CrossMark

Assortment optimization over time

James M. Davis *, Huseyin Topaloglu, David P. Williamson

School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14853, United States

A B S T R A C T

In this note we introduce the problem of assortment optimization over time. We have a sequence of time steps and can introduce one new product per time step. Once introduced a product cannot be removed. The goal is to determine which products to introduce so as to maximize revenue over all time steps under some choice model. Given a $1/\alpha$-approximation algorithm for the capacitated assortment optimization problem we give a $1/2\alpha$-approximation algorithm for this problem.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

A fundamental problem in revenue management involves finding a profitable assortment of products to offer to customers given that customers choose within the offered assortment according to a certain choice model. The main tradeoff in this problem setting is that if we offer a limited assortment, then customers may often decide to leave without making a purchase, but if we offer a wide assortment, then we may end up adding products with low revenue contributions to the assortment, which may dilute the revenue potential. This fundamental tradeoff is often complicated by the fact that it may take time for a firm to build up the product assortment it wants to offer to its customers.

In this paper, we consider assortment optimization problems over time, where a firm has some initial product portfolio and must make a strategic decision to gradually change to some different product portfolio in order to maximize revenue. We consider a firm that can initially drop any number of products from its portfolio but must sequentially decide which product, if any, to add to its offered assortment at each time period. The sequential nature of the product addition into the marketplace may be a result of product development time or constraints on manufacturing volume. Customers arriving at a particular time period choose among the current assortment according to a particular choice model. The goal is to decide how to build the product portfolio over time to maximize the total expected revenue over a finite planning horizon.

A special case of this sequential problem, when the initial product portfolio is empty and the time horizon is at least the number of products, is particularly relevant in online retail. When a customer searches for a product on a web site they are frequently presented with a list of items available to purchase. Customer have a finite patience level and, based on this patience level, will look through some length of this list, form a consideration set to choose from, and then make a purchasing decision from the consideration set based on some choice model. The different patience levels correspond to the different time periods in the sequential problem. In this setting the firm must decide how to order the items on its web site which, implicitly, is a decision about how to sequentially add items to customers' consideration sets.

There is growing literature on assortment optimization problems, as these problems are faced on strategic and tactical levels in numerous retail and revenue management settings. In brick-and-mortar retail, firms are interested in finding the right mix of products to offer and such decisions are made infrequently on a strategic level. In revenue management, airlines and hotels continuously update their product offerings in response to the remaining time in the selling horizon and the remaining capacity, and so the assortment problems in those setting have more of an operational nature. Several authors studied the so-called static assortment problem, where the goal is to find an assortment of products that maximizes the expected revenue obtained from each customer. Talluri and van Ryzin [18] show that if the customers choose according to the multinomial logit model, then the static assortment problem can be solved efficiently by checking the expected revenue from each assortment that includes a certain number of products with the largest revenues. Rusmevichientong et al. [17] solve the same problem when there is a limit on the number

of products that can be offered. Bront et al. [3], Desir and Goyal [8] and Mendez-Diaz et al. [15] study the static assortment problem when customers choose according to a mixture of multinomial logit models, show that the problem is NP-complete, strengthen an integer programming formulation for the problem with valid inequalities and propose various approximation schemes. Davis et al. [7] consider the same problem under the nested logit model and show that the corresponding static assortment problem can be solved efficiently. Alptekinoglu and Semple [1] focus on static assortment problems when customers choose under the exponential choice model.

In dynamic assortment problems, the offered assortment is adjusted over time, possibly due to depleted product inventories, better understanding of customer choice processes or changes in customer tastes. Honhon et al. [13] and Mahajan and van Ryzin [14] study the problem of finding an assortment to offer and the corresponding stocking quantities with the understanding that customers choose only among the products that are still in stock. Bernstein et al. [2] and Golrezaei et al. [12] consider the problem of dynamically customizing the assortment offerings based on the preferences of each customer and remaining product inventories. Caro et al. [5] and Cinar and Martinez-de-Albeniz [6] study assortment problems where the attractiveness of the products diminishes over time and they seek optimal policies to replace such products. Caro and Gallien [4] and Ulu et al. [19] develop models where the assortment offering needs to be adjusted over time in response to a better understanding of the customer choice process.

The rest of the note is organized as follows. Section 2 describes our assortment problem, where the firm needs to gradually build its product portfolio, and introduces the two problem settings. Section 3 analyzes approximation algorithms for the first problem setting and Section 4 analyzes approximation algorithms for the second setting. Section 5 discusses the complexity of the problem.

2. Preliminaries

Let N be the set of items that can be offered for sale, and let $n = |N|$. We let r_j be the revenue earned when item j is sold. Let $P_j(S)$ be the probability that item j is purchased if $S \subseteq N$ is offered for sale; then $P_j(S) = 0$ if $j \notin S$.

We wish to study the *assortment optimization problem over time*. Intuitively, we start with some initial set of items being offered and would like to remove items from this initial set, followed by sequentially adding items, at most one item at each of T time steps, so that we maximize the overall expected revenue achieved over the given time horizon. The items we remove from the initial set are removed before the beginning of the first time period. Once an item is offered for sale, it remains available to purchase for the remainder of the time horizon. More precisely, given an initial set I we would like to find sets $S_0, S_1, S_2, \ldots, S_T$ such that $S_0 \subseteq I$, $|S_t| \leq |S_0| + t$ for all $1 \leq t \leq T$, and $S_0 \subseteq S_1 \subseteq S_2 \subseteq \cdots \subseteq S_T$ that maximizes $\sum_{t=1}^{T} R(S_t)$. Note that S_0, the set of items we retain from I, does not appear in the expression for expected revenue we would like to maximize.

We consider two problem settings. In the first setting we consider choice models where two properties hold. First,

$$P_j(S) \geq P_j(T) \quad \forall T \; \forall j \in S \subset T;$$

that is, the probability of purchasing item j cannot increase if we offer a larger set of items. This holds for any utility maximization models, for example. Second, $\sum_{j \in S} P_j(S) \leq 1$ for any non-empty set of items S. If an assortment S is offered for sale, then with probability $1 - \sum_{j \in S} P_j(S)$, no item is purchased. The expected revenue for a set S of items is $R(S) = \sum_{j \in S} r_j P_j(S)$. For lack of a better term, let us call such choice models *monotone* choice

models. The multinomial logit and nested logit choice model are two examples of a monotone choice models.

For our first result, we will use a solution to the assortment optimization problem in which there is a capacity c on the number of items that can be offered for sale; that is, we wish to find a set S_c^* with $|S_c^*| \leq c$ that maximizes $R(S_c^*)$. We call this the *capacitated assortment optimization problem*. For an optimal solution S_c^*, let OPT_c be the expected revenue obtained; that is, $\text{OPT}_c = R(S_c^*)$. For our first result, we will suppose that we have a polynomial-time $\frac{1}{\alpha}$-approximation algorithm for the capacitated assortment optimization problem under the given monotone choice model. The algorithm is guaranteed to find a set with revenue at least $\frac{1}{\alpha} \text{OPT}_c$. For example, there are algorithms that find assortments of value exactly OPT_c for the multinomial logit and nested logit choice models (see [11,17]).

In the second problem setting, we will consider the case in which we only know that the revenue function $R(S)$ is *monotone* (that is, $R(S) \leq R(T)$ for any $S \subseteq T \subseteq N$) and *submodular* (that is, for any $j \notin S \subseteq T \subseteq N$, then $R(S \cup \{j\}) - R(S) \geq R(T \cup \{j\}) - R(T)$), without knowing anything about the underlying choice model. We present an example of a choice model that satisfies these properties in the section showing our hardness result. Additionally, in this setting we assume that our initial set of items $I = \emptyset$.

For our second result, we will use a well-known greedy $(1 - 1/e)$-approximation algorithm for finding a maximum valued set S with $|S| \leq c$ for any monotone, submodular set function due to [16]. The greedy algorithm repeatedly chooses an element to add to the set S until $|S| = c$, and each time adds the element in $N - S$ that maximizes the marginal gain; that is, it selects $j \in N - S$ that maximizes $R(S \cup \{j\}) - R(S)$ and adds it to S.

3. Monotone choice model

In this section, we first give a 1/2-approximation algorithm for the assortment optimization problem over time for monotone choice models, given an exact polynomial-time algorithm for the capacitated assortment optimization problem under the choice model. From this result it is straightforward to generalize and achieve a $1/2\alpha$-approximation algorithm given a $1/\alpha$-approximation algorithm for the capacitated assortment optimization problem. We then give a $(1 - 1/e)$-approximation algorithm for the assortment optimization problem over time for monotone submodular revenue functions.

The first algorithm works as follows. We initially use the polynomial-time algorithm to find an optimal assortment of at most capacity t for all t, $1 \leq t \leq T$. Let τ be the capacity for which we get the largest revenue assortment and let S_τ be the assortment. We set our initial set of items to be $S_0 = S_\tau \cap I$. We then order the items of S_τ/I by nonincreasing value of $r_j P_j(S_\tau)$; we will offer items for sale in this order. Let $k = |S_\tau/I|$. Without loss of generality, assume that items of S_τ/I are indexed by $1, 2, \ldots, k$ so that $r_1 P_1(S_\tau) \geq r_2 P_2(S_\tau) \geq \cdots \geq r_k P_k(S_\tau)$. Then we set $S_1 = S_0 \cup \{1\}, S_2 = S_0 \cup \{1, 2\}, \ldots, S_{k-1} = S_0 \cup \{1, \ldots, k-1\}$, and $S_t = S_0 \cup \{1, \ldots, k\} = S_\tau$ for all $k \leq t \leq T$.

We now analyze the algorithm. Let OPT be the overall expected revenue of an optimal assortment over time. We observe the following.

Observation 3.1.

$$\text{OPT} \leq \sum_{t=1}^{T} \text{OPT}_t \leq T \cdot \text{OPT}_\tau = T \cdot R(S_\tau^*),$$

where $\text{OPT}_t = R(S_t^*)$ is the optimal expected revenue for the capacitated assortment optimization problem with capacity t, S_t^* is the optimal solution to this capacitated problem, and τ is such that $\text{OPT}_\tau \geq \text{OPT}_t$ for $1 \leq t \leq T$.