$$\begin{aligned} 2\,(1+\epsilon)\,L \max\Big\{\textsc{Rev}(\hat{S}_\ell) : \ell = 1, \ldots, L\Big\} &\geq 2\,(1+\epsilon)\,L \max\Big\{\textsc{Rev}(\hat{S}_\ell, G_\ell) : \ell = 1, \ldots, L\Big\} \\ &\geq L \max\Big\{\textsc{Rev}(S^*_\ell, G_\ell) : \ell = 1, \ldots, L\Big\} \geq \sum_{\ell=1}^{L} \textsc{Rev}(S^*_\ell, G_\ell) = \tilde{z} \geq z^*. \end{aligned}$$

So, since $L = O(\log n)$, we obtain $(1+\epsilon)\,O(\log n) \max\{\textsc{Rev}(\hat{S}_\ell) : \ell = 1, \ldots, L\} \geq z^*$. In other words, if we check the objective value provided for problem (18) by each one of the subsets $\{\hat{S}_\ell : \ell = 1, \ldots, L\}$ and pick the one that provides the largest objective value, then we obtain a subset of products whose expected revenue deviates from the optimal by at most a factor of $(1+\epsilon)\,O(\log n)$, as desired. In the rest of this section, we show how to obtain a subset $\hat{S}_\ell$ of products such that $2\,(1+\epsilon)\,\textsc{Rev}(\hat{S}_\ell, G_\ell) \geq \textsc{Rev}(S^*_\ell, G_\ell)$ for each $\ell = 1, \ldots, L$. We begin by two observations which are also stated by Aouad, Levi and Segev (2015). These observations follow from the way we construct the sets of intervals $I_1, \ldots, I_L$ and the groups $G_1, \ldots, G_L$.

- First, fix any $\ell = 1, \ldots, L$ and any customer type $k \in G_\ell$. There exists exactly one interval $[a,b] \in I_\ell$ such that $N_k \subseteq [a,b]$. For any other interval $[a',b'] \in I_\ell$, we have $[a',b'] \cap N_k = \varnothing$.
- Second, fix any $\ell = 1, \ldots, L$ and any customer type $k \in G_\ell$. Let the interval $[a,b] \in I_\ell$ be such that $N_k \subseteq [a,b]$, which exists by the first observation. We have $\mathsf{Mid}(a,b) \in N_k$.

For $\ell = 1, \ldots, L$ and $j \in J_\ell$, we define the set of customer types $G_\ell(j) = \{k \in G_\ell : j \in N_k\}$. In the next lemma, we use the observations above to show that $\{G_\ell(j) : j \in J_\ell\}$ partition G_ℓ.

Lemma 16 *For fixed $\ell = 1, \ldots, L$, consider the products $j \in J_\ell$ and $i \in J_\ell$ with $j \neq i$. Then, we have $G_\ell(j) \cap G_\ell(i) = \varnothing$. Furthermore, we have $G_\ell = \cup_{j \in J_\ell} G_\ell(j)$.*

Proof. First, we show that $G_\ell(j) \cap G_\ell(i) = \varnothing$ for $j \neq i$. To get a contradiction, assume that there exists some k such that $k \in G_\ell(j)$ and $k \in G_\ell(i)$. Noting that $j \in J_\ell$ and $i \in J_\ell$, let $[a,b] \in I_\ell$ be such that $\mathsf{Mid}(a,b) = j$ and $[a',b'] \in I_\ell$ be such that $\mathsf{Mid}(a',b') = i$. Since $j \neq i$, the intervals $[a,b]$ and $[a',b']$ are distinct. Since $k \in G_\ell(j)$ and $k \in G_\ell(i)$, we have $\mathsf{Mid}(a,b) = j \in N_k$ and $\mathsf{Mid}(a',b') = i \in N_k$, which implies that $[a,b] \cap N_k \neq \varnothing$ and $[a',b'] \cap N_k \neq \varnothing$ for two distinct intervals $[a,b]$ and $[a',b']$, which contradicts the first observation above. Second, we show that $G_\ell = \cup_{j \in J_\ell} G_\ell(j)$. By the definition of $G_\ell(j)$, we immediately have $\cup_{j \in J_\ell} G_\ell(j) \subseteq G_\ell$. It remains to show that $G_\ell \subseteq \cup_{j \in J_\ell} G_\ell(j)$. Consider a customer type $k \in G_\ell$. By the second observation above, let $[a,b] \in I_\ell$ be such that $\mathsf{Mid}(a,b) \in N_k$. Thus, by the definition of $G_\ell(j)$, noting that $\mathsf{Mid}(a,b) \in J_\ell$, we obtain $k \in G_\ell(\mathsf{Mid}(a,b))$, which implies that $k \in \cup_{j \in J_\ell} G_\ell(j)$. □

In the next lemma, we show that the consideration sets of two customer types $k \in G_\ell(j)$ and $k' \in G_\ell(i)$ with $j \neq i$ do not overlap with each other. This lemma ultimately allows us to decompose the problem $\max_{S \subseteq N} \mathrm{Rev}(S, G_\ell)$ by the elements of J_ℓ and solve this problem by focusing on each set of customer types in $\{G_\ell(j) : j \in J_\ell\}$ separately. The observations right before Lemma 16 play an important role in the proof of the next lemma as well.

Lemma 17 *For fixed $\ell = 1, \ldots, L$, consider the customer types $k \in G_\ell(j)$ and $k' \in G_\ell(i)$ with $j \in J_\ell$, $i \in J_\ell$ and $j \neq i$. Then, we have $N_k \cap N_{k'} = \varnothing$.*

Proof. Noting the first observation right before Lemma 16, we let the interval $[a,b] \in I_\ell$ be such that $N_k \subseteq [a,b]$. Since $k \in G_\ell(j)$, the definition of $G_\ell(j)$ implies that $j \in N_k$. Therefore, we obtain $j \in N_k \subseteq [a,b]$ and $j \in J_\ell$, which is to say that j is in the interval $[a,b] \in I_\ell$ and j is a middle point of one of the intervals in I_ℓ. Since the intervals in I_ℓ are disjoint, it follows that we must have $j = \mathsf{Mid}(a,b)$. Similarly, we let the interval $[a',b'] \in I_\ell$ be such that $N_{k'} \subseteq [a',b']$. By using the same argument, it also follows that $i = \mathsf{Mid}(a',b')$. In this case, since we have $i = \mathsf{Mid}(a,b)$, $j = \mathsf{Mid}(a',b')$ and $j \neq i$, the intervals $[a,b]$ and $[a',b']$ must be distinct from each other. Therefore, using the first observation right before Lemma 16 once more, since we have $N_k \subseteq [a,b]$ and the intervals $[a,b]$ and $[a',b']$ are distinct from each other, we get $[a',b'] \cap N_k = \varnothing$. Noting that $N_{k'} \subseteq [a',b']$, having $[a',b'] \cap N_k = \varnothing$ implies that $N_k \cap N_{k'} = \varnothing$. □

By Lemma 16, $\{G_\ell(j) : j \in J_\ell\}$ partition G_ℓ. Therefore, using the definition of $\textsc{Rev}(S,G)$, we have $\textsc{Rev}(S, G_\ell) = \sum_{k \in G_\ell} \lambda_k \sum_{i \in N_k} r_i \, P_{ik}(S) = \sum_{j \in J_\ell} \sum_{k \in G_\ell(j)} \lambda_k \sum_{i \in N_k} r_i \, P_{ik}(S)$. Recalling

S^*_ℓ is an optimal solution to the problem $\max_{S\subseteq N} \textsc{Rev}(S, G_\ell)$, noting the equivalent expression for $\textsc{Rev}(S, G_\ell)$ given in the last chain of equalities, S^*_ℓ is also an optimal solution to the problem $\max_{S\subseteq N}\{\sum_{j\in J_\ell}\sum_{k\in G_\ell(j)}\lambda_k \sum_{i\in N_k} r_i\, P_{ik}(S)\}$. By Lemma 17, the consideration sets of two customer types $k \in G_\ell(j)$ and $k' \in G_\ell(i)$ with $j \neq i$ do not overlap. Thus, the last maximization problem decomposes by the elements of J_ℓ and we obtain

$$\max_{S\subseteq N}\Big\{\textsc{Rev}(S, G_\ell)\Big\} = \max_{S\subseteq N}\left\{\sum_{j\in J_\ell}\sum_{k\in G_\ell(j)}\lambda_k\left\{\sum_{i\in N_k} r_i\, P_{ik}(S)\right\}\right\}$$
$$= \sum_{j\in J_\ell}\max_{S\subseteq \cup_{k\in G_\ell(j)} N_k}\left\{\sum_{k\in G_\ell(j)}\lambda_k\left\{\sum_{i\in N_k} r_i\, P_{ik}(S)\right\}\right\} = \sum_{j\in J_\ell}\max_{S\subseteq \cup_{k\in G_\ell(j)} N_k}\Big\{\textsc{Rev}(S, G_\ell(j))\Big\}. \quad (20)$$

Since the subset S^*_ℓ of products is an optimal solution to the first maximization problem in (20), if we use $S^*_\ell(j)$ to denote an optimal solution to the last maximization problem in (20), then we obtain $\textsc{Rev}(S^*_\ell, G_\ell) = \sum_{j\in J_\ell}\textsc{Rev}(S^*_\ell(j), G_\ell(j))$. Furthermore, the subset $\cup_{j\in J_\ell} S^*_\ell(j)$ of products is also an optimal solution to the first maximization problem in (20) as well. Thus, we focus on solving the last maximization problem in (20). In the next theorem, we show how to obtain an approximate solution to the last maximization problem in (20). In the proof of this theorem, we establish that we can approximate the problem $\max_{S\subseteq\cup_{k\in G_\ell(j)}N_k}\textsc{Rev}(S, G_\ell(j))$ by using assortment problems with nested consideration sets, instead of interval consideration sets. In this case, we can use our FPTAS to obtain an approximate solution to the assortment problems with nested consideration sets. In this theorem, $R_{\max}$ and $V_{\max}$ are as defined right before Lemma 2.

Theorem 18 *Letting $z^*_\ell(j) = \max_{S\subseteq\cup_{k\in G_\ell(j)}N_k}\textsc{Rev}(S, G_\ell(j))$, there exists an algorithm such that for any $\epsilon \in (0,1)$, the algorithm runs in $O(n^3\log(nR_{\max})\log(nV_{\max})/\epsilon^2)$ operations and generates a subset $\hat{S}_\ell(j) \subseteq \cup_{k\in G_\ell(j)}N_k$ of products that satisfies $2\,(1+\epsilon)\,\textsc{Rev}(\hat{S}_\ell(j), G_\ell(j)) \geq z^*_\ell(j)$.*

Proof. We let $S^*_\ell(j)$ be an optimal solution to the problem $\max_{S\subseteq\cup_{k\in G_\ell(j)}N_k}\textsc{Rev}(S, G_\ell(j))$. For any customer type $k \in G_\ell(j)$, the definition of $G_\ell(j)$ implies that $j \in N_k$. In this case, we split the consideration set $N_k = \{l_k, \ldots, u_k\}$ into two sets $N^L_k = \{l_k, \ldots, j-1\}$ and $N^U_k = \{j, \ldots, u_k\}$ so that $N^L_k \cup N^U_k = N_k$ and $N^L_k \cap N^U_k = \varnothing$. Note that $\sum_{k\in G_\ell(j)}\lambda_k\sum_{i\in N^L_k} r_i\, P_{ik}(S^*_\ell(j)) \leq \sum_{k\in G_\ell(j)}\lambda_k\sum_{i\in N^L_k} r_i\, P_{ik}(S^*_\ell(j)\cap N^L_k) \leq \max_{S\subseteq\cup_{k\in G_\ell(j)}N^L_k}\{\sum_{k\in G_\ell(j)}\lambda_k\sum_{i\in N^L_k} r_i\, P_{ik}(S)\}$, where the first inequality uses the fact that if $S \subseteq T$, then $P_{jk}(S) \geq P_{jk}(T)$ for all $j \in S$. By using a similar argument, but focusing on N^U_k instead of N^L_k, we have $\sum_{k\in G_\ell(j)}\lambda_k\sum_{i\in N^U_k} r_i\, P_{ik}(S^*_\ell(j)) \leq$

$\sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^U} r_i\, P_{ik}(S_\ell^*(j) \cap N_k^U) \leq \max_{S\subseteq \cup_{k\in G_\ell(j)} N_k^U}\{\sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^U} r_i\, P_{ik}(S)\}$ as well. So, noting the definition of $S_\ell^*(j)$ at the beginning of the proof, we get

$$z_\ell^*(j) = \textsc{Rev}(S_\ell^*(j), G_\ell(j)) = \sum_{k\in G_\ell(j)} \lambda_k \left\{ \sum_{i\in N_k^L} r_i\, P_{ik}(S_\ell^*(j)) + \sum_{i\in N_k^U} r_i\, P_{ik}(S_\ell^*(j)) \right\}$$
$$\leq \max_{S\subseteq \cup_{k\in G_\ell(j)} N_k^L} \left\{ \sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^L} r_i\, P_{ik}(S) \right\} + \max_{S\subseteq \cup_{k\in G_\ell(j)} N_k^U} \left\{ \sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^U} r_i\, P_{ik}(S) \right\}. \quad (21)$$

The consideration sets in the first maximization problem above are of the form $\{l_k, \ldots, j-1\}$ for all $k \in G_\ell(j)$, where $j \in J_\ell$ is fixed. Therefore, these consideration sets are nested. Similarly, the consideration sets in the second maximization problem above are of the form $\{j, \ldots, u_k\}$ for all $k \in G_\ell(j)$, where $j \in J_\ell$ is fixed. These consideration sets are nested as well. We use $\hat{z}_\ell^L(j)$ to denote the optimal objective value of the first maximization problem in (21). Since the consideration sets in this problem are nested, we can use our FPTAS with a performance guarantee of $1-\epsilon/2$ to find a subset $\hat{S}_\ell^L(j) \subseteq \cup_{k\in G_\ell(j)} N_k$ of products such that $\sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^L} r_i\, P_{ik}(\hat{S}_\ell^L(j)) \geq (1-\epsilon/2)\, \hat{z}_\ell^L(j)$. In this case, we obtain $\hat{z}_\ell^L(j) \leq (1+\epsilon)(1-\epsilon/2)\, \hat{z}_\ell^L(j) \leq (1+\epsilon) \sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k^L} r_i\, P_{ik}(\hat{S}_\ell^L(j)) \leq (1+\epsilon) \sum_{k\in G_\ell(j)} \lambda_k \sum_{i\in N_k} r_i\, P_{ik}(\hat{S}_\ell^L(j)) = (1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell^L(j), G_\ell(j))$. Thus, we have $\hat{z}_\ell^L(j) \leq (1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell^L(j), G_\ell(j))$. By using a similar argument, letting $\hat{z}_\ell^U(j)$ be the optimal objective value of the second maximization problem in (21), we can use our FPTAS with a performance guarantee of $1-\epsilon/2$ to find a subset $\hat{S}_\ell^U(j) \subseteq \cup_{k\in G_\ell(j)} N_k$ of products that satisfies $\hat{z}_\ell^U(j) \leq (1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell^U(j), G_\ell(j))$ as well. Therefore, by (21), we get

$$z_\ell^*(j) \leq \hat{z}_\ell^L(j) + \hat{z}_\ell^U(j) \leq (1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell^L(j), G_\ell(j)) + (1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell^U(j), G_\ell(j))$$
$$\leq 2\,(1+\epsilon) \max\left\{ \textsc{Rev}(\hat{S}_\ell^L(j), G_\ell(j)),\ \textsc{Rev}(\hat{S}_\ell^U(j), G_\ell(j)) \right\},$$

which implies that if we let $\hat{S}_\ell(j) = \arg\max\{\textsc{Rev}(S, G_\ell(j)) : S \in \{\hat{S}_\ell^L(j), \hat{S}_\ell^U(j)\}\}$, then we have $z_\ell^*(j) \leq 2\,(1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell(j), G_\ell(j))$. The running time to get $\hat{S}_\ell(j)$ follows from Theorem 11. $\square$

Recall that S_ℓ^* is an optimal solution to the first maximization problem in (20). To conclude the discussion, we use the subsets $\{\hat{S}_\ell(j) : j \in J_\ell\}$ in Theorem 18 to come up with a subset $\hat{S}_\ell$ that satisfies $2\,(1+\epsilon)\, \textsc{Rev}(\hat{S}_\ell, G_\ell) \geq \textsc{Rev}(S_\ell^*, G_\ell)$. For $j \neq i$, consider the subsets $\hat{S}_\ell(j)$ and $\hat{S}_\ell(i)$ in Theorem 18. Since $\hat{S}_\ell(j) \subseteq \cup_{k\in G_\ell(j)} N_k$ and $\hat{S}_\ell(i) \subseteq \cup_{k\in G_\ell(i)} N_k$, Lemma 17 implies that $\hat{S}_\ell(j) \cap \hat{S}_\ell(i) = \varnothing$ for $j \neq i$. In this case, noting that $\cup_{j\in J_\ell} G_\ell(j) = G_\ell$ by Lemma 16, if

we let $\hat{S}_\ell = \cup_{j \in J_\ell} \hat{S}_\ell(j)$, then we obtain $\textsc{Rev}(\hat{S}_\ell, G_\ell) = \sum_{j \in J_\ell} \textsc{Rev}(\hat{S}_\ell(j), G_\ell(j))$. Furthermore, since S^*_ℓ and $S^*_\ell(j)$ are optimal solutions to the first and last maximization problems in (20) and noting the definition of $z^*_\ell(j)$ in Theorem 18, we obtain $\textsc{Rev}(S^*_\ell, G_\ell) = \sum_{j \in J_\ell} z^*_\ell(j) \leq 2\,(1+\epsilon) \sum_{j \in J_\ell} \textsc{Rev}(\hat{S}_\ell(j), G_\ell(j)) = 2\,(1+\epsilon)\,\textsc{Rev}(\hat{S}_\ell, G_\ell)$, where the inequality follows from Theorem 18. Therefore, we obtain $\textsc{Rev}(S^*_\ell, G_\ell) \leq 2\,(1+\epsilon)\,\textsc{Rev}(\hat{S}_\ell, G_\ell)$, as desired. To sum up our discussion, we compute the subset $\hat{S}_\ell(j)$ of products for all $\ell = 1, \ldots, L$ and $j \in J_\ell$ by using our FPTAS, as discussed in the proof of Theorem 18. We let $\hat{S}_\ell = \cup_{j \in J_\ell} \hat{S}_\ell(j)$ for all $\ell = 1, \ldots, L$. In this case, if we check the objective value provided for problem (18) by each one of the subsets $\{\hat{S}_\ell : \ell = 1, \ldots, L\}$ and pick the one that provides the largest objective value, then we obtain a subset of products providing an expected revenue that deviates from the optimal by at most a factor of $(1+\epsilon)\,O(\log n)$. Since $|J_\ell| = 2^{\ell-1}$ and $L = O(\log n)$, there are $O(n)$ subsets in the collection $\{\hat{S}_\ell(j) : \ell = 1, \ldots, L,\ j \in J_\ell\}$. Therefore, computing all of the subsets in the collection $\{\hat{S}_\ell(j) : \ell = 1, \ldots, L,\ j \in J_\ell\}$ requires running our FPTAS $O(n)$ times.