Multi-stage and Multi-customer Assortment Optimization with Inventory Constraints

Elaheh Fata
Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, efata@mit.edu

Will Ma
Decision, Risk, and Operations Division, Graduate School of Business, Columbia University, wm2428@gsb.columbia.edu

David Simchi-Levi
Institute for Data, Systems, and Society, Department of Civil and Environmental Engineering, and Operations Research Center, Massachusetts Institute of Technology, dslevi@mit.edu

We consider an assortment optimization problem where a customer chooses a single item from a sequence of sets shown to her, while limited inventories constrain the items offered to customers over time. In the special case where all of the assortments have size one, our problem captures the online stochastic matching with timeouts problem. For this problem, we derive a polynomial-time approximation algorithm which earns at least $1 - \ln(1/e)$, or 0.51, of the optimum. This improves upon the previous-best approximation ratio of 0.46, and furthermore, we show that it is tight. For the general assortment problem, we establish the first constant-factor approximation ratio of 0.09 for the case that different types of customers value items differently, and an approximation ratio of 0.15 for the case that different customers value each item the same. Our algorithms are based on rounding an LP relaxation for multi-stage assortment optimization, and improve upon previous randomized rounding schemes to derive the tight ratio of $1 - \ln(1/e)$.

1. Introduction

In the assortment optimization problem, a firm decides on a set of items to offer to a customer for sale. One of the key considerations is cannibalization, where the firm might not offer an item that serves as a substitute for a different, more profitable item. In the traditional assortment optimization problem, the customer is presented with a single assortment, and chooses an item to purchase (or no item at all) based on all her options. With the advent of e-commerce, more sophisticated interactions with the customer that occur over multiple stages have become possible. This has led to a booming literature for modeling these interactions, which we review in Section 1.2. In this work, we closely follow the model from Liu et al. (2019b,a), which we now describe. An arriving customer is interested in purchasing a specific category of items, e.g. a shirt. The e-tailer shows her an assortment of shirts on the first web/mobile page. If the customer sees a shirt she is satisfied with, then she purchases it and leaves the system. Otherwise, she proceeds to the next page, where the e-tailer shows her a new assortment of shirts. The different assortments shown to a customer can be overlapping or non-overlapping, and both are actively used in different online...
advertising campaigns. Some advertisers, such as video games developers, might prefer to offer an item at most once to a customer as they believe if the customer is not interested in that video game then she would not purchase it regardless of the other video games that accompany it in that stage of advertising. On the other hand, as shown by Chernev (2006) and Chernev et al. (2012), for some category of products, such as jewelry items, advertisers might prefer to show an item multiple times to a customer via different assortments. This is because these advertisers believe that the assortment through which an item is presented can affect the possibility that a customer purchases that item. The customer then leaves the system once she either purchases an item, or runs out of patience to view more pages.

While the aforementioned literature has led to insightful models backed by empirical evidence, it focuses on the optimization problem for a single customer, and ignores the original revenue management problem of how to control a firm’s limited item inventories over time (see Talluri and Van Ryzin (2006)). In this work, we study a multi-stage, multi-customer assortment optimization problem, where limited inventories are offered to multiple customers over a finite time horizon, and furthermore can be offered to those customers in multiple stages. This captures both demand-facing and supply-facing considerations for an e-tailer, where the assortments are optimized to reduce both the cannibalization within a single customer’s demand, and also the “cannibalization” of inventory supplies that are better suited for another customer.

We also study the special case where all of the assortments have size 1. In this case, the firm shows the items to the customer one-at-a-time, which gives the firm increased control and is also natural in some applications such as online dating. This special case corresponds to the online stochastic matching with timeouts problem, which has been studied in a stream of work (Bansal et al. 2012, Adamczyk et al. 2015, Brubach et al. 2017) originating from Chen et al. (2009).

In our model, we assume that the arrival process is known and stationary, although we discuss where this assumption can be relaxed. That is, there is a finite universe of customer types, and the probability distribution for the type of an arriving customer is known and unchanging over time. This is the simplest arrival process to consider, which is also assumed throughout the work in online stochastic matching. In our case, a customer type can be interpreted as a customer segment (e.g. females from LA aged 25–35), and is assumed to have a known choice model (estimated from historical data) as well as a patience level indicating the number of stages over which assortments can be offered to that customer type. Different customer types are also allowed to have different willingness-to-pay for the same item. The firm’s objective is to maximize the total revenue it expects to earn over the time horizon, through a policy which dynamically computes assortments to offer while respecting inventory constraints.
This problem can be solved using dynamic programming, but that is impractical due to the curse of dimensionality; in fact, Liu et al. (2019a) show that even the single-customer problem is NP-hard. Therefore, we seek polynomial-time approximation algorithms which have performance guarantees relative to the optimal dynamic program. In fact, our guarantees are relative to a stronger Multi-stage Choice-based Deterministic Linear Program (MCDLP) benchmark which we introduce. The MCDLP provides an upper-bound on the revenue of any feasible policy, and generalizes the existing CDLP by allowing for assortments to be offered in multiple stages with the constraint that these assortments must respect patience levels.

1.1. Contributions
We derive two main sets of algorithms both based on rounding an optimal MCDLP solution. First, in the special case of online stochastic matching with timeouts, our algorithm imitates a scaled copy of the LP solution using a randomized rounding scheme. We improve previous schemes by increasing the likelihood that items with lower sales probabilities are offered first, and then use the framework of Brubach et al. (2017) to bound the probability that each item has positive remaining inventory. Under some additional technical assumptions (see Section 3), this leads to an approximation ratio of $1 - \frac{\ln(2 - \frac{1}{e})}{\ln(2 - 1/e)} \approx 0.51$ relative to the MCDLP, improving the best-known guarantee from 0.46 (due to Brubach et al. (2017)) to 0.51. We also improve the best-known impossibility result from $1 - 1/e$ to $1 - \ln(2 - 1/e)$ through the analysis of a differential equation, showing that our approximation ratio of $1 - \frac{\ln(2 - 1/e)}{\ln(2 - 1/e)}$ is tight, and closes the gap for online stochastic matching with timeouts.

We then make appropriate modifications to this algorithm so that it can offer assortments of items in each stage of the problem as opposed to a single item. However, in the general assortment problem, rounding the MCDLP solution is more challenging, because the LP can suggest fractionally-intersecting assortments for a single customer, effectively having “multiple chances” to sell her the same item over multiple stages. As discussed earlier, the possibility of showing an item multiple times to a customer is desirable in some settings while not favorable in some other settings. We show that in the settings for which it is allowed to offer an item repeatedly to a customer, the algorithm proposed to address the online stochastic matching with timeouts problem can be modified to offer assortments of items to customers. We prove that this algorithm has an approximation guarantee of 0.51, under some technical assumptions.

To address the settings in which it is not desirable to show an item multiple times to a customer, our previous random ordering is not possible to maintain, we instead consider the assortments in a uniformly random order, like in (Bansal et al. 2012). We discard from each assortment the items which have already been offered, and show that this leads to a constant-factor approximation ratio
Table 1  Summary of the main approximation algorithms in the work. The second column demonstrates the approximation guarantees provided in this work and whether they are tight. The last column specifies the arrival model of customers. The results in the first three rows hold under some technical assumptions that are further explained in Sections 3 and 4.

| Problem                                                                 | Approx. guarantee | Arrival model                      |
|------------------------------------------------------------------------|-------------------|------------------------------------|
| Online stochastic matching with timeouts                                | $1 - \ln(2 - \frac{1}{e}) \simeq 0.51$ (optimal) | Known and stationary               |
| Multi-stage and multi-customer assortment optimization with repeated offerings | $1 - \ln(2 - \frac{1}{e}) \simeq 0.51$ (optimal) | Known and stationary               |
| Multi-stage and multi-customer assortment optimization without repeated offerings | 0.09              | Known, stationary, and Integral    |
| Multi-stage and multi-customer assortment optimization without repeated offerings; items priced homogeneously across customers | 0.15              | Known                             |

of 0.09. Moreover, for the case that items are valued homogeneously across customers we improve this guarantee by making a slight modification to this algorithm. Table 1 summarizes the results of these algorithms.

In the special case where all of the patience levels equal 1, our problems reduce to the classical online stochastic matching/assortment problems where the firm only gets one chance to make an offering to each customer. Under stationary arrivals, the tight approximation ratio relative to the LP is $1 - 1/e$ in both of these cases (see Brubach et al. (2017) and Ma et al. (2018), respectively). Our work, and specifically our impossibility result, shows that both of these problems are substantially more challenging when there are multiple chances to interact with each customer. We derive a tight approximation ratio for the special case of online stochastic matching with timeouts, and the first constant-factor approximation ratios for the general assortment problem.

Moreover, this work studies heuristics to solve the MCDLP’s when repeated offerings are permitted and not allowed. To do so, we use the column generation framework and we show that for general choice models the column generation subproblem cannot be solved in polynomial-time, unless P=NP. However, for the special case of MNL choice model, when assortments shown to a customer can overlap the column generation subproblem can be solved efficiently, and if they cannot overlap then the column generation subproblem can be closely approximated. To do so, we develop a fully polynomial-time approximation scheme (FPTAS) for the column generation subproblem and show that this method results in an FPTAS for the MCDLP.
Lastly, in Section 6 we run simulations on the publicly accessible hotel data set by Bodea et al. (2009). To estimate customer choice model, we use the product availability and booking information and we consider a stochastic arrival of customers with identical arrival rates. There are four different rooms that can be offered to customers upon their arrival, those are, King room, Queen room, Suite and Two-double room. Furthermore, we try out different patience levels for the customers to observe their effects on the performance of the algorithms. Therefore, we have a multi-stage and multi-customer assortment problem where each room can be offered at multiple prices (those are, rack rate, discounted rate, etc.). On this data set, we compare the performance of our algorithms for the multi-stage and multi-customer assortment problem to various benchmarks. These simulations show that in different settings of the problem, in the two extremes where the number of rooms is very small compared to the number of arriving customers or when the there is an abundant number of rooms available to be offered, other benchmark heuristics might perform very well. However, in the regime that lies in between these two extremes, our algorithms outperform these benchmarks. This is significant as most practical cases lie in such regimes and our simulations show a strong performance of our algorithms for them.

1.2. Comparison of Model to Related Literature

We compare our modeling assumptions to those in the literature, first focusing on the aspect of multi-stage interactions with the same customer. Our model for sequential assortment offerings is inspired by Flores et al. (2019) and Liu et al. (2019a), where the customer leaves upon accepting an offered item. This model was originally used by Liu et al. (2019b) for healthcare scheduling systems, where the patient leaves upon accepting an appointment. By contrast, in other applications where the firm is selling a sequence of “add-ons”, the customer leaves upon declining an offer (Chen et al. 2016, Xu and Wang 2018). We use the former model because it naturally coincides with the online stochastic matching with timeouts model when the assortments are restricted to have size 1.

The works above focus specifically on the Multi-nomial Logit (MNL) choice model, and derive refined results using the structure of MNL. By contrast, we are the first to attack these multi-stage assortment problems using the generality of the MCDLP, and our results hold for the very large class of substitutable choice models. An assumption made in our work is that the number of stages a customer is willing to interact is known when she arrives; this is also assumed in the works above (and in some cases, assumed to always equal two). Other models have been proposed when a single customer views a variable number of pages; see (Gallego et al. 2018), and also (Wang and Sahin 2017) who models the “search cost” paid by the customer to view more pages.

Regarding the second aspect of our work—inventory-constrained assortment optimization, we provide the first constant-factor approximation guarantee in the multi-stage, multi-customer setting. Chen et al. (2016) have previously established a constant-factor guarantee for a firm selling
add-ons, but in their case there are only two stages and an assortment is offered in only one of the stages. We should mention that Lei et al. (2018) have recently studied an inventory-constrained assortment optimization problem where the item display order affects each customer’s choice.

Finally, we point out that in our model the next customer does not arrive until the previous customer has finished making her decisions across the stages. This is a modeling assumption which is also made throughout the previous literature (see Liu et al. (2019b), Chen et al. (2016)), as well as in online stochastic matching with timeouts. Nonetheless, performance should only improve if multiple customers are progressing through different stages at the same time, since the online algorithm has gained information about future customer arrivals. We believe that formally modeling this behavior would be an interesting future work.

2. Problem Description

An e-commerce platform is selling a set of \( n \) different items. Suppose there exists \( m \) customer types that can arrive to the platform. Each item \( i \) can have an initial inventory level \( b_i \) which is a non-negative integer. For the sake of simplicity, we assume that there is a single copy of each item, i.e., \( b_i = 1 \) for all \( i \in [n] \). This assumption is without loss of generality, as an item with multiple units of inventory can be split into separate items. Furthermore, for each customer type, item \( i \) has a selling revenue \( r_{ij} \geq 0 \). We consider a finite selling horizon, consisting of \( T \) discrete time-steps, over which there is no inventory replenishment.

During each time-step \( t = 1, \ldots, T \), a customer of one of the \( m \) types could arrive, we refer to whom as customer \( t \). The probability that customer \( t \) has type \( j \) is given by \( q_{tj} \), where \( \sum_j q_{tj} \leq 1 \); the inequality can be strict to model a time-step \( t \) which could have no customer arrival. Each type \( j \) is endowed with its own choice model, given by a choice function \( p_j(\cdot, \cdot) \) such that for every assortment \( S \) of items and every \( i \in S \),

\[
p_j(i, S) = \Pr[\text{a type-}j \text{ customer purchases item} \ i \ \text{when offered subset} \ S].
\] (1)

Note that \( \sum_{i \in S} p_j(i, S) \leq 1 \) for all types \( j \) and assortments \( S \); the inequality can be strict to model the fact that the customer could purchase nothing. We denote the probability of making no purchase by a customer of type \( j \) when assortment \( S \) is offered to her by \( p_j(0, S) \). We assume that assortment \( S \) must lie in some downward-closed family \( \mathcal{S} \), over which the probabilities (1) can be input in a computationally-efficient manner and satisfy a substitutability condition. All of

---

1 For a positive integer \( a \), the set \( \{1, \ldots, a\} \) is denoted by \([a]\).

2 We assume the more general case of type-dependent revenues for items. Of course our methods and formulations can be used for the cases that the selling revenue of each item is the same for all customer types.

3 Let index 0 refer to the “no purchase” option.
these are standard assumptions in the literature on inventory-constrained assortment optimization. Moreover, we assume that all random realizations (customer arrivals and purchase choices) are independent from each other. This includes the multiple purchase choices made by a customer, which is why we impose that the same customer cannot see the same item through different assortments.

The novel aspect of this work is that we allow for multiple assortments to be shown to the same customer. Specifically, each type \( j \) has a patience level \( \ell_j \), a positive integer denoting the number of assortments that the customer is willing to view before “timing out” and leaving. When a type \( j \) customer arrives, the platform shows her a sequence of up to \( \ell_j \) non-intersecting assortments, which she views in order, stopping once she either makes a purchase or reaches the end of the sequence. We refer to each time an assortment is offered to a customer by a stage. If \( \ell_j = 1 \) for all \( j \), then this is the classical setting where the platform has only one chance to sell to each customer.

2.1. Online Stochastic Matching with Timeouts

The online stochastic matching with timeouts problem considers a special case of the previous model. It makes the following simplifying assumptions:

1. \( q_{tj} = q_j \) for all \( t \) and \( j \), i.e. the customer arrival probabilities are stationary over time;
2. \( S = \{ S \subseteq [n] : |S| \leq 1 \} \), i.e. a customer can be shown only one item at a time.

Since \( |S| \leq 1 \), we can further simplify notation by letting \( p_{ij} \) denote \( p_j(i, \{i\}) \), the probability that a type \( j \) customer purchases item \( i \) when shown. The non-intersection property reduces to the constraint that a particular customer cannot be shown an item she previously rejected.

3. Tight Result for Online Stochastic Matching with Timeouts

In each time-step, a customer of one of the types \( 1, \ldots, m \) arrives. To a customer of type \( j \) we can offer at most \( \ell_j \) items. If the customer does not purchase any of these items, she loses interest and leaves the system. The goal is to offer sequences of items to arriving customers that maximize the total expected revenue. To measure the performance of an online algorithm for an instance of a problem, it is conventional to compare the online algorithm’s collected revenue with that of the offline optimal solution for that instance. Lemma 1 shows that the following linear program provides an upper-bound on the revenue of the offline optimal solution for any instance of the problem. Therefore, instead of comparing the total expected revenue of an online algorithm with that of the offline optimal solution, we make the comparison with the objective value of LP2. We refer to LP2 simply as the LP for the rest of this work.

\[
\max \sum_{j=1}^{m} Tq_j \sum_{i=1}^{n} r_{ij}p_{ij}x_{ij}
\]
\[
\sum_{j=1}^{m} T_j p_{ij} x_{ij} \leq 1 \quad \forall i = 1, \ldots, n \quad (2a)
\]
\[
\sum_{i=1}^{n} p_{ij} x_{ij} \leq 1 \quad \forall j = 1, \ldots, m \quad (2b)
\]
\[
\sum_{i=1}^{n} x_{ij} \leq \ell_j \quad \forall j = 1, \ldots, m \quad (2c)
\]
\[
0 \leq x_{ij} \leq 1 \quad \forall i = 1, \ldots, n; \forall j = 1, \ldots, m \quad (2d)
\]

where decision variable \( x_{ij} \) denotes the probability that item \( i \) is offered to a customer of type \( j \) and \( p_{ij} x_{ij} \) is the probability that item \( i \) is matched to a customer of type \( j \). Constraint (2a) ensures that no item is purchased more than once, its inventory. Constraint (2b) states that at most one item is sold to a visiting customer, and (2c) ensures that the number of items offered to a customer is no more than her patience limit. Let \( \text{OPT} \) be the optimal objective function of the LP and for an online algorithm let \( \mathbb{E}[\text{ALG}] \) denote the total expected revenue of that algorithm on the same instance of the problem. Our goal is to devise an online algorithm that achieves a constant approximation guarantee, formally defined below.

**Definition 1 (**\( \alpha \)-approximation Algorithm**). For a maximization problem we say that an algorithm has an approximation guarantee of \( \alpha \) if for any instance of the problem it provides a solution in polynomial-time whose total expected revenue, \( \mathbb{E}[\text{ALG}] \), is within an \( \alpha \) factor of the optimum, \( \text{OPT} \). In other words,

\[
\alpha \text{OPT} \leq \mathbb{E}[\text{ALG}] \leq \text{OPT}.
\]

Theorem 1 summarizes our main result for the online stochastic matching with timeouts problem.

**Theorem 1.** Suppose that \( \sum_{i=1}^{n} p_{ij} \leq 1 \) or \( \ell_j \geq n \) for all customer types \( j \). Then there is a polynomial-time algorithm whose expected revenue is at least \( (1 - \ln(2 - 1/e)) \cdot \text{OPT} \), which implies an approximation ratio of \( 1 - \ln(2 - 1/e) \approx 0.51 \).

In the rest of this section we discuss the algorithm and prove that it achieve this guarantee. The online stochastic matching with timeouts problem can be divided into two distinct subproblems: the **offline subproblem** that handles which items should be offered to a visiting customer, and the **online subproblem** that manages how series of customer arrivals should be handled in order to have a desired approximation ratio. To address these two subproblems, our algorithm consists of two steps: an **offline black-box** to address the offline subproblem and an **attenuation framework** to take care of the online subproblem. In a nutshell, the offline black-box decides which items to be offered to an arriving customer and the attenuation framework modifies this decision to provide the desired bounds on the performance of the overall algorithm. We will discuss these two steps in more details later in this section. With this intuition in mind we introduce the following definitions.
Definition 2. For each $t \in [T]$, item $i \in [n]$, and type $j \in [m]$ let us define the following events:

- **Type$_t(j)$**: the type of customer $t$ realizes to $j$;
- **Avail$_t(i)$**: item $i$ is still available at the start of time-step $t$;
- **Offer$_t(i,j)$**: the algorithm (pre-attenuation) intends to offer $i$ to customer $t$, who has type $j$;
- **True-offer$_t(i,j)$**: the algorithm (post-attenuation) offers item $i$ to customer $t$, whose type is $j$;
- **Accept$_t(i,j)$**: customer $t$, with type $j$, would have purchased item $i$ if truly offered (i.e., offered post-attenuation).

With these definitions, in the following lemma we establishes an upper-bound on the expected revenue of any algorithm for the problem, with the proof provided in Appendix A.

**Lemma 1.** For any instance of the online stochastic matching with timeouts problem, the total expected revenue of any algorithm is upper-bounded by the optimal value of the LP. The expectation is taken with respect to the purchase choice of customer types for each item as well as the random selections of the algorithm (in case it is a randomized algorithm).

As discussed earlier, we propose an offline black-box to address the offline problem, which will be later used in the algorithm for the overall problem. Suppose customer $t$ is of type $j$. We denote the set of items that are still available when this customer arrives and for which $x_{ij} > 0$ by $U^t_j$. Our offline black-box offers a subset of at most $\ell_j$ items $i \in U^t_j$ to customer $t$ such that each such item $i$ is offered with at least a certain probability. Lemma 2 discusses the black-box in details.

**Lemma 2 (Black-box Randomized Procedure).** Let $A$ be a set of coins. Each coin $i \in A$ can be flipped at most once and lands on “heads” independently with probability $p_i$. We can flip the coins in any (possibly randomized) order, and must stop once we get a “heads”, or have flipped $\ell$ coins, where $\ell$ is a positive integer.

Let $(x_i)_{i \in A}$ be any vector of weights in $[0,1]^{|A|}$ satisfying $\sum_{i \in A} p_i x_i \leq 1$ and $\sum_{i \in A} x_i \leq \ell$. Then there exists a randomized procedure for flipping the coins such that the probability of any coin $i$ being flipped, before the process is stopped, is at least

$$\frac{1 - e^{-w_i}}{w_i} \cdot x_i,$$

where $w_i = \frac{1}{1 - p_i} \sum_{i' \neq i} p_{i'} x_{i'}$ (or $w_i$ is understood to be 1 if $p_i = 1$) if $\sum_{i \in A} p_i \leq 1$ and $w_i = \frac{1}{1 - p_i x_i} \sum_{i' \neq i} p_{i'} x_{i'}$ (or $w_i$ is understood to be 1 if $p_i x_i = 1$) if $\ell \geq |A|$.

**Proof.** The elements of $(x_i)_{i \in A}$ are fractional, hence they do not clearly determine whether a coin $i$ should be flipped. We use a rounding procedure introduced by Gandhi et al. (2006) on $x_i$’s to make them integral, i.e., 0 or 1. If $x_i$ is rounded up to 1 then coin $i$ would be among the coins that can be flipped and otherwise it would not be. We refer to this rounding procedure by GKPS in the rest of this work. The following theorem states the main properties of the GKPS rounding and we refer the interested readers to (Gandhi et al. 2006) for a thorough discussion on it.
Theorem 2 (Gandhi et al. (2006)). Let \((z_i)_{i \in A}\) be any vector of weights in \([0,1]\). The GKPS algorithm is a randomized algorithm that in polynomial-time creates an integral solution \((Z_i)_{i \in A} \in \{0,1\}\) that guarantees the following three properties:

1. Marginal distribution: For each coin \(i\), \(Pr[Z_i = 1] = z_i\).
2. Degree preservation: \(\sum_{i \in A} Z_i \leq \lfloor \sum_{i \in A} z_i \rfloor \leq \ell\).
3. Negative correlation: For any subset of coins \(S\) and any \(b \in \{0,1\}\), \(Pr[\bigwedge_{i \in S}(Z_i = b)] \leq \prod_{i \in S} Pr[Z_i = b]\).

Using the GKPS rounding, we can round vector \((x_i)_{i \in A}\) to get \((X_i)_{i \in A}\). For each coin \(i\), \(X_i \in \{0,1\}\), satisfying the marginal distribution, degree preservation and negative correlation properties mentioned in Theorem 2. With the help of this rounding procedure, we first prove the bound for the case that \(\ell \geq |A|\) and then discuss the case \(\sum_{i \in A} p_i \leq 1\).

The case of \(\ell \geq |A|\): The randomized procedure we run works as the following: Firstly, the GKPS rounding procedure is used on \((x_i)_{i \in A}\), let \(\tilde{U}\) denote the set of coins rounded up by GKPS. Secondly, for each rounded coin \(i\), i.e., \(i \in \tilde{U}\), a random variable \(Y_i\) is picked from the interval [0, 1], IID and uniformly at random. The coins in \(\tilde{U}\) are then ordered independently according to the weights \(p_i x_i\) and random variables \(Y_i\), that is, in an increasing order of \(Y_i/(1-p_i x_i)\), and flipped accordingly until the process stops by seeing a heads or running out of the \(\ell\) flip chances, see Algorithm 1.

Algorithm 1 Black-box

**INPUT:** \(\ell\), \(A\), \(x_i\) and \(p_i\) for all \(i \in A\)

1. Apply the GKPS rounding to \((x_i)_{i \in A}\), let \(\tilde{U}\) be the set of coins that are rounded up by the GKPS process.
2. For each coin \(i \in \tilde{U}\) pick a number \(Y_i\) uniformly at random and IID from [0, 1].
3. Flip coins \(i \in \tilde{U}\) in an increasing order of \(\frac{Y_i}{1-p_i x_i}\) (if \(\sum_{i \in A} p_i \leq 1\)) and \(\frac{Y_i}{1-p_i x_i}\) (if \(\ell \geq |A|\)) until a “heads” comes or \(\ell\) coins are flipped.

Consider any coin \(i \in A\) and suppose it is rounded up by the GKPS process. Assuming \(Y_i = y\), a coin \(i'\) that has also passed through the GKPS process is flipped before \(i\) if \(\frac{Y_{i'}}{1-p_{i'} x_{i'}} \leq \frac{y}{1-p_i x_i}\). We first provide an upper-bound on the probability that coin \(i'\) is flipped before coin \(i\). To do this, we divide the problem into two cases: \(y \geq 2(1-p_i x_i)\), and otherwise, \(y < 2(1-p_i x_i)\). We begin with the former case. If \(y \geq 2(1-p_i x_i)\), then

\[
1 - \exp\left(-\frac{yp_{i'} x_{i'}}{1-p_{i'} x_{i'}}\right) = 1 - \exp\left(-\frac{2p_{i'} x_{i'}}{p_{i'} x_{i'}}\right) \geq (1-p_{i'} x_{i'}),
\]

where the last inequality uses the fact that \(1 - 2p_{i'} x_{i'} + 2p_{i'}^2 x_{i'}^2 \geq \exp(-2p_{i'} x_{i'})\), derived using the Taylor expansion of \(\exp(-2p_{i'} x_{i'})\). Moreover, since \(\sum_{i \in A} p_i x_i \leq 1\), we have that \((1-p_i x_i) \geq p_{i'} x_{i'}\),
which in addition to $y \geq 2(1 - p_i x_i)$ and $y \leq 1$ concludes that $p_i x_i' \leq \frac{1}{2}$. This, in combination with (4) implies that $\frac{1-\exp(-\frac{yp_i x_i'}{1-p_i x_i})}{p_i x_i'} \geq 1$, making $\frac{1-\exp(-\frac{yp_i x_i'}{1-p_i x_i})}{p_i x_i'}$ a potential upper-bound on the probability of coin $i'$ getting flipped before coin $i$. Now, consider the case that $y < 2(1 - p_i x_i)$. We have that

\[
1 - \exp(-\frac{yp_i x_i'}{1-p_i x_i}) = 1 - \exp(-\frac{yp_i x_i'}{1-p_i x_i}) \cdot \frac{y}{1-p_i x_i} \geq (1 - \frac{yp_i x_i'}{2(1-p_i x_i)}) \cdot \frac{y}{1-p_i x_i}
\]

(5)

where for (5) we used a similar Taylor expansion to the one discussed earlier and assumption $y < 2(1 - p_i x_i)$ was used in (6). Finally, recall that coin $i'$ is flipped before $i$ if $Y_i' \leq (1 - p_i x_i') \cdot \frac{y}{1-p_i x_i}$, where $Y_i'$ is picked uniformly at random in $[0,1]$. Therefore, the probability of flipping $i'$ before $i$ is at most $(1 - p_i x_i') \cdot \frac{y}{1-p_i x_i}$, which in combination with (6) makes $\frac{1-\exp(-\frac{yp_i x_i'}{1-p_i x_i})}{p_i x_i'}$ a valid upper-bound for the probability of coin $i'$ getting flipped before $i$.

We can now bound the probability that a coin $i$ is flipped before the process is stopped for the case that $\ell \geq |A|$. Note that a coin $i$ is flipped only if it passes through the GKPS rounding, which happens with probability $x_i$, by the marginal distribution property of GKPS. After the GKPS process, we flip each coin $i \in \mathcal{U}$, sequentially according to its $Y_i/(1-p_i x_i)$ value, as described in Algorithm 1, until a coin comes heads, happening with probability $p_i$ for each $i \in \mathcal{U}$. Observe that by the degree preservation property of the GKPS rounding, there are at most $\ell$ coins in $\mathcal{U}$. Thus, even if all the coins in $\mathcal{U}$ are flipped we have no more than $\ell$ total flips. The key property of Algorithm 1 is to flip each coin in $\mathcal{U}$ in a suitable random order so that $i$ is flipped with a probability at least $x_i \cdot (1 - e^{-w_i})/w_i$. Ranking coins based on their $Y_i/(1-p_i x_i)$ provides us with this property. We use $I_i^{',i}$ to denote the indicator variable that coin $i'$ was flipped before coin $i$ and we so far know that $\text{Pr}[I_i^{',i}|i, i' \in \mathcal{U}, Y_i = y] \leq \frac{1-\exp(-\frac{yp_i x_i'}{1-p_i x_i})}{p_i x_i'}$. Below we provide a lower-bound on the probability that coin $i$ is flipped conditioned on $i$ being rounded up by the GKPS process:

\[
\text{Pr}[i \text{ flipped}|i \in \mathcal{U}] \geq \text{Pr}\left[\bigcap_{i' \neq i} I_i^{',i} \text{ not heads before } i|i \in \mathcal{U}\right]
\]

(7)

\[
= \prod_{i' \neq i} \text{Pr}[i' \text{ not heads before } i|i \in \mathcal{U}]
\]

\[
= \int_0^1 \prod_{i' \neq i} \text{Pr}[i' \text{ not heads before } i|i \in \mathcal{U}, Y_i = y]dy
\]

\[
= \int_0^1 \prod_{i' \neq i} (1 - \text{Pr}[i' \text{ heads before } i|i \in \mathcal{U}, Y_i = y])dy
\]

\[
= \int_0^1 \prod_{i' \neq i} (1 - \text{Pr}[i' \in \mathcal{U}|i \in \mathcal{U}, Y_i = y]) \text{Pr}[I_i^{',i}|i, i' \in \mathcal{U}, Y_i = y] \text{Pr}[i' \text{ flips heads}|i, i' \in \mathcal{U}, Y_i = y, I_i^{',i}]dy
\]
\[ \Pr[x > y] \geq \int_0^1 \prod_{i' \neq i} \left( 1 - x_{i'} \right) \frac{1 - \exp\left(-\frac{y p_{i',x_{i'}}}{1-p_i x_i}\right)}{p_{i',x_{i'}}} \, dy \]  \quad (8)

\[
= \int_0^1 \prod_{i' \neq i} \left( 1 - \left( 1 - \exp\left(-\frac{y p_{i',x_{i'}}}{1-p_i x_i}\right)\right) \right) dy
\]

\[
= \int_0^1 \prod_{i' \neq i} \exp\left(-\frac{y p_{i',x_{i'}}}{1-p_i x_i}\right) dy
\]

\[
= \int_0^1 \exp\left(-\sum_{i' \neq i} \frac{y p_{i',x_{i'}}}{1-p_i x_i}\right) dy
\]

\[
= \frac{1}{\sum_{i' \neq i} \frac{p_{i',x_{i'}}}{1-p_i x_i}} (1 - \exp\left(-\sum_{i' \neq i} \frac{p_{i',x_{i'}}}{1-p_i x_i}\right)) = \frac{1 - \exp(-w_i)}{w_i}.
\]

For (7), we used the fact that as there is enough patience to flip all coins when \( \ell \geq |A| \), the probability that coins \( i', i \neq i \) are not flipped heads is independent. Inequality (8) uses \( \Pr[I^i,i'|i', i \in \hat{U}, Y_i = y] \leq \frac{1 - \exp(-\frac{y p_{i',x_{i'}}}{1-p_i x_i})}{p_{i',x_{i'}}} \) and \( \Pr[i' \in \hat{U}|i \in \hat{U}, Y_i = y] \leq \Pr[i' \in \hat{U}] = x_{i'} \) which is derived from independence of \( Y_i \) from whether \( i', i \in \hat{U} \) and, more importantly, the fact that \( \Pr[i' \in \hat{U}|i \in \hat{U}] \leq \Pr[i' \in \hat{U}] \). The latter is derived from the negative correlation property of the GKPS rounding as

\[
\Pr[i' \in \hat{U}|i \in \hat{U}] = \frac{\Pr[(i' \in \hat{U}) \cap (i \in \hat{U})]}{\Pr[i \in \hat{U}]} \leq \frac{\Pr[i' \in \hat{U}] \Pr[i \in \hat{U}]}{\Pr[i \in \hat{U}]} = \Pr[i' \in \hat{U}].
\]

Therefore,

\[
\Pr[i \text{ flipped}] = \Pr[i \text{ flipped}|i \in \hat{U}] \Pr[i \in \hat{U}] = \Pr[i \text{ flipped}|i \in \hat{U}] \cdot x_i \geq \frac{1 - \exp(-w_i)}{w_i} \cdot x_i,
\]

providing the desired bound for the case that \( \ell \geq |A| \).

The case of \( \sum_{i \in A} p_i \leq 1 \): This case is similar to the previous case with some small yet essential modifications. Similar to the former case, we first run the GKPS process on \((x_i)_{i \in A}\) and choose a random variable \( Y_i \), IID and uniformly at random for each coin that is rounded up, i.e., \( i \in \hat{U} \). This time, we rank coins in an increasing order of \( Y_i/(1-p_i) \) and flip them in this order until the process stops, see Algorithm 1. Therefore, assuming \( Y_i = y \), coin \( i' \) is flipped before coin \( i \) if \( \frac{y}{1-p_{i'}} \leq \frac{y}{1-p_i} \).

We consider the following two cases: \( y \geq 2(1-p_i) \), and otherwise, \( y < 2(1-p_i) \) and we begin with the former one: If \( y \geq 2(1-p_i) \), then

\[
\frac{1 - \exp\left(-\frac{y p_{i'}}{1-p_i}\right)}{p_{i'}} \geq \frac{1 - \exp\left(-2p_{i'}\right)}{p_{i'}} \geq 2(1-p_{i'}).
\]  \quad (9)

Assumption \( \sum_{i \in A} p_i \leq 1 \) implies \( p_i + p_{i'} \leq 1 \), which in addition to \( y \geq 2(1-p_i) \) and \( y \leq 1 \) gives \( p_{i'} \leq \frac{1}{2} \). This, in combination with (9) concludes that \( \frac{1 - \exp\left(-\frac{y p_{i'}}{1-p_i}\right)}{p_{i'}} \geq 1 \), making \( \frac{1 - \exp\left(-\frac{y p_{i'}}{1-p_i}\right)}{p_{i'}} \) a possible upper-bound on the probability that coin \( i' \) gets flipped before coin \( i \) for this case of the problem.

Now let us consider the case \( y < 2(1-p_i) \). Here,

\[
\frac{1 - \exp\left(-\frac{y p_{i'}}{1-p_i}\right)}{p_{i'}} \geq \left(1 - \frac{y p_{i'}}{2(1-p_i)} \right) \cdot \frac{y}{1-p_i}
\]  \quad (10)
\[ \geq (1 - p_i) \cdot \frac{y}{1 - p_i}, \quad (11) \]

where (10) uses a Taylor expansion similar to the one discussed for (5) and \( y < 2(1 - p_\ell) \) is used for (11). Since the probability of flipping \( i' \) before \( i \) is at most \((1 - p_i') \cdot \frac{y}{1 - p_i}\), we have \((1 - \exp(-\frac{yp_{i'}}{1 - p_i})) / p_i' \) is a valid upper-bound for the probability of the event of interest. We are now equipped to prove the lower-bound on the probability that \( i \) is flipped before the process stops.

\[
\Pr[i \text{ flipped}|i \in U] \geq \mathbb{E}_U \left[ \int_0^1 \prod_{i' \in \mathcal{U}\setminus\{i\}} (1 - \frac{1 - \exp(-\frac{yp_{i'}}{1 - p_i})}{p_i'}) dy|i \in \mathcal{U} \right] \\
= \mathbb{E}_U \left[ \int_0^1 \exp(- \sum_{i' \in \mathcal{U}\setminus\{i\}} \frac{yp_{i'}}{1 - p_i}) dy|i \in \mathcal{U} \right] \\
= \mathbb{E}_U \left[ \frac{1}{\sum_{i' \in \mathcal{U}\setminus\{i\}} p_{i'} / (1 - p_i)} (1 - \exp(- \sum_{i' \in \mathcal{U}\setminus\{i\}} \frac{p_{i'}}{1 - p_i})|i \in \mathcal{U}) \right] \\
\geq \frac{1}{\mathbb{E}_U \left[ \sum_{i' \in \mathcal{U}\setminus\{i\}} p_{i'} / (1 - p_i)|i \in \mathcal{U} \right]} (1 - \exp(- \mathbb{E}_U \left[ \sum_{i' \in \mathcal{U}\setminus\{i\}} \frac{p_{i'}}{1 - p_i}|i \in \mathcal{U} \right])) \\
\geq \frac{1}{\sum_{i' \neq i} p_{i'} x_{i'}} (1 - \exp(- \sum_{i' \neq i} \frac{p_{i'} x_{i'}}{1 - p_i})) \quad (14) \\
\geq \frac{1 - \exp(-w_i)}{w_i}.
\]

Every coin that is flipped before \( i \) has to come tails so that \( i \) gets a chance to get flipped before the process is stopped, this is captured by (12). Moreover, the expectation with respect to \( \mathcal{U} \) in (12) is due to the fact that only coins that are rounded up by GKPS, which is a random process, can get flipped. Inequality (13) uses Jensen’s inequality and the fact that function \((1 - \exp(-z))/z\) is convex, which will be later proved in Lemma 3. To show (14) holds, we prove and use the following two claims: \( \mathbb{E}_U \left[ \sum_{i' \in \mathcal{U}\setminus\{i\}} p_{i'} / (1 - p_i)|i \in \mathcal{U} \right] \leq \sum_{i' \neq i} p_{i'} x_{i'} / (1 - p_i) \) and, function \((1 - \exp(-z))/z\) is decreasing. To prove the former claim, observe that by the negative correlation and marginal distribution of the GKPS process, respectively, we know that \( \mathbb{E}_U [\mathbb{1}(i' \in \mathcal{U}\setminus\{i\})|i \in \mathcal{U}] \leq \mathbb{E}_U [\mathbb{1}(i' \in \mathcal{U}\setminus\{i\})] \leq x_{i'}, \) where \( \mathbb{1}(\cdot) \) is the indicator function. Thus, \( \mathbb{E}_U \left[ \sum_{i' \in \mathcal{U}\setminus\{i\}} p_{i'}|i \in \mathcal{U} \right] \leq \sum_{i' \neq i} p_{i'} x_{i'}, \) proving the former claim. Since \((1 - \exp(-z))/z\) is decreasing (see Lemma 3), replacing \( \mathbb{E}_U \left[ \sum_{i' \in \mathcal{U}\setminus\{i\}} p_{i'}|i \in \mathcal{U} \right] \) with its upper-bound would not increase (13). Subsequently, it can be concluded that

\[
\Pr[i \text{ flipped}] \geq \frac{1 - \exp(-w_i)}{w_i} \cdot x_i,
\]

for this case.

Finally, for both the case \( \ell \geq |A| \) and \( \sum_{i \in A} p_i \leq 1 \), in order for \((1 - \exp(-w_i))/w_i \cdot x_i\) to be a valid lower-bound on the probability of coin \( i \) getting flipped before the process stops it is required.
that \((1 - \exp(-w_i))/w_i \cdot x_i \leq 1\). As \(x_i \leq 1\), it is sufficient to have \((1 - \exp(-w_i))/w_i \leq 1\). Defining \(f(w_i) = (1 - \exp(-w_i))/w_i\), Lemma 3 later shows that \(f(z) \leq 1\) for all \(z \in [0, 1]\) and addresses this concern. 

\[\square\]

**Corollary 1.** For both cases in Lemma 2 we have that \(w_i \leq 1\).

**Proof.** We begin with the case that \(\ell \geq |A|\). By Lemma 2, for this case \(w_i = \sum_{i' \neq i} \frac{p_i x_{i'}}{1 - p_i x_i}\). Recall that the weight vector \((x_i)_{i \in A}\) satisfies \(\sum_{i \in A} p_i x_i \leq 1\). In particular, the weight vector satisfies \(p_i x_i + \sum_{i' \neq i} p_i' x_{i'} \leq 1\), implying that \(w_i = \sum_{i' \neq i} \frac{p_i x_{i'}}{1 - p_i x_i} \leq 1\). For the second case, since \(\sum_{i \in A} p_i \leq 1\), it holds that \(\sum_{i' \neq i} \frac{p_i x_{i'}}{1 - p_i x_i} \leq 1\). Moreover, \((x_i)_{i \in A} \in [0, 1]^A\), thus \(w_i = \sum_{i' \neq i} \frac{p_i x_{i'}}{1 - p_i x_i} \leq \sum_{i' \neq i} \frac{p_i x_{i'}}{1 - p_i} \leq 1\). 

**Remark 1.** In Lemma 2 we showed that \(w_i^{(1)} = \sum_{i' \neq i} p_i' x_{i'}/(1 - p_i x_i)\) and \(w_i^{(2)} = \sum_{i' \neq i} p_i x_{i'}/(1 - p_i)\) for the cases \(\ell \geq |A|\) and \(\sum_{i \in A} p_i \leq 1\), respectively. It can be easily seen that \(w_i^{(1)} \leq w_i^{(2)}\), which combined with the fact that \(f(z) = (1 - \exp(-z))/z\) is a decreasing function (see Lemma 3) implies that \(f(w_i^{(1)}) \geq f(w_i^{(1)})\). In other words, Lemma 2 provides a tighter lower-bound on the probability of any coin \(i \in A\) being flipped before the process is stopped for the case \(\ell \geq |A|\) compared to the case \(\sum_{i \in A} p_i \leq 1\).

**Remark 2.** When we use Lemma 2, \(A\) will be a random set, hence \(w_i\) will be a random variable. The attenuation framework of Brubach et al. (2017) uses a black-box randomized procedure where the guarantee on a coin \(i\) being flipped depends on the random variable \(\tilde{w}_i = \sum_{i' \neq i \in A} p_i' x_{i'}\) instead. Their guarantee is \((1 - \tilde{w}_i / 2) \cdot x_i\), which is a weaker guarantee than ours in that \(1 - \tilde{w}_i / 2 \leq (1 - e^{-\tilde{w}_i}) / \tilde{w}_i\). Therefore, we modify their attenuation framework using our random variable \(w_i\), which leads to our tight approximation ratio.

**Remark 3.** Unlike the case of \(\sum_{i \in A} p_i \leq 1\), if \(\sum_{i \in A} p_i > 1\) then ranking coins based on solely \(p_i\)'s (e.g., ranking in an increasing order of \(Y_i/(1 - p_i)\)) would not attain a lower-bound similar to that of Lemma 2 on the probability of coin \(i\) getting flipped before the process gets stopped. We show this through a counter-example. Suppose \(\ell = 2\), \(|A| = 3\) and \(p_1 = 1 - \epsilon, p_2 = 0, p_3 = 1 - \epsilon\) and \(x_1 = 1, x_2 = 1 - \epsilon, x_3 = \epsilon\) for some small \(0.5 > \epsilon > 0\). Observe that vector \((x_i)_{i \in A}\) satisfies \(\sum_{i \in A} p_i x_i \leq 1\) and \(\sum_{i \in A} x_i \leq \ell\), as stated in Lemma 2. By the marginal distribution property of the GKPS rounding algorithm, coin 1 is in \(\hat{U}\) with probability \(x_1 = 1\) (i.e., coin 1 is always in \(\hat{U}\)). Moreover, by the degree preservation property of GKPS, at most \(\ell\) coins can be in \(\hat{U}\) (2 coins in this example). We are interested in finding the probability that coin 3 gets flipped before the stop of the process. Conditioned on coin 3 in \(\hat{U}\), we have that \(\hat{U} = \{1, 3\}\); therefore, \(\sum_{i \in \hat{U}} p_i = p_1 + p_3 = 2 - 2\epsilon > 1\). If we order the coins in \(\hat{U}\) solely based on \(p_i\)'s then, since \(p_1 = p_3\), coins 1 and 3 have the same priority in expectation, resulting in \(\Pr[3 \text{ flipped} | 3 \in \hat{U}] = 0.5 < 1 - 1/\epsilon \leq (1 - \exp(w_3))/w_3\) for \(w_3 \in [0, 1]\), as our offline black-box would guarantee by Lemma 2.
The offline subproblem was addressed by the offline black-box algorithm, discussed in Lemma 2. In the rest of this section, we study the online subproblem. As discussed earlier in this section, attenuation framework is used to address the online subproblem. Brubach et al. (2017) provide three different attenuation frameworks: the edge-attenuation, the vertex-attenuation and the combined edge and vertex-attenuation. The last one is the most powerful one, giving Brubach et al. (2017) the approximation ratio of 0.46 for the online stochastic matching with timeouts problem. Thus, we only study the performance of our online algorithm when the combined edge and vertex-attenuation framework (called the attenuation framework henceforth) is used alongside the offline black-box. Before discussing this framework, in the following lemma let us review a few properties of function \( f(z) = \frac{1}{z}(1 - \exp(-z)) \) introduced in the proof of Lemma 2. Lemma 3 is proven in Appendix A.

**Lemma 3.** \( f(z) = \frac{1}{z}(1 - \exp(-z)) \) is a decreasing and convex function with finitely bounded first derivatives in \([0,1]\). Moreover, for any \( z \in [0,1] \), we have \( f(z) \leq \lim_{z \to 0} f(z) \leq 1 \).

The attenuation framework is the combination of the edge-attenuation and the vertex-attenuation frameworks, hence its role is twofold. Lemma 2 established lower-bounds on the probability of each item getting offered to the customers. In reality, some items might be offered to some customers with probabilities higher than these lower-bounds. These over-performing items cause other items to have poor performances, lowering the approximation ratio of the algorithm. Edge-attenuation solves this issue by weakening the performance of the over-performing items through forcing the probability that each item gets offered to be equal to the bounds established in Lemma 2. This allows the lowest performing items, which determine the approximation ratio of the algorithm, to perform better and in turn improving the approximation ratio.

Furthermore, as more customers visit the platform, more items get purchased, thus, less items are available. This makes \( w_i \)'s in Lemma 2 smaller as time goes on. Since \( f(w_i) \) is a decreasing function, the probability of offering an available item \( i \) increases with time. Thus, the available items are offered with higher probabilities as time goes on, resulting in smaller availability probabilities of items with time. Vertex-attenuation allows us to take advantage of the decrease of the availability probability of items and ensures that, at the beginning of each time-step, the probability of items being available is uniform and decreasing over time. This leads to a higher probability of offering the available items, improving the performance of the low-performing items and ameliorating the algorithm’s approximation ratio.

We remark that for the attenuation framework, we assume that it is possible to obtain an accurate estimate of probabilities of interest via Monte-Carlo simulations. This relies on the results by Adamczyk et al. (2015) and Ma (2014) that demonstrate that boundedness of the first derivative of purchase probabilities results in the boundedness of the accumulated error of the simulations.
For an item \(i\) and a customer of type \(j\), the purchase probability is \(f(w_i) \cdot x^*_ij \cdot p_{ij}\) and Lemma 3 shows that the derivative of \(f(w_i)\) is finitely bounded. Therefore, the simulation errors can be manipulated such that the ratio of the total expected revenue of our algorithm to the optimum of the LP only loses an additive factor of \(\epsilon = o(1)^4\).

Algorithm 2 gathers the offline black-box (discussed in Lemma 2) and the attenuation framework, and Theorem 3 discusses the attenuation framework and its properties. Before that, we introduce the following notation that comes handy in the algorithm.

**Definition 3.** For \(t \in [T]\), let \(\gamma_1 = 1\) and \(\gamma_t\) be recursively computable by \(\gamma_t = \gamma_{t-1} - \frac{1-e^{-\gamma_{t-1}}}{T}\).

**Algorithm 2 Online Algorithm**

1: Before any customers arrive, solve the LP to get \(x^*\).
2: **for** time-steps \(t = 1, \ldots, T\) **do**
   3:   Let customer \(t\) be of type \(j\), \(U^t_j\) be the set of available items such that for each \(i \in U^t_j\) we have \(x^*_ij > 0\), and \(x^* (U^t_j)\) be the elements of \(x^*\) that correspond to type \(j\) and items in \(U^t_j\).
   4:   Run the offline black-box (Algorithm 1) on \(x^*(U^t_j)\) as a subroutine and apply edge-attenuation to ensure each item \(i \in U^t_j\) is offered to customer \(t\) with probability \(x^*_ij \cdot (1 - e^{-\gamma_t})/\gamma_t\).
   5:   Apply vertex-attenuation to each item so that they are available with probability equal to \(\gamma_{t+1} = \gamma_t - (1 - \exp(-\gamma_t))/T\) at time-step \(t+1\).

**Theorem 3 (Modified Attenuation Framework).** Consider any time-step \(t \in [T]\). Suppose there exist attenuation factors \(a^\text{vertex}_t(i), a^\text{edge}_t(i, j) \in [0, 1]\) for all items \(i \in [n]\), customer types \(j \in [m]\), and time-steps \(t' < t\) on which Algorithm 2 can be run until the start of time-step \(t\), at which point for all items \(i\),

\[
\Pr[\text{Avail}_t(i)] = \gamma_t.
\]

Then there exist attenuation factors \(a^\text{vertex}_t(i), a^\text{edge}_t(i, j) \in [0, 1]\) for all \(i \in [n], j \in [m]\) on which Algorithm 2 can be run during time-step \(t\), so that for all items \(i\) and types \(j\),

\[
\Pr[\text{Accept}_t(i,j)] = (1 - e^{-\gamma_t})q_j p_{ij} x^*_ij;
\]

\[
\Pr[\text{Avail}_{t+1}(i)] = \gamma_{t+1}.
\]

The proof of Theorem 3 is provided in Appendix A.

\(o(1)\) is the set of all functions that are asymptotically smaller than constants; thus, \(\lim_{n \to \infty} o(1) = 0\).
Proof of Theorem 1 In the proof of Theorem 3 we showed that at time-step $t$, $\Pr[\text{Accept}_t(i, j)] = (1 - e^{-\gamma t})q_j x_{ij}^*$. Thus, for the total expected revenue of Algorithm 2, denoted by $\mathbb{E}[\text{ALG}]$, we have

$$\mathbb{E}[\text{ALG}] = \sum_{t=1}^{T} \sum_{j=1}^{m} \sum_{i=1}^{n} r_{ij} \Pr[\text{Accept}_t(i, j)] = \sum_{t=1}^{T} (1 - e^{-\gamma t}) \sum_{j=1}^{m} q_j \sum_{i=1}^{n} x_{ij}^* r_{ij} p_{ij}.$$ 

On the other hand,

$$\text{OPT} = \sum_{j=1}^{m} T q_j \sum_{i=1}^{n} x_{ij}^* r_{ij} p_{ij}.$$ 

Hence, to bound the approximation ratio of Algorithm 2 it suffices to bound the ratio between $\mathbb{E}[\text{ALG}]$ and $\text{OPT}$, i.e., finding a lower-bound for

$$\frac{\sum_{t=1}^{T} (1 - e^{-\gamma t})}{T}.$$ 

We take care of this in the following:

$$\frac{\sum_{t=1}^{T} (1 - e^{-\gamma t})}{T} = \frac{1}{T} \sum_{t=1}^{T} T (\gamma_{t+1} - \gamma_t) \quad (15)$$

$$= \gamma_1 - \gamma_T + 1 \quad (16)$$

$$\geq 1 - \ln(2 - \frac{1}{e}). \quad (17)$$

We use the recursive formula of $\gamma_t$ from (42) in (15), and in (16) the telescopic property of $\sum_{t=1}^{T} (\gamma_{t+1} - \gamma_t)$ was of use. Lastly, for (17) we use Lemma 4. Therefore, Algorithm 2 has an approximation ratio of at least $(1 - \ln(2 - \frac{1}{e})).$ □

In Lemma 4, we provide an upper-bound on $\gamma_{T+1}$, as required in the proof of Theorem 1. The proof of this lemma is available in Appendix A.

**Lemma 4.** For all $T \in \mathbb{N}$ we have $\gamma_{T+1} \leq h(1) = \ln(2 - \frac{1}{e})$, where $h(z) = \ln((e - 1) \exp(-z) + 1)$.

### 3.1. Tightness of Approximation Ratio

Brubach et al. (2017) showed that there are instances of the problem for which no online algorithm can provide an approximation ratio better than $1 - \frac{1}{e} \simeq 0.63$ for the online stochastic matching with timeouts problem. We improve this bound and show that in fact there exist instances of this problem for which no online algorithm can obtain an approximation ratio larger than $1 - \ln(2 - 1/e)$. This in addition to Theorem 1 shows that Algorithm 2 has a tight approximation ratio. The proof of the following lemma is provided in Appendix A.

$^5$ $\mathbb{N}$ denotes the set of positive integers.
Lemma 5. There exist instances of the online stochastic matching with timeouts problem for which no algorithm can have an approximation ratio larger than \(1 - \ln(2 - 1/e)\).

Remark 4. The example discussed in Lemma 5 satisfies both assumptions in Lemma 2, those are, \(\ell_j \geq n\) or \(\sum_{i=1}^{n} p_{ij} \leq 1\) for all types \(j\). Therefore, even in these two restricted cases of the problem no online algorithm can have an approximation ratio better than the one provided in Theorem 1 in the worst case. Furthermore, one might think that if some items have the same expected revenues then it would be possible to have a better approximation ratio for the problem. However, in the example provided in Lemma 5 the expected revenues of all items are the same, yet the best approximation ratio for any algorithm on that instance is no better than what was established by Theorem 1.

4. Result for General Assortments

In this section, we extend the online stochastic matching with timeouts problem to the multi-stage and multi-customer assortment offering setting. In particular, once a customer arrives, based on her type and the remaining inventory of items, the platform can decide to show her an assortment of items \(S \in \mathcal{S}\) as opposed to a single item. Here, \(\mathcal{S}\) denotes the set of all feasible assortments, set by the platform, that can be customized according to the constraints specific to that platform, such as, assortment size. We assume that \(\{0\} \in \mathcal{S}\), meaning that there exists the option of not offering any assortment to a customer at some stage. As discussed in Section 2, customer types are assumed to be associated with general choice models that determine the purchase probability of each item in an offered assortment. Moreover, we assume the substitutability assumption on choice models, which is defined formally below (see Golrezaei et al. (2014) for further discussions). Our goal is to design an online algorithm that maximizes the total revenue over the time horizon, while respecting the inventory constraints.

Definition 4 (Substitutability). The substitutability condition is a mild condition on choice models and for all customer types \(j\) and assortments \(S\) and items \(i \neq i'\) it implies that \(p_j(i, S) \geq p_j(i, S \cup \{i'\})\).

Before discussing the results for the multi-stage and multi-customer assortment optimization problem we remark that other than the substitutability assumption we impose no constraints on choice models and assortments that can be offered. In some application of this problem, only assortments of a limited size are permitted to be offered to customers. That is, there exists a constant \(k\) such that assortments with size larger than \(k\) are not allowed to be offered. Let \(\mathcal{S}^k\) denote the set of all assortments with size less than or equal to \(k\). Solving the assortment optimization problem in this case can be done by using the set of assortments \(\mathcal{S}^k\) instead of \(\mathcal{S}\). In fact it can be
easily seen that doing so all algorithms discussed in this section provide the same approximation guarantees they had on $S$.

In the rest of this section, we study the multi-stage and multi-customer assortment optimization problem in two cases. Section 4.1 studies the case that an item can be offered multiple times to a customer through different assortment; whereas in Section 4.2 an approximation algorithm for the case that items can be offered at most once to each customer is provided. This algorithm is then improved in Section 4.3 for the special case that items are valued homogeneously across customer types. Finally, Section 4.4 discusses the integrality gap of the MCDLP associated to the case that only a single-customer visits the platform and an item cannot be offered to her more than once.

4.1. General Assortments with Repeated Items Offerings

In this section we allow for the same item to be offered multiple times to the same customer. The customer makes choices with the same probabilities as if she had not seen the item before. This can be justified by Chernev (2006) and Chernev et al. (2012) that showed that the relative attractiveness of the items available in an assortment can affect the items’ purchase provability. More specifically, even though a customer is not inclined to purchase an item, e.g., a pair of headphones, in one stage of offerings, it is possible that she is willing to purchase it in a later stage if it is accompanied with trusted brands, e.g., specific cellphone brand.

Below we demonstrate the generalization of the LP to the case that a general assortment of items can be offered to a customer at each stage. Observe that this formulation does not avoid multiple offerings of an item to a customer through different assortments. That is, if an assortment $S_1 \ni i$ is first offered to a customer of type $j$ and no item of it, including item $i$, is purchased by that customer then $i$ can be offered to the customer through another assortment $S_2 \ni i$.

$$\max \sum_{j=1}^{m} T_{q_j} \sum_{S \in S} x_j(S) \sum_{i \in S} r_{ij} p_j(i, S)$$

$$\sum_{j=1}^{m} T_{q_j} \sum_{S \in S: S \ni i} x_j(S) p_j(i, S) \leq 1 \quad \forall i = 1, \ldots, n \quad (18a)$$

$$\sum_{S \in S} x_j(S) \sum_{i \in S} p_j(i, S) \leq 1 \quad \forall j = 1, \ldots, m \quad (18b)$$

$$\sum_{S \in S} x_j(S) \leq \ell_j \quad \forall j = 1, \ldots, m \quad (18c)$$

$$0 \leq x_j(S) \leq 1 \quad \forall j = 1, \ldots, m; \forall S \in S \quad (18d)$$

We refer to the above LP as the MCDLP-R, as it is a multi-stage choice-based deterministic LP that allows repeated offerings of items to customers. In this section, let $\text{OPT}$ denote the optimal value of the MCDLP-R. With a proof similar to that of Lemma 1 the following lemma can be shown about $\text{OPT}$.
Lemma 6. The optimal value of the MCDLP-R provides an upper-bound on the total expected revenue of any algorithm for any instance of the multi-stage and multi-customer assortment optimization problem when an item can be presented multiple times to a customer.

We show that a modification of Algorithms 1 and 2 results in a 0.51 approximation guarantee when items can be offered more than once to a customer.

Theorem 4. Suppose that \( \sum_{S \in \mathcal{S}} \sum_{i \in S} p_j(i, S) \leq 1 \) or \( \ell_j \geq |\mathcal{S}| \) for all customer types \( j \). Then there is a polynomial-time algorithm whose expected revenue is at least \( (1 - \ln(2 - 1/e)) \cdot \text{OPT} \), which implies an approximation ratio of \( 1 - \ln(2 - 1/e) \approx 0.51 \) for the multi-stage and multi-customer assortment optimization with the platform is allowed to show repeated items to customers.

The proof of Theorem 4 as well as the supplementary lemmas are provided in Appendix B.1.

4.2. General Assortments with No Repeated Items Offerings

In this section, we study the multi-stage and multi-customer optimization problem when items cannot be offered multiple times to customers through different assortments. This is a reasonable assumption in the settings where if a customer is not interested in purchasing an item, then changing the other items accompanying it would not change the decision of the customer.

Similar to the assortment optimization case studied in Section 4.1, we use a linear program to provide an upper-bound on the total expected revenue of the offline optimal solution when items cannot be offered repeatedly. The linear program below is the extension of the MCDLP-R, where Compared to the MCDLP-R here we have a new constraint, (19d), to ensure that the assortments shown to each customer are non-overlapping. This is to capture the assumption that a customer’s disinterest in an item would not get changed by the combination of the items that accompany it.

We refer to LP19 as the MCDLP-NR as it does not repeatedly offer items to customers.

\[
\begin{align*}
\max & \sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \in \mathcal{S}} x_j(S) \sum_{i \in S} r_{ij} p_j(i, S) \\
\text{s.t.} & \sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \in \mathcal{S}: S \ni i} x_j(S) p_j(i, S) \leq 1 \quad \forall i = 1, \ldots, n \quad (19a) \\
& \sum_{S \in \mathcal{S}} x_j(S) \sum_{i \in S} p_j(i, S) \leq 1 \quad \forall j = 1, \ldots, m \quad (19b) \\
& \sum_{S \in \mathcal{S}} x_j(S) \leq \ell_j \quad \forall j = 1, \ldots, m \quad (19c) \\
& \sum_{S \ni i} x_j(S) \leq 1 \quad \forall i = 1, \ldots, n; \forall j = 1, \ldots, m \quad (19d) \\
& x_j(S) \geq 0 \quad \forall j = 1, \ldots, m; \forall S \in \mathcal{S} \quad (19e)
\end{align*}
\]

This section studies the case that revenue of an item can be different for different customer types and Section 4.3 discusses the problem when items have homogeneous revenues across different

Electronic copy available at: https://ssrn.com/abstract=3443109
customer types. For the sake of this section, we impose an additional technical assumption on the customers arrival rates, that is, arrival rates are integral, meaning the expected number of arrivals of each type is integral. This is not a strict assumptions and has been used by Bansal et al. (2012) and Adamczyk et al. (2015) previously. Due to the integrality of arrival rates of all types, without loss of generality, one can always split a customer type in such a way that arrival rates of each of these new types is unit, i.e., $Tq_j = 1$ for all customer types $j$. Thus, we assume $T = m$, $q_j = 1/m$ for all $j$, where $m$ is the number of types that might arrive. This assumption allows to cancel $T$ and $q_j$ out and omit them from the objective function and constraint (19a) of the MCDLP-NR.

Let the optimal objective function of the MCDLP-NR be denoted by $OPT$. In the following, we provide an algorithm for the multi-stage and multi-customer assortment optimization with no repeated item offerings problem and show that it achieves a constant approximation ratio. Algorithm 3 is a modification of the algorithm designed by Bansal et al. (2012) which has a constant approximation ratio for the case of single item offerings, as opposed to assortments. We modify the algorithm to offer assortments and ensure that a customer would not see an item multiple times through different assortments. Theorem 5 discusses the approximation ratio of Algorithm 3, the proof of which is later provided in this section.

**Algorithm 3** Approximation Algorithm for General Assortments with No Repeated Offerings
1: Before any customer arrives, solve the MCDLP to get values $x^*$.
2: for time-steps $t = 1, \ldots, T$ do
3: Let the arriving customer be of type $j$.
4: if customer $t$ is first customer of type $j$ arriving then
5: Let $\pi$ be a uniformly at random order of assortments $S \in S$.
6: while less than $\ell_j$ offers are made and customer $t$ has not purchased any item do
7: One by one, offer each assortment $S$ along $\pi$ (after removing its sold-out items and items previously seen by the customer) independently with probability $x_j^*(S)/\alpha$.
8: else
9: Do not offer any assortments to the customer.

**Theorem 5.** There is a polynomial-time algorithm whose expected revenue is at least $0.093 \cdot OPT$, implying an approximation ratio of $0.093$ for the multi-stage and multi-customer assortment optimization problem.

Lemma 1 can be extended to establish an upper-bound on the total expected revenue of any algorithm for the multi-stage and multi-customer assortment optimization problem in Lemma 7. The proof of this Lemma is available in Appendix B.2.
Lemma 7. For any instance of the multi-stage and multi-customer assortment optimization problem, the expected revenue of any algorithm is upper-bounded by $\text{OPT}$.

Before analyzing the performance of Algorithm 3, let us define a few notations. Let $t_j$ denote the time-step in which the first customer of type $j$ arrives, and if no such customer arrives let $t_j = \infty$. Thus, if a customer of type $j$ arrives, then $t_j$ has a value between 1 and $T$; otherwise, it is $\infty$. We also use $t_j$ to refer to the first customer of type $j$ arriving to the platform, if any. Furthermore, we use $t_j^S$ to denote the stage at which assortment $S$ is offered to customer $t_j$, where if $S$ is offered to customer $t_j$ then $t_j^S$ has a value between 1 and $\ell_j$; otherwise, it is $\infty$. Below we introduce a few events, some are special or generalized cases of the events introduced in Definition 2.

Definition 5. For each $i \in [n]$, $j \in [m]$, and $S \in \mathcal{S}$ let us define the following events:

- Type$(j)$: a customer of type $j$ arrives, i.e., $t_j < \infty$;
- Offer$(S, j)$: the algorithm offers assortment $S$ to customer $t$, who has type $j$;
- Offer$(i, (i, S), j)$: the algorithm offers assortment $S$ to customer $t$ that has type $j$, such that $i \in S$ and $i$ is not removed from $S$ by the algorithm;
- Accept$(S, j)$: customer $t$, with type $j$, purchases an item from assortment $S$;
- IMatch$(i)$: item $i$ is already matched before the first arrival of type $j$, i.e., before $t_j$;
- Seen$(i, j)$: item $i$ is already shown to customer $t_j$ before $S$ is offered to her, i.e., before $t_j^S$;
- Timeout$(S, j)$: customer $t_j$ is already timed out when assortment $S$ is offered to her;
- CMatch$(S, j)$: customer $t_j$ is already matched when assortment $S$ is offered to her.

Let $\mathbb{E}[\text{ALG}]$ denote total expected revenue of Algorithm 3. Lemma 8 establishes a lower-bound on $\mathbb{E}[\text{ALG}]$ that will be later used to prove Theorem 5.

Lemma 8. The total expected revenue of Algorithm 3 is at least $\left(1 - \frac{1}{e}\right) \left(1 - \frac{3}{2a} - \frac{e^2}{3a^2}\right) \cdot \text{OPT}$, that is $\mathbb{E}[\text{ALG}] \geq (1 - \frac{1}{e}) \left(1 - \frac{3}{2a} - \frac{e^2}{3a^2}\right) \cdot \text{OPT}$.

Proof. We can break the total expected revenue collected by Algorithm 3 into the sum of the revenues the algorithm collects from each customer $t_j$. In other words, we have $\mathbb{E}[\text{ALG}] = \mathbb{E} \left[ \sum_{j=1}^{m} R_j \right]$, where $R_j$ denotes the revenue collected from $t_j$. This is due to the fact that Algorithm 3 offers assortments and so collects revenue only from the first customer of each type arriving. Similarly, let $R_j^i$ be the part of $R_j$ that comes from purchasing item $i$ by customer $t_j$, if any. In the following we focus on bounding $\mathbb{E}[R_j^i]$.

The customer arrival rates are uniformly $1/m$ and the number of time-steps is $T = m$; therefore, $\Pr[\text{Type}(j)] = 1 - \left(1 - \frac{1}{m}\right)^m \geq 1 - \frac{1}{e}$. Thus, $\mathbb{E}[R_j] \geq \mathbb{E}[R_j | \text{Type}(j)] \cdot (1 - \frac{1}{e})$. In the following we bound $\mathbb{E}[R_j | \text{Type}(j)]$, hence it is safe to assume that a customer of type $j$ arrives. Recall that Algorithm 3 removes the already seen and already matched items from assortments before offering them. Therefore, even if an item $i$ is in an assortment $S$, it is not necessarily shown to a customer.
when $S$ is offered to her. Removing the already seen and matched items without negatively affecting the purchase probabilities of other items in that assortment is made possible by the substitutability assumption. This assumption states that removing an item from an assortment does not decrease the purchase probabilities of the other items in that assortment, i.e., for each customer type $j$, assortment $S$ and item $i \neq i'$ we have $p_j(i, S) \leq p_j(i, S \setminus \{i'\})$. We establish a lower-bound on the probability of $\text{Offer}_{t_j}((i, S), j)$ that will be later useful to lower-bound $E[R_j|\text{Type}(j)]$.

$$Pr[\text{Offer}_{t_j}((i, S), j)|\text{Type}(j)] = 1 - Pr[\text{IMatch}_{t_j}(i) \cup \text{Seen}_S(i, j) \cup \text{Timeout}_S(j) \cup \text{CMatch}_S(j)|\text{Type}(j)] \frac{x_j^*(S)}{\alpha} \geq 1 - \left(Pr[\text{IMatch}_{t_j}(i)|\text{Type}(j)] - Pr[\text{Seen}_S(i, j)|\text{Type}(j)] - Pr[\text{Timeout}_S(j) \cup \text{CMatch}_S(j)|\text{Type}(j)]\right) \frac{x_j^*(S)}{\alpha}.$$ 

By Lemma 9 we have $Pr[\text{IMatch}_{t_j}(i)|\text{Type}(j)] \leq \frac{1}{2\alpha}$. Lemma 10 shows that $Pr[\text{Seen}_S(i, j)|\text{Type}(j)] \leq \frac{1}{2\alpha}$. Finally, Lemma 11 proves that $Pr[\text{Timeout}_S(j) \cup \text{CMatch}_S(j)|\text{Type}(j)] \leq \frac{1}{2\alpha} + \frac{2}{3\alpha^2}$. Therefore,

$$Pr[\text{Offer}_{t_j}((i, S), j)|\text{Type}(j)] \geq \frac{1}{\alpha}(1 - \frac{3}{2\alpha} - \frac{2}{3\alpha^2})x_j^*(S),$$

resulting in

$$E[R_j^*|\text{Type}(j)] = \sum_{S \ni i} r_{ij}p_j(i, S) Pr[\text{Offer}_{t_j}((i, S), j)|\text{Type}(j)] \geq \frac{1}{\alpha}(1 - \frac{3}{2\alpha} - \frac{2}{3\alpha^2}) \sum_{S \ni i} x_j^*(S)r_{ij}p_j(i, S).$$

Since $E[R_j|\text{Type}(j)] = \sum_{i=1}^n E[R_j^*|\text{Type}(j)]$, we have

$$E[R_j|\text{Type}(j)] \geq \frac{1}{\alpha}(1 - \frac{3}{2\alpha} - \frac{2}{3\alpha^2}) \sum_{i=1}^n \sum_{S \ni i} x_j^*(S)r_{ij}p_j(i, S).$$

Moreover, we showed that $Pr[\text{Type}(j)] \geq 1 - \frac{1}{e}$, which gives us $E[R_j] \geq (1 - \frac{1}{e})\frac{1}{\alpha}(1 - \frac{3}{2\alpha} - \frac{2}{3\alpha^2}) \sum_{i=1}^n \sum_{S \ni i} x_j^*(S)r_{ij}p_j(i, S)$. Therefore,

$$E[\text{ALG}] = \sum_{j=1}^m E[R_j] \geq (1 - \frac{1}{e})\frac{1}{\alpha}(1 - \frac{3}{2\alpha} - \frac{2}{3\alpha^2}) \cdot \text{OPT}.$$ 

□

In the following we discuss Lemmas 9, 10 and 11 that were used in the proof of Lemma 8.

**Lemma 9.** For any item $i$ and customer type $j$, $Pr[\text{IMatch}_{t_j}(i)|\text{Type}(j)] \leq \frac{1}{2\alpha}$.

**Proof.** For any customer type $j' \neq j$, let $I_{j'}$ denote the indicator variable that a customer of type $j'$ appeared before the first arrival of a customer of type $j$, that is $t_{j'} < t_j$. Thus,

$$Pr[\text{IMatch}_{t_j}(i)|\text{Type}(j)] = \sum_{j' \neq j} Pr[I_{j'} \cap \text{Accept}_{t_j}(i, j')|\text{Type}(j)]$$

Electronic copy available at: https://ssrn.com/abstract=3443109
\[
= \sum_{j' \neq j} \Pr[I_{j'}^{t'} | \text{Type}(j)] \Pr[\text{Accept}_{t_{j'}^{i}}(i, j') | I_{j'}^{t'}, \text{Type}(j)] \\
\leq \sum_{j' \neq j} \Pr[I_{j'}^{t'} | \text{Type}(j)] \frac{\sum_{S \ni i} x_{j'}^{*}(S) p_{j'}^{*}(i, S)}{\alpha} \\
\leq \frac{1}{2} \sum_{j' \neq j} \frac{\sum_{S \ni i} x_{j'}^{*}(S) p_{j'}^{*}(i, S)}{\alpha} \\
\leq \frac{1}{2\alpha}
\]

An assortment \( S \) is offered to a customer of type \( j' \) with probability at most \( x_{j'}^{*}(S) \); hence, an item \( i \in S \) is purchased by a type \( j' \) customer with probability at most \( \frac{x_{j'}^{*}(S) p_{j'}^{*}(i, S)}{\alpha} \). Union bound on all the assortments that contain \( i \) results in (20). Moreover, (21) and (22) are due to the symmetry in customer arrivals (as \( q_{j} \) is the same for all types \( j \)) and constraint (19a) in the MCDLP, respectively.

□

**Lemma 10.** For any item \( i \) and assortment \( S \) that contains \( i \) and for any customer type \( j \), we have \( \Pr[\text{Seen}_{S}(i, j) | \text{Type}(j)] \leq \frac{1}{2\alpha} \).

**Proof.** Item \( i \in S \) is already seen by a customer if an assortment \( S' \ni i \) was offered to that customer before \( S \). More formally, for any assortment \( S' \neq S \), let \( I_{S'}^{t} \) denote the indicator variable that \( S' \) is ordered before \( S \) on \( \pi \). Given that a customer \( t_{j} \) arrives, the probability that \( t_{j} \) has already seen item \( i \) when \( S \ni i \) is offered to her is the sum of the probabilities that another assortment \( S' \ni i \) was offered to \( t_{j} \) before \( S \). By conditioning on \( S' \) appearing before \( S \) on \( \pi \) we have

\[
\Pr[\text{Seen}_{S}(i, j) | \text{Type}(j)] = \sum_{S' \ni i, S' \neq S} \Pr[I_{S'}^{t} | \text{Type}(j)] \Pr[\text{Offer}_{t_{j}}(S', j) | I_{S'}^{t}, \text{Type}(j)] \\
\leq \sum_{S' \ni i, S' \neq S} \Pr[I_{S'}^{t} | \text{Type}(j)] \frac{x_{j}^{*}(S')}{\alpha} \\
\leq \frac{1}{2} \sum_{S' \ni i, S' \neq S} \frac{x_{j}^{*}(S')}{\alpha} \\
\leq \frac{1}{2\alpha}
\]

where (23) follows as Algorithm 3 offers assortments to a customer independently from those that were previously offered to her, (24) is due to the symmetry in the assortment ordering (\( \pi \) is a uniformly at random ordering of assortments) and (25) holds by constraint (19d) in the MCDLP.

□

**Lemma 11.** For any assortment \( S \) and customer type \( j \), \( \Pr[\text{Timeout}_{S}(j) \cup \text{CMatch}_{S}(j) | \text{Type}(j)] \leq \frac{1}{2\alpha} + \frac{2}{3\alpha^2} \).
Proof. We divide the problem into two cases: (i) \( \ell_j = 1 \), and (ii) \( \ell_j \geq 2 \). Let us focus on case (i) first. If \( \ell_j = 1 \), then \( \text{CMatch}_S(j) \subseteq \text{Timeout}_S(j) \) as there is only one chance of offering assortments to the arrived customer; thus, \( \Pr[\text{Timeout}_S(j) \cup \text{CMatch}_S(j)|\text{Type}(j)] = \Pr[\text{Timeout}_S(j)|\text{Type}(j)] \).

Let \( U \) be the random variable indicating the number of assortments offered to customer \( t_j \) right before \( S \) is about to be offered to her.

\[
\mathbb{E}[U] = \sum_{S' \neq S} \Pr[I_{S', S} \cap \text{Offer}_{t_j}(S', j)]
\]

\[
\leq \sum_{S' \neq S} \Pr[I_{S', S}] \frac{x^*_j(S')}{\alpha}
\]

\[
= \sum_{S' \neq S} \frac{x^*_j(S')}{2\alpha}
\]

\[
\leq \frac{\ell_j}{2\alpha},
\]

where (27) comes from the fact that Algorithm 3 offers an assortment \( S' \) with probability at most \( \frac{x^*_j(S')}{\alpha} \). Inequality (28) is from constraint (19c) in the MCDLP. Using Markov’s inequality we have

\[
\Pr[\text{Timeout}_S(j)|\text{Type}(j)] = \Pr[U \geq 1] \leq \frac{\mathbb{E}[U]}{\ell_j} = \frac{1}{2\alpha}.
\]

Now, consider the case that \( \ell_j \geq 2 \). By union bound \( \Pr[\text{Timeout}_S(j) \cup \text{CMatch}_S(j)|\text{Type}(j)] \leq \Pr[\text{CMatch}_S(j)|\text{Type}(j)] + \Pr[\text{Timeout}_S(j)|\text{Type}(j)] \). We bound the two terms \( \Pr[\text{CMatch}_S(j)|\text{Type}(j)] \) and \( \Pr[\text{Timeout}_S(j)|\text{Type}(j)] \) separately, starting with the former. Bounding \( \Pr[\text{CMatch}_S(j)|\text{Type}(j)] \) is similar to the above discussion with some modifications. Let \( U \) be the random variable indicating the number of assortments of which \( t_j \) purchases any items right before \( S \) is about to be offered to her (assuming \( t_j \) can purchase items for any number of stages).

\[
\mathbb{E}[U] = \sum_{S' \neq S} \Pr[I_{S', S} \cap \text{Accept}_{t_j}(S', j)] \leq \sum_{S' \neq S} \Pr[I_{S', S}] \frac{x^*_j(S')}{\alpha} \sum_{i \in S'} p_j(i, S')
\]

\[
= \sum_{S' \neq S} \frac{x^*_j(S')}{2\alpha} \sum_{i \in S'} p_j(i, S')
\]

\[
\leq \frac{1}{2\alpha},
\]

where (29) is by union bound because if an assortment \( S' \) is offered to a customer of type \( j \), then the customer purchases an item of the assortment with probability at most \( \sum_{i \in S'} p_j(i, S') \). The last inequality is from constraint (19b) in the MCDLP. Using Markov’s inequality we have

\[
\Pr[\text{CMatch}_S(j)|\text{Type}(j)] = \Pr[U \geq 1] \leq \frac{\mathbb{E}[U]}{1} \leq \frac{1}{2\alpha}.
\]
We finish the proof with upper-bounding \( \Pr[\text{Timeout}_S(j) | \text{Type}(j)] \) when \( \ell_j \geq 2 \). Suppose \( \ell_j = a \),

\[
\Pr[\text{Timeout}_S(j) | \text{Type}(j)] \leq \sum_{S_1, \ldots, S_a \neq S} \Pr[(I^{S_1, S} \cap \ldots \cap I^{S_a, S}) \cap (\text{Offer}_{t_j}(S_1, j) \cap \ldots \cap \text{Offer}_{t_j}(S_a, j))] | \text{Type}(j)] \\
\leq \frac{1}{a!} \sum_{S_1, \ldots, S_a \neq S} \Pr[(I^{S_1, S} \cap \ldots \cap I^{S_a, S}) \cap (\text{Offer}_{t_j}(S_1, j) \cap \ldots \cap \text{Offer}_{t_j}(S_a, j))] | \text{Type}(j)] \\
\leq \frac{1}{a!} \sum_{S_1, \ldots, S_a \neq S} \Pr[I^{S_1, S} \cap \ldots \cap I^{S_a, S}] | \text{Type}(j)] \prod_{k=1}^{a} \frac{x_j^*(S_k)}{\alpha} \\
= \frac{1}{(a+1)!} \sum_{S_1, \ldots, S_a \neq S} \prod_{k=1}^{a} \frac{x_j^*(S_k)}{\alpha} \\
\leq \frac{1}{(a+1)!} \left( \sum_{S \in S} \frac{x_j^*(S)}{\alpha} \right)^a \leq \frac{1}{(a+1)!} \left( \frac{a}{\alpha} \right)^a. \tag{31}
\]

Note that the first summation is over unordered assortments, while it changes to ordered assortments in the next step, hence we have a factor of \( \frac{1}{a!} \). In the third inequality we used the way Algorithm 3 works that a considered assortment \( S_k \) is offered to \( t_j \) with probability \( \frac{x_j^*(S_k)}{\alpha} \). The next summation comes from the fact that assortments \((S_1, \ldots, S_a)\) and \( S \) appear uniformly at random on \( \pi \), and the last inequality is from constraint (19c) in the MCDLP. Finally, Bansal et al. (2012) showed that \( \frac{1}{(a+1)!} \left( \frac{a}{\alpha} \right)^a \leq \frac{2}{\alpha^a} \), proving that \( \Pr[\text{Timeout}_S(j) | \text{Type}(j)] \leq \frac{2}{\alpha^a} \) when \( \ell_j \geq 2 \). \( \square \)

Proof of Theorem 5. Lemmas 7 and 8, and setting \( \alpha = \frac{3 + \sqrt{7}}{2} \) proves the approximation ratio of 0.093 for Algorithm 3. \( \square \)

In the following theorem, we discuss a variant of the multi-stage multi-customer assortment optimization problem where patience levels are non-deterministic. As opposed to constant patience levels that the patience level of each customer type is known, here patience level of customers is only known in expectation. In other words after seeing each assortment, a type \( j \) customer decides to leave the platform with probability \( p_j^{\text{out}} \) or choose to see the next assortment with probability \( 1 - p_j^{\text{out}} \). Theorem 6 discusses the approximation ratio of Algorithm 3 for this case of the problem, with the proof provided in Appendix B.2.

**Theorem 6.** Algorithm 3 has an approximation ratio of 0.093 for the multi-stage and multi-customer assortment optimization problem with non-deterministic patience levels.

**Remark 5.** GKPS-based algorithms can have arbitrarily bad approximation ratios for the multi-stage and multi-customer assortment optimization problem when items cannot be offered repeatedly to customers. To see it consider the instance of the problem discussed in Section 4.4. In this instance of the problem there exists one customer type, \( M \) assortments and \( M(M-1)/2 \) items such that each item appears in exactly two assortments. Section 4.4 shows that the optimum is 1.
On the other hand, since any pair of assortments has a shared item, any GKPS-based algorithm can contain at most one assortment to respect LP constraint (19d). Therefore, the total expected revenue of any such algorithm is at most \( \sum_{i \in S} p(i, S) = \frac{2}{M(M-1)} \cdot (M-1) = \frac{2}{M} \) for the rounded up assortment \( S \). So, the approximation ratio of the algorithm is \( \frac{2}{M} \), which goes to zero as \( M \) goes to infinity.

4.3. General Assortments with Homogeneous Item Revenues Across Customers and No Repeated Offerings

In the previous section, we discussed a 0.09-approximation algorithm for the problem of offering assortments to customers, while refraining from showing an item multiple times to a single customer. For this algorithm to work, there is no restrictions on the revenue acquired from selling an item to different types of customers. In this section, we study a scenario that is more restricted yet practical in many settings, that is, the case when the revenue of an item \( i \) does not depend on the type of the customer it is presented to, that is, \( r_{ij} = r_i \) for all customer types \( j \). We show that a slight modification of Algorithm 3 has an improved performance for this case of the problem.

Note that here, we do not require the assumption that all customer types have the same stationary arrival rate of \( q_j = \frac{1}{m} \), where \( T = m \). In fact we show that even for the non-stationary and heterogeneous customer arrivals, the modified Algorithm 3 provides a 0.15 approximation guarantee for the problem. Thus, the linear program that provides an upper-bound on any algorithm for this case of the problem is similar to the MCDLP-NR with the slight change of \( r_{ij} = r_i \) for all types \( j \). This modified linear program is referred to as the MCDLP-NRS to highlight that no repeated items can be shown to a customer and items are valued the same across customer types. In this section \( \text{OPT} \) is used to denote the optimal objective value of the MCDLP-NRS. We use a slight modification of Algorithm 3, where assortments are offered to a customer \( t \), whether she is the first arrival of her type or not. We then use the MCDLP-NRS to prove the approximation guarantee of this algorithm in the following theorem, with the proof available in Appendix B.2.

**Theorem 7.** When item revenues are homogeneous across customer types, there is a polynomial-time algorithm whose expected revenue is at least \( 0.15 \cdot \text{OPT} \), implying an approximation ratio of 0.15 for this case of the multi-stage and multi-customer assortment optimization problem.

4.4. Integrality Gap for Single-Customer MCDLP-NRS

In this section we focus on the single-customer multi-stage assortment optimization problem, as studied in Liu et al. (2019a). In other words, here only a single customer visits the platform and the problem is what assortments of items and in what order should be offered to her. Consider the LP relaxation for this problem:

\[
\max \sum_{S \in \mathcal{S}} x(S) \sum_{i \in S} r_ip(i, S)
\]
Feldman and Segev (2019) provided a PTAS for the single-customer multi-stage assortment problem via dynamic programming, where they compared their algorithm’s total expected revenue with that of the optimal solution. In contrast, in this work we use the optimum of different MCDLP’s as upper-bounds on the total expected revenue of the optimal solutions of different versions of the problem. The following theorem shows that, when compared to the MCDLP-NRS, the multi-stage assortment optimization problem has an integrality gap of 0.527, even when a single customer visits the platform. The proof of this theorem can be found in Appendix B.2. Note that our proof of Theorem 7 shows that the integrality gap for a single customer is at worst 1/6.

**Theorem 8.** There exists an instance of the multi-stage assortment problem where the fraction of the optimal MCDLP-NRS value obtained by any multi-stage assortment algorithm is at most $1 - e^{-3/4} \approx 0.527$.

5. **Solving the MCDLP**

In Section 4, we studied different versions of the assortment problem and for each of them we showed that there exists an MCDLP that provides an upper-bound on the total expected revenue of any algorithm. We later used the solution to those MCDLP’s to design approximation algorithms for the different versions of the multi-stage and multi-customer assortment optimization problem.

The MCDLP’s has exponential number of decision variables, as there can be $2^n$ different assortments in $S$; thus, solving the MCDLP’s can take a very long time. Gallego et al. (2004) suggested the use of column generation technique to overcome this issue. The main idea of the column generation technique is to solve a reduced LP with only a limited number of assortments $S'$ and check the corresponding dual LP to check whether the exists any assortment in $S$ for which the resulted dual variables constitute a positive reduced cost. In that case, the assortment with positive reduced cost is then added to $S'$ and the process is iterated. Otherwise, the current solution is optimal and no further assortments are needed to be added to $S'$. In other words, decision variables are added to the MCDLP as needed, as opposed to having a decision variable for each assortment in $S$, which might run into long time-runs. In Appendix C we discuss the column generation algorithm and how its approximation ratio translates into that of the MCDLP-NR. Moreover, we provide an FPTAS for the column generation subproblem when the choice model is MNL.
6. Simulations on Hotel Data Set of Bodea et al. (2009)

In this section, we test our algorithms designed to address the assortment planning problem on the publicly accessible hotel data gathered by Bodea et al. (2009) and compare their performances with those of several benchmark algorithms. Based on the data, here we consider a more general setting of the assortment planning problem, that is, a multi-price multi-stage multi-customer assortment optimization problem. In this problem, an item \( i \) can be offered to a customer \( t \) of type \( j \) at different prices, \( r^k_{ij} \) and \( r^k'_{ij} \), which are called price \( k \) and \( k' \), respectively. The probability that a customer of type \( j \) purchases an item \( i \) offered as a part of an assortment \( S \) with price \( k \) can be different if this item was offered as a part of the same assortment with a different price \( k' \). Due to this additional dimension in the problem, the choice function is modified as

\[
p_j(i, k, S) = \Pr[\text{a type-} j \text{ customer purchases item } i \text{ when offered subset } S \text{ with price } k]. \tag{33}
\]

For clarity, we use the term *product* to refer to an (item, price) combination, e.g., product \( (i, k) \). Therefore, upon the arrival of a customer \( t \) with type \( j \), for any assortment \( S \) and product \( (i, k) \in S \), we are given the probability that the customer purchases product \( (i, k) \), that is, \( p_j(i, k, S) \). Having these probabilities, the goal is to determine which assortment \( S \) should be offered to customer \( t \) at each stage, that is, which products and at what prices among their permitted prices should be shown to the customer. We note that the assortments offered to a customer can be limited so that they lie in an arbitrary downward-closed family, for instance, one can constrain the assortments offered to a customer to not offer an item at different prices simultaneously. The following LP captures the problem when items can be offered at multiple prices, which we refer to as MMCDLP-NR to signify items can be offered at multiple prices and it is not allowed to offer repeated products to the same customer:

\[
\begin{align*}
\max & \sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \in \mathcal{S}} x_j(S) \sum_{(i,k) \in S} r^k_{ij} p_j(i, k, S) \\
\text{s.t.} & \sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \subseteq \mathcal{S} : S \ni (i,k)} x_j(S) p_j(i, k, S) \leq 1 \quad \forall i = 1, \ldots, n \tag{34a} \\
& \sum_{S \in \mathcal{S}} x_j(S) \sum_{(i,k) \in S} p_j(i, k, S) \leq 1 \quad \forall j = 1, \ldots, m \tag{34b} \\
& \sum_{S \in \mathcal{S}} x_j(S) \leq \ell_j \quad \forall j = 1, \ldots, m \tag{34c} \\
& \sum_{S \ni (i,k)} x_j(S) \leq 1 \quad \forall i = 1, \ldots, n; \forall j = 1, \ldots, m \tag{34d} \\
& x_j(S) \geq 0 \quad \forall j = 1, \ldots, m; \forall S \in \mathcal{S} \tag{34e}
\end{align*}
\]

Note that constraint (34d) is written so that it allows the same item to be offered to a customer at different prices in different offering stages. For the case of hotel room offering, this assumption
is sensible. However, it is possible to rewrite this constraint so that once an item is offered to a customer, it is not allowed to be offered to the same customer again at a different price. In this section we use the hotel data set to test the algorithms developed for the multi-price multi-stage multi-customer assortment problem in the setting where items are not allowed to be shown repeatedly to a customer through different offering stages, as studied in Section 4.

6.1. Experimental Setup

From the hotel data set collected by Bodea et al. (2009) we consider the occupancies taking place in the 5-week period from March 11th 2007 to April 15th 2007. The hotel rooms available to be offered to customers are merged into the 4 following categories: King rooms, Queen rooms, Suites and Two-double rooms. Different categories of rooms have their own distinct inventory of rooms. Moreover, there are two fare classes for each room, those are, regular rates and discounted rates. Hence there are 8 different combinations constituted by these 4 rooms categories and 2 fare classes. We refer to each of these 8 combinations as a *product*. Note that a customer might choose a room offered at its regular rate to its discounted rate as oftentimes higher fares are packaged with extra offers, such as free breakfast, higher speed internet, etc.

Furthermore, 1315 different customer types appear in the data set, which are set based on the 8 different features available in the data: (i) whether the booking was done on the hotel website, (ii) whether the booking was done via a travel agency, (iii) party size, (iv) membership level, (v) VIP level, (vi) number of days booking was done in advance, (vii) whether booking was done over the weekend, (viii) whether the customer checked-in over the weekend. Thus, these customer types capture the heterogeneity in customers’ preferences. For each customer type, the data set provides the assortment of products offered to the customer as well as the product picked by her. Note that the hotel data set was collected using a platform that offers assortments in a single stage. To make the data set compatible to the multi-stage setting, which is the focus of this paper, we synthetically attach patience levels to customers, as further explained in Section 6.3. Moreover, each occupancy date for a customer is considered as a separate instance of the problem. This simplifying assumption allows for the multi-price multi-stage multi-customer assortment optimization problem to be studied for each day separately without complications coming from new rooms becoming available when customers depart.

On the available 8 products and for each of these 1315 customer types we estimate a Multinomial Logit (MNL) choice model. The MNL choice model is useful for two reasons. Firstly, it has been assessed to perform reasonably well on this data set (van Ryzin and Vulcano 2015), and secondly, in Section 5 we showed that there is an FPTAS to solve MCDLP under MNL.

6 Among these, features (i), (ii), (vii) and (viii) are binary features, feature (iii) is at least 1, feature (iv) is a number from 0 to 3, feature (v) is a number from 0 to 2, and feature (vi) is a non-negative number.
Next, we discuss the limitations for our analysis on the hotel data set. First, the data set does not provide the number of available rooms in each room category. In the experiments we test a wide range of initial room availabilities to observe the effect of the starting room capacities on the performance of the algorithms. Moreover, The data set does not provide any information on the customers that do not make a purchase after an assortment of hotel rooms are offered to them. While this is a standard challenge in choice modeling, we address it by trying different assumptions on the weight of the no-purchase option in MNL model for each customer type. In general, setting larger weights for the no-purchase option causes the myopically revenue-maximizing assortments to offer rooms at their lower prices. This creates a nontrivial tradeoff between offering lower prices which maximizes immediate revenue, v.s. offering higher prices which conserve inventory. Therefore, in many of our experiments we set the no-purchase option to have a large weight.

Having the data set and the choice model, we are now ready to discuss a test instance. As mentioned earlier, an instance is defined for a single occupancy date to ensure the inventory of room does not replenish. Once a customer arrives, her features and patience level are revealed; therefore, her type and purchase probabilities are given. The goal is to show personalized assortments of products, which are (room,fare) combinations, in different stages to the customer until she purchases a product or runs out of patience. Below we summarize the test instances:

- The total number of customer types is \( m = 1315 \). To comply with the assumptions made in Section 4.2, we assume that the rate of customer arrivals for all customer types is the same and it is equal to \( 1/m \), i.e, the total number of customers that might arrive on a day is 1315.
- The total number of products is the same for all instances and is equal to 8.
- As shown in Table 2, rooms can be offered at two prices and they are identical for all instances. In the experiments, we manipulate the room prices and increase the high fares and add heterogeneity to room prices for different customer types to demonstrate the strengths and weaknesses of algorithms more clearly. These manipulations are explained in Section 6.3.
- The hotel data set only provides the relative inventories of the different categories of rooms; see Table 2. To address this, we consider different starting inventory of rooms. Similar to (Golrezaei et al. 2014, Ma and Simchi-Levi 2017), this is done through setting different loading factors, which is defined as the ratio of total arriving customers to the total initial inventory of rooms. Section 6.3 further discusses different loading factors used in the tests.

### Table 2 Room categories’ fares and inventory percentages

| Room Category | Low Fare | High Fare | Percentage of Rooms |
|---------------|----------|-----------|---------------------|
| King          | $307     | $361      | 52%                 |
| Queen         | $304     | $361      | 15%                 |
| Suite         | $384     | $496      | 13%                 |
| Two-Double    | $306     | $342      | 20%                 |
• The hotel data is collected by offering assortments to customers in a single stage. That is, all customers have a patience level of 1. To be able to account for different patience levels, we synthetically assign patience levels to different customer types.

Knowing the experiment setup, in the next section we discuss the different algorithms used on the data set.

6.2. Algorithms Compared

As mentioned earlier, we constrained the algorithms for the experiments to not offer repeated products to a customer in different stages. On different instances of the problem we compare 4 algorithms, listed below:

1. **Greedy:** At each stage, the algorithm myopically offers the assortment of available products which maximizes the immediate expected revenue, irrespective of inventory.

2. **Conservative:** At each stage, the algorithm offers rooms only at their high prices. This algorithm chooses the assortment of rooms from the rooms that are not sold out using a greedy algorithm that maximizes the immediate expected revenue.

3. **Algorithm 3:** This is the 9% approximation algorithm from Section 4.2, with the difference that here assortments are defined over products and not items (i.e., rooms in the experiments). Moreover, we tested different values for $\alpha$ and it was evident that for these experiments $\alpha = 1$ provides the best performance. In general, $\alpha = 1$ performs more greedily than $\alpha = \frac{3 + \sqrt{17}}{2}$, which is the optimal value of $\alpha$ from our approximation algorithm. The reason that $\alpha = 1$ works better in the experiments is the limited number of products that can be offered to customers.

4. **Modified Algorithm 3:** In Section 4.3, we discussed a modification of Algorithm 3 for cases that all customers have the same rewards for each item and showed that the approximation guarantee of this algorithm was 15%. As opposed to Algorithm 3, this modified algorithm offers assortments of items to customers even if it is not the first time a customer of that type has visited the platform. We test this algorithm on the hotel data set even when the room fares are synthetically manipulated so that rooms are offered at different prices to different customers. Similarly, $\alpha$ is set to $\alpha = 1$ here, as opposed to $\alpha = 3$ from our approximation algorithm.

There are two extremes to the problem. One extreme is when there exists an infinite initial room inventories and a limited number of customers arriving on the platform, i.e., a loading factor of 0. The other extreme is where the loading factor approaches infinity and there exists too many customers relative to rooms. It can be easily seen that when the loading factor is 0 the greedy algorithm is the optimal approach as it extracts the most expected revenue for each arriving customer. On the other hand, when the loading factor is $\infty$ the conservative algorithm has the best performance as it collects the maximum expected revenue from each unit of room inventory.
The problems becomes challenging when the loading factor lies in-between these two extremes, where most practical assortment planning problems also lie. In this region, our algorithms select the assortments to be offered to customers by balancing the revenue collected per unit of inventory and the revenue collected per customer.

6.3. Results

On every instance, we run the 4 algorithms discussed in Section 6.2 and consider the average revenues over the 35 runs. The performance of these algorithms is measured as a percentage of MMCDLP-NR upper-bound introduced in Section 6.

In these experiments, we synthetically introduce heterogeneity in room fares for different customer types. To do so, for each customer type we pick a random Gaussian number for each room fare, while ensuring the low fares are smaller than the high fares. More specifically, for each room category and fare, we draw a random number from a Gaussian distribution whose mean is the original room fare (see Table 2) and whose standard deviation is the square root of the original room fare. While different methods and parameters can be used to add this heterogeneity, we chose these parameters to be close to real world scenarios, that is, room fares are different for different customer types but not considerably different.

As discussed earlier, the number of customer types and the number customer arrivals are equal. This means that each customer type arrives once on the platform in expectation, as we assumed that all customer types have the same stationary rate of arrival of 1/m. Naturally, it does not mean that each customer type arrives exactly once on the platform. Furthermore, we assign different patience levels to customers to see how the four algorithms behave when they are allowed to offer assortments to customers in different number of stages. Moreover, in different settings, a platform might be interested in offering assortments of at most a certain size to avoid overwhelming customers or giving them too many choices. To observe how this affects the performance of algorithms, we also try different upper-bounds on the size of assortments that can be offered. Furthermore, we try out a range of loading factors from 1 to 7 to see its impact on different algorithms’ performances.

Lastly, we are also interested in seeing how the gap between the low and high fares of rooms might affect algorithms’ performances. To do this, we first scale the original high fares of rooms by a scale factor and then use that in the Gaussian distribution discussed earlier to introduce heterogeneity in high fares for rooms for different customer types. Moreover, as discussed earlier, the weight of no-purchase option in MNL model cannot be estimated from the data. Accordingly, we shift the mean utilities in the MNL model so that for each customer type the weights of the no-purchase option and the most preferable purchase option is equal. In the settings that the scale factor is greater than 1 (i.e., where greater fare differentiation is induced in the model), the mean
utility of the no-purchase option is also multiplied by the scale factor for each customer type. This is to ensure that the revenue-maximizing assortments still contain both high and low fares, hence, keeping the problem instances nontrivial.

Figure 1 depicts the performance of the 4 algorithms on the hotel data set for uniform patience levels of 1 to 4 and two different bounds on the size of allowed assortments, which are 1 and 4. Moreover, the scale factor is 2 here. Appendix D demonstrates the desirable performance of our algorithms on some additional problem settings on the hotel data set.
Figure 1 Continued: Performance of Greedy, Conservative, Algorithm 3 (i.e., the 9% algorithm) and Modified Algorithm 3 (i.e., the 15% algorithm) on hotel data set with heterogeneous room fares for different patience levels, sizes of permissible assortments for scale factor=2 over different loading factors.

Figure 1 shows that the performance of Algorithm 3 (i.e., the 9% algorithm) is not as good as other algorithms tested on the data. This was expected as Algorithm 3 offers assortments only to the first customer arrival of each type; whereas, this is not the case for the other three algorithms. Moreover, it can be seen that in some cases, for low loading factors the Greedy algorithm outperforms the other algorithms; on the other extreme, for large loading factors, the Conservative algorithm outperforms all the other methods. However, it is important to observe that in cases where the loading factor is in-between these two extremes the Modified Algorithm 3 (i.e., the 15% algorithm) has the best performance of all. This is significant as a low loading factor means that there is not many customer arrivals to the hotel compared to its number of rooms on any day. In such scenario, the best strategy would be to follow the Greedy algorithm for offering assortments of rooms to customers. On the other hand, a high loading factor means that the hotel gets full everyday, in which case it would be best to always offer rooms at their high prices as the Conservative algorithm does. Most hotels usually lie between these two extremes, that is, depending on occasions they are sometimes full and other times empty. In such cases, these experiments show that the Modified Algorithm 3 (i.e., the 15% algorithm) has the best performance.
7. Conclusions
In this paper, we studied the problem of offering assortments of items with limited inventories at different stages to customers of different types that have limited patience levels. Customers appear on the platform in a stochastic online fashion, that is, even though there is no information on the type of the future customer arrivals, their arrival rate is known. We studied the problem when only a single item (i.e., assortments of size 1) and general assortments can be offered to customers. We tested our algorithms on real world data and showed their significant improvements compared to the benchmark methods.

References
Adamczyk M, Grandoni F, Mukherjee J (2015) Improved approximation algorithms for stochastic matching. *Algorithms-ESA 2015*, 1–12 (Springer).
Bansal N, Gupta A, Li J, Mestre J, Nagarajan V, Rudra A (2012) When lp is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica* 63(4):733–762.
Bodea T, Ferguson M, Garrow L (2009) Data set—choice-based revenue management: Data from a major hotel chain. *Manufacturing & Service Operations Management* 11(2):356–361.
Brubach B, Sankararaman KA, Srinivasan A, Xu P (2017) Attenuate locally, win globally: An attenuation-based framework for online stochastic matching with timeouts. *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, 1223–1231.
Chen N, Immorlica N, Karlin AR, Mahdian M, Rudra A (2009) Approximating matches made in heaven. *International Colloquium on Automata, Languages, and Programming*, 266–278 (Springer).
Chen X, Ma W, Simchi-Levi D, Xin L (2016) Assortment planning for recommendations at checkout under inventory constraints. *Available at SSRN 2853093*.
Chernev A (2006) Decision focus and consumer choice among assortments. *Journal of Consumer Research* 33(1):50–59.
Chernev A, et al. (2012) Product assortment and consumer choice: An interdisciplinary review. *Foundations and Trends® in Marketing* 6(1):1–61.
Cheung WC, Simchi-Levi D (2016) Efficiency and performance guarantees for choice-based network revenue management problems with flexible products. *Available at SSRN 2823339*.
Désir A, Goyal V, Zhang J (2014) Near-optimal algorithms for capacity constrained assortment optimization. *Available at SSRN 2543309*.
Feldman J, Segev D (2019) Improved approximation schemes for mnl-driven sequential assortment optimization. *Available at SSRN 3440645*.
Flores A, Berbeglia G, Van Hentenryck P (2019) Assortment optimization under the sequential multinomial logit model. *European Journal of Operational Research* 273(3):1052–1064.
Gallego G, Iyengar G, Phillips R, Dubey A (2004) Managing flexible products on a network.

Gallego G, Li A, Truong VA, Wang X (2018) Approximation algorithms for product framing and pricing. Available at SSRN 3260279.

Gandhi R, Khuller S, Parthasarathy S, Srinivasan A (2006) Dependent rounding and its applications to approximation algorithms. Journal of the ACM (JACM) 53(3):324–360.

Garey MR, Johnson DS (2002) Computers and intractability, volume 29 (wh freeman New York).

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

Khalil HK (2002) Nonlinear systems. Upper Saddle River.

Lei YM, Jasin S, Uichanco J, Vakhutinsky A (2018) Randomized product display (ranking), pricing, and order fulfillment for e-commerce retailers. Available at SSRN 3282019.

Liu N, Ma Y, Topaloglu H (2019a) Assortment optimization under the multinomial logit model with sequential offerings.

Liu N, van de Ven PM, Zhang B (2019b) Managing appointment booking under customer choices. Management Science.

Liu Q, Van Ryzin G (2008) On the choice-based linear programming model for network revenue management. Manufacturing & Service Operations Management 10(2):288–310.

Ma W (2014) Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms. Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms, 1154–1163 (Society for Industrial and Applied Mathematics).

Ma W, Simchi-Levi D (2017) Online resource allocation under arbitrary arrivals: Optimal algorithms and tight competitive ratios. Available at SSRN 2989332.

Ma W, Simchi-Levi D, Zhao J (2018) Dynamic pricing under a static calendar. Available at SSRN 3251015.

Talluri K, Van Ryzin G (2004) Revenue management under a general discrete choice model of consumer behavior. Management Science 50(1):15–33.

Talluri KT, Van Ryzin GJ (2006) The theory and practice of revenue management, volume 68 (Springer Science & Business Media).

van Ryzin G, Vulcano G (2015) A market discovery algorithm to estimate a general class of nonparametric choice models. Management Science 61(2):281–300.

Wang R, Sahin O (2017) The impact of consumer search cost on assortment planning and pricing. Management Science 64(8):3649–3666.

Xu Y, Wang Z (2018) Assortment optimization for a multi-stage choice model.
Appendix A: Deferred Proofs from Section 3

Proof of Lemma 1. To prove this, we need to show that the item offering strategy of any online algorithm satisfies the constraints of the LP on any instance of the problem. If so, the offering strategy is a feasible solution to the LP; therefore, the total revenue of the algorithm is bounded by the optimal value of the LP.

Let \( x_{ij} \) be the probability that algorithm \( A \) offers item \( i \) to a customer of type \( j \), that is, \( x_{ij} = \Pr[\text{Offer}_t(i, j)] \); thus, \( 0 \leq x_{ij} \leq 1 \) satisfying constraint (2d). Clearly, no algorithm can offer more than \( \ell_j \) items to a customer of type \( j \). Therefore, for each type \( j \) and time-step \( t \)

\[
\sum_{i=1}^{n} \text{Offer}_t(i, j) \leq \ell_j.
\]

Taking the expectation on algorithm’s offering strategy we have that any valid strategy satisfies constraints (2c) of the LP, that is \( \sum_{i=1}^{n} x_{ij} \leq \ell_j \) for all customer types \( j \). A customer purchases either 0 or 1 of the items offered to her, thus for any type \( j \) and time-step \( t \),

\[
\sum_{i=1}^{n} \text{Accept}_t(i, j) \leq 1.
\]

By taking expectation of above and conditioning on whether each item \( i \) was offered, we have

\[
\sum_{i=1}^{n} \Pr[\text{Accept}_t(i, j)|\text{Offer}_t(i, j)] \Pr[\text{Offer}_t(i, j)] = \sum_{i=1}^{n} p_{ij} x_{ij} \leq 1.
\]

Therefore, any valid item offering strategy satisfies constraint (2b). Similarly, since we are assuming there exists only a single copy of each item, each one of the items is matched to at most one customer. In other words, an item \( i \) is matched at most at one of the time-steps \( 1, \ldots, T \) and the type of the customer arriving at that time-step is one among \( 1, \ldots, j \). Therefore, for each item \( i \),

\[
\sum_{t=1}^{T} \sum_{j=1}^{m} \text{Accept}_t(i, j) \cdot \text{Type}_t(j) \leq 1,
\]

and taking the expectation of it and using the tower property of conditional expectation we have

\[
\sum_{t=1}^{T} \sum_{j=1}^{m} \Pr[\text{Accept}_t(i, j)|\text{Offer}_t(i, j) \cap \text{Type}_t(j)] \Pr[\text{Offer}_t(i, j)|\text{Type}_t(j)] \Pr[\text{Type}_t(j)] = \sum_{j=1}^{m} T p_{ij} x_{ij} q_j \leq 1.
\]

Thus, any valid offering strategy satisfies constraint (2a) as well. This proves that \( x_{ij} \) is a feasible solution of the LP, and it follows that the revenue of each online algorithm is upper-bounded by the optimal value of the LP. \( \square \)

Proof of Lemma 3. We first find \( \lim_{z \to 0} f(z) \). Observe that both \( z \) and \( 1 - \exp(-z) \) are 0 at \( z = 0 \). Therefore, we use L’Hôpital rule to determine this, which gives \( \lim_{z \to 0} f(z) = \lim_{z \to 0} \frac{\exp(-z)}{z} = 1 \), as claimed. Moreover, \( f’(z) = \frac{-1 + z \exp(-z) + \exp(-z)}{z^2} \) and \( f''(z) = \frac{2 - 2z \exp(-z) - 2z \exp(-z) - z^2 \exp(-z)}{z^3} \). It can easily be checked that \( f’(z) < 0 \) for all \( z \in [0, 1] \) as the largest value of the numerator of \( f’(z) \) takes place when \( z = 1 \) and it is still negative. Therefore, \( f(z) \) is a decreasing function on the interval of interest, which in combination with \( \lim_{z \to 0} f(z) = 1 \) concludes that \( f(z) \leq 1 \) for \( z \in [0, 1] \). To see whether \( f’(z) \) is bounded for \( z \in [0, 1] \), we first calculate \( f’(1) \) and \( \lim_{z \to 0} f’(z) \) which are \( -1 + 2 \exp(-1) \) and \( -\frac{1}{2} \), respectively, where the latter was calculated by applying L’Hôpital rule two times. Since \( f’(z) \) is continuous and it is bounded at the
boundaries, it is also bounded in the \([0,1]\) interval. The last thing to show is the convexity of \(f(z)\) in the given interval. Observe that on the boundaries of \([0,1]\), \(\lim_{z \to 0} f''(z) = \frac{1}{3} > 0\), again using L'Hôpital rule, and \(f''(1) = 2 - \frac{5}{3} > 0\). Hence, if the numerator of \(f''(z)\) never equals 0, we have that \(f''(z) \geq 0\) in the given interval, this \(f(z)\) being convex. \(f''(z) = 0\) only if \(-2 - 2\exp(-z) - 2z\exp(-z) - z^2\exp(-z) = 0\). We show this does not take place by the way of contradiction. To do so, we rewrite this as \(\exp(z) = 1 + z + \frac{z^2}{2}\). Note that by the Taylor expansion, \(\exp(z) > 1 + z + \frac{z^2}{2}\), implying that we the numerator of \(f''(x)\) and in turn \(f''(z)\) cannot be 0. Thus, \(f''(z) > 0\) for \(z \in [0,1]\); in other words, for this interval \(f(z)\) is convex. \(\square\)

**Proof of Theorem 3.** Let \(x^*\) be the optimal solution of the LP. Consider an arbitrary item \(i\) and type \(j\). Events \(\text{Avail}(i)\) and \(\text{Type}(j)\) are prerequisites for event \(\text{Offer}(i,j)\) to occur, which in turn is a prerequisite for events \(\text{True-offer}(i,j)\) and \(\text{Accept}(i,j)\) to take place. Therefore, we can decompose \(\Pr[\text{Accept}(i,j)]\) as

\[
\Pr[\text{Accept}(i,j)|\text{True-offer}(i,j)] \cdot \Pr[\text{True-offer}(i,j)|\text{Offer}(i,j)] \times \Pr[\text{Offer}(i,j)|\text{Avail}(i) \cap \text{Type}(j)] \cdot \Pr[\text{Avail}(i) \cap \text{Type}(j)].
\]

(35)

\(\Pr[\text{Accept}(i,j)|\text{True-offer}(i,j)]\) is simply equal to \(p_{ij}\), because conditional on customer \(t\) having type \(j\) and being offered item \(i\), her purchase choice is independent from all other events. Similarly, \(\Pr[\text{Avail}(i) \cap \text{Type}(j)] = \gamma_t q_j\).

To analyze \(\Pr[\text{Offer}(i,j)|\text{Avail}(i) \cap \text{Type}(j)]\), note that the algorithm will offer items during time-step \(t\) using the randomized procedure from Lemma 2, with \(\ell = \ell_j\), \(A = U^t_j = \{i' : \text{Avail}(i')\}\), and \(p_{ij} = p_{ij}^*, x_{ij}^* = x_{ij}^*\) for all \(i' \in A\). By LP constraints (2b)-(2d), we know that \((x_{ij}^*)_{i' \in A}\) satisfies the required conditions in Lemma 2. Therefore, by Lemma 2, conditional on \(i \in A\), the probability of \(i\) being offered to customer \(t\) who is of type \(j\) is at least \(x_{ij}^* \cdot (1 - \exp(-W_t(i,j))/W_t(i,j))\). If \(\ell_j \geq n\) for all types \(j\), then \(W_t(i,j)\) denotes the random variable

\[
W_t(i,j) = \frac{1}{1 - p_{ij} x_{ij}^*} \sum_{i' \neq i} p_{ij} x_{ij}^* \mathbb{1}(\text{Avail}(i')).
\]

(36)

where \(W_t(i,j)\) is understood to be \(1\) if \(p_{ij} x_{ij}^* = 1\). Moreover, if \(\sum_{i' \in [n]} p_{ij} \leq 1\) for all types \(j\), then

\[
W_t(i,j) = \frac{1}{1 - p_{ij}} \sum_{i' \neq i} p_{ij} x_{ij}^* \mathbb{1}(\text{Avail}(i'))
\]

(37)

where \(W_t(i,j)\) is understood to be \(1\) if \(p_{ij} = 1\). Going back to finding the probability of an item \(i\) being offered, using the tower property of conditional expectation, we get that

\[
\Pr[\text{Offer}(i,j)|\text{Avail}(i) \cap \text{Type}(j)] = \mathbb{E}_{W_t(i,j)|\text{Avail}(i) \cap \text{Type}(j)} \left[ \Pr[\text{Offer}(i,j)|\text{Avail}(i) \cap \text{Type}(j), W_t(i,j)] \right]
\]

\[
\geq \mathbb{E}_{W_t(i,j)|\text{Avail}(i) \cap \text{Type}(j)} \left[ x_{ij}^* \cdot \frac{1 - \exp(-W_t(i,j))}{W_t(i,j)} \right]
\]

\[
= x_{ij}^* \cdot \mathbb{E}_{W_t(i,j)|\text{Avail}(i) \cap \text{Type}(j)} \left[ f(W_t(i,j)) \right]
\]

(38)

By Lemma 3, \(f(z) = (1 - e^{-z})/z\) is convex and decreasing, thus, applying Jensen's inequality

\[
\mathbb{E}_{W_t(i,j)|\text{Avail}(i) \cap \text{Type}(j)} \left[ f(W_t(i,j)) \right] \geq f(\mathbb{E}[W_t(i,j)|\text{Avail}(i) \cap \text{Type}(j)])
\]

(39)
We aim to bound the right-hand side of (39) below. Since conditioning on Type\(_i(j)\) has no effect on \(\text{Avail}_i(i')\) for any \(i' \neq i\), by taking conditional expectation of (36) we have

\[
E[W_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] = \frac{1}{1 - p_{ij}x_{ij}^*} \sum_{i' \neq i} p_{i'j}x_{i'j}^* E[1(\text{Avail}_i(i'))|\text{Avail}_i(i)].
\]

At this point, we invoke (Brubach et al. 2017, Lemma 3.1) from their original attenuation framework, which implies that at time-step \(t\), item \(i\) being available and item \(i'\) being available are negatively correlated. Therefore, we can subsequently derive

\[
E[W_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] \leq \frac{1}{1 - p_{ij}x_{ij}^*} \sum_{i' \neq i} p_{i'j}x_{i'j}^* \Pr[\text{Avail}_i(i')]
= \frac{1}{1 - p_{ij}x_{ij}^*} \sum_{i' \neq i} p_{i'j}x_{i'j}^* = \gamma_t \cdot \Pr[\text{Avail}_i(i')],
\]

with the final inequality coming from Corollary 1. Likewise, for the case that \(\sum_{i' \in [n]} p_{i'j} \leq 1\):

\[
E[W_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] \leq \frac{1}{1 - p_{ij}x_{ij}^*} \sum_{i' \neq i} p_{i'j}x_{i'j}^* \Pr[\text{Avail}_i(i')]
= \frac{1}{1 - p_{ij}x_{ij}^*} \sum_{i' \neq i} p_{i'j}x_{i'j}^* = \gamma_t \cdot \Pr[\text{Avail}_i(i')],
\]

where again the last inequality comes from Corollary 1.

Combining (38) and (39) with (40) and (41), we get that for both cases of \(\ell_j \geq n\) and \(\sum_{i' \in [n]} p_{i'j} \leq 1\) it holds that \(\Pr[\text{Offer}_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] \geq x_{ij}^* \cdot (1 - e^{-\gamma_t})/\gamma_t\); thus,

\[
\Pr[\text{Offer}_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] \cdot \Pr[\text{Avail}_i(i) \cap \text{Type}_i(j)] = (1 - e^{-\gamma_t})q_j x_{ij}^*.
\]

Clearly there exists an attenuation factor \(a_{ij}^{\text{edge}} \in [0, 1]\) such that \(a_{ij}^{\text{edge}} \cdot \Pr[\text{Offer}_t(i,j)|\text{Avail}_i(i) \cap \text{Type}_i(j)] \geq (x_{ij}^* - 1 - e^{-\gamma_t} q_j x_{ij}^*).

Defining such an attenuation factor for all \(i \in [n]\) and \(j \in [m]\), we can deduce that for any item \(i\),

\[
\Pr[\text{Avail}_{i+1}(i)] = \Pr[\text{Avail}_i(i)] - \sum_{j=1}^{m} \Pr[\text{Accept}_t(i,j)] = \gamma_t - (1 - e^{-\gamma_t}) \sum_{j=1}^{m} q_j p_{ij} x_{ij}^* \geq \gamma_t - \frac{1 - e^{-\gamma_t}}{T},
\]

with the inequality coming from LP constraint (2a). Therefore, there exists an attenuation factor \(a^{\text{vertex}}_t \in [0, 1]\) such that

\[
\Pr[\text{Avail}_{i+1}(i)] = \gamma_t - \frac{1 - e^{-\gamma_t}}{T} = \gamma_{t+1},
\]

completing the proof and inductively establishing our modified attenuation framework. □

**Proof of Lemma 4.** First, we remark that for each \(t \in [T]\), \(\gamma_t\) is not only a function of \(t\), but also a function of \(T\), which can be clearly seen from (42). To capture these two dependences, for a fixed \(T\), we define function \(g^T : [0, 1] \rightarrow [0, 1]\) such that for all \(t \in [T]\), \(g^T((t-1)/T) = \gamma_t\) and for all \(t_1 < t < t_1 + 1\), where \(t_1 \in [T]\), \(g^T((t-1)/T)\) is the linear combination of \(g^T((t_1-1)/T)\) and \(g^T(t_1/T)\). Defining \(z = (t-1)/T\) and
for a fixed $z$, $\{g^T(z)\}_{T \in \mathbb{N}}$ is a sequence of function and we want to study the behavior $g^T(z)$ as $T$ goes to infinity. Using this notation, (42) can be written as:

$$\frac{g^T(z + \frac{1}{T}) - g^T(z)}{\frac{1}{T}} = -1 + \exp(-g^T(z)).$$

(43)

By taking the limit of the above equation as $T \to \infty$ we define a new function $h(z)$, i.e., $h(z) = \lim_{T \to \infty} g^T(z)$. Recall that $\gamma_t$ is the probability that each item is available at time-step $t$ if there are a total of $T$ customer arrivals. Therefore $0 \leq g^T(z) = \gamma_t \leq 1$. Since for such $g^T(z)$’s $(1 - \exp(-g^T(z)))$ is positive, recursion (43) indicates that as $z$ increases $g^T(z)$ decreases as $z$ grows. In addition to this, we here want to prove that as $T$ increases $g^T(z)$ does not decrease. This is something stronger than what the lemma states and from this it can be concluded that for all $t$ and $T$ we have

$$g^T(z) \leq \lim_{T \to \infty} g^T(z) = h(z),$$

(44)

which then concludes that $\gamma_{T+1} = g^T(1) \leq h(1)$. To show (44) holds we use induction on $T$, with the inductive hypothesis that if $g^T(z)$ does not decrease with $T$ then $g^T(z + 1/T)$ does not decrease $T$ either. The base of this induction is that $g^T(0)$ (i.e., when $t = 1$) does not decrease with $T$. This is clearly true as $\gamma_1 = g^T(0) = 1$ and constant. Now assume that for some $0 \leq z \leq 1$, we have that $g^T(z)$ is not decreasing with $T$. We can formulate $g^T(z + 1/T)$ from (43), which gives

$$g^T(z + \frac{1}{T}) = g^T(z) - \frac{1}{T}(1 - \exp(-g^T(z))).$$

To show $g^T(z + \frac{1}{T})$ does not decrease with time, we differentiate $g^T(z + \frac{1}{T})$ with respect to $T$:

$$\frac{dg^T(z + \frac{1}{T})}{dT} = \frac{dg^T(z)}{dT} + T \frac{dg^T(z)}{dT} \frac{dT}{T^2} - (1 - \exp(-g^T(z))) \frac{dT}{T^2} = \frac{dg^T(z)}{dT} \left(T - \exp(-g^T(z))\right) \frac{T}{T^2} + \frac{1 - \exp(-g^T(z))}{T^2}.$$

(45)

Since $g^T(z) \geq 0$, the second term in (45) is clearly positive. For the first term of (45), by the same reason $(T - \exp(-g^T(z))) \geq 0$; moreover, recall that by the inductive assumption, $dg^T(z)/dT \geq 0$. Therefore, we have that $dg^T(z + \frac{1}{T})/dT \geq 0$ and our induction proof concludes. This implies that (44) holds and consequently, $\gamma_{T+1} = g^T(1) \leq h(1)$.

It remains to find the value of $h(1)$. Recall that $h(z)$ was defined $h(z) = \lim_{T \to \infty} g^T(z)$. Taking the limit of (43) as $T$ goes to infinity results in $\frac{dh}{dz}$ on the right-hand side. Therefore, in the limit, function $h(z)$ needs to satisfy the following differential equation

$$\frac{dh(z)}{dz} = -1 + \exp(-h(z)),$$

(46)

with the boundary condition the same as that of $g^T(0)$, that is, $h(0) = 1$. We solve this differential equation for $h(z)$. By multiplying both sides by $\exp(h(z))$ and $dz$ we have

$$\exp(h(z))dh(z) = -(\exp(h(z)) - 1)dz,$$

thus

$$\exp(h(z)) = \frac{-1}{\exp(h(z)) - 1}dz.$$
The above equation is equivalent to
\[ d \ln(\exp(h(z)) - 1) = -dz. \]

Integrating the above in combination to the boundary condition \( h(0) = 1 \) concludes that
\[ h(z) = \ln((e - 1) \exp(-z) + 1). \]  

Plugging \( z = 1 \) in (47) gives \( h(1) = \ln(2 - 1/e) \), finalizing our claim of \( \gamma_{T+1} \leq h(1) = \ln(2 - 1/e) \). \( \Box \)

**Proof of Lemma 5.** We prove the lemma by coming up with an instance of the problem for which we can show it is not possible to have an algorithm with an approximation ratio larger than \( 1 - \ln(2 - 1/e) \). Consider an instance of the problem where there exists \( T = n \) time-steps in which customers of \( n \) different types may arrive, where \( \ell_j = n \) and \( q_j = 1/n \) for all customer types \( j = 1, \ldots, n \). Moreover, there are \( n \) items and for all pairs of \( i \) and \( j \), \( p_{ij} = \frac{1}{n} \) and \( r_{ij} = 1 \). As \( n \) goes to infinity, we show that no online algorithm can have an approximation ratio larger than \( 1 - \ln(2 - 1/e) \).

Since the patience level is equal to the total number of items, a visiting customer would not go out of patience even if all the items are shown to her. Therefore, customers’ patience is not a concern in this case of the problem. To study the best possible approximation ratio we first need to determine the value of \( \text{OPT} \). For each time-step \( t = 1, \ldots, n \), let \( N_t \) denote the total number of available items at the beginning of time-step \( t \); therefore, we have \( N_1 = n \). Since \( N_{n+1} \) denotes the number of remaining items once the process is over, for any algorithm its total revenue is \( \text{ALG} = n - N_{n+1} \). The approximation ratio of an algorithm is the ratio of its total expected revenue to \( \text{OPT} \). Denoting the approximation ratio with \( \alpha \) we have
\[ \alpha = \frac{\mathbb{E}[\text{ALG}]}{\text{OPT}} \leq \frac{\mathbb{E}[n - N_{n+1}]}{n} = 1 - \mathbb{E}\left[\frac{N_{n+1}}{n}\right]. \]  

In what follows we show that in the limit that \( n \) goes to infinity \( \mathbb{E}[N_{n+1}]/n \geq \ln(2 - 1/e) \), and using (48) conclude that \( \alpha \leq 1 - \ln(2 - 1/e) \). To do this, first note that since at each time-step at most one item can be purchased by a customer, \( N_t - N_{t+1} \) is either 0 or 1. Therefore, \( \mathbb{E}[N_t - N_{t+1}|N_t] \) is equal to the probability that a purchase is made during time-step \( t \). As there are \( N_t \) items at the beginning of this time-step and all of them can get the opportunity to be offered to the visiting customer and their purchase probability is \( 1/n \) we have
\[ \mathbb{E}[N_t - N_{t+1}|N_t] = 1 - \left(1 - \frac{1}{n}\right)^{N_t}. \]

Taking another expectation of the above equation we have
\[ \mathbb{E}[N_t - N_{t+1}] = \mathbb{E}\left[1 - \left(1 - \frac{1}{n}\right)^{N_t}\right]. \]

With simple calculus and taking twice differentiation it can be seen that \( 1 - \left(1 - \frac{1}{n}\right)^{N_t} \) is a concave function. Therefore, using Jensen’s inequality and negating the above equation we have
\[ \mathbb{E}[N_{t+1} - N_t] \geq -1 + \left(1 - \frac{1}{n}\right)^{\mathbb{E}[N_t]} = -1 + \exp\left(-\mathbb{E}\left[\frac{N_t}{n}\right]\right) - o(1). \]  

Electronic copy available at: https://ssrn.com/abstract=3443109
Defining function \( g^n : [0, 1] \rightarrow [0, 1] \) such that for all \( t \in [n] \), \( g^n((t-1)/n) = \mathbb{E}[N_t/n] \) and setting \( z = (t-1)/n \), inequality (49) becomes

\[
\frac{g^n(z + \frac{1}{n}) + g^n(z)}{\frac{1}{n}} \geq -1 + \exp(-g^n(z)) - o(1).
\] (50)

Taking the limit of above as \( n \) goes to infinity and introducing \( b(z) = \lim_{n \to \infty} g^n(z) \) we have

\[
\frac{db(z)}{dz} \geq -1 + \exp(-b(z)),
\] (51)

with the boundary condition \( b(0) = \mathbb{E}[N_1/n] = 1 \). Observe that the way \( b(z) \) is defined we have \( b(1) = \lim_{n \to \infty} \mathbb{E}[N_{n+1}/n] \). We now show \( b(1) \geq \ln(2 - 1/e) \). Differential inequality (51) is similar to differential equation (46) (that is, \( dh(z)/dz = -1 + \exp(-h(z)) \)) with boundary condition \( h(0) = 1 \) discussed in the proof of Lemma 4, with the difference of having an equation there and an inequality here. By (Khalil 2002, Lemma 3.4), for all \( 0 < z \leq 1 \) we have \( b(z) \geq h(z) \), in particular, \( b(1) \geq h(1) = \ln(2 - 1/e) \). Hence, taking the limit of (48) as \( n \) goes to infinity we have

\[
\alpha = \frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPT}]} \leq 1 - \mathbb{E}[\frac{N_{n+1}}{n}] \leq 1 - \ln(2 - 1/e) \simeq 0.51.
\]

Therefore, there exists no algorithm with an approximation ratio larger than \( 1 - \ln(2 - 1/e) \) for the stochastic matching with timeouts problem. \( \square \)

**Appendix B: Supplements to Section 4**

This section is constituted of two subsections. The first subsection discusses the proof of Theorem 4 as well as the supplementary lemmas for it to prove the 0.51 approximation for the multi-stage and multi-customer assortment optimization when the platform is allowed to show repeated items to customers. In the second subsection, we provide the deferred proofs from Section 4.

**B.1. Supplements to Section 4.1**

Our approach here is similar to, in fact a generalization of, the algorithms we had for the online stochastic matching with timeouts problem. That is, the problem is divided into two subproblems: the online subproblem is addressed via an offline black-box and the online problem uses an attenuation framework. Lemma 12 discusses the black-box algorithm and Lemma 13 is about the attenuation framework for this assortment optimization problem.

**Lemma 12 (Black-box Randomized Procedure (Assortment Version)).** Let \( A \) be a set of assortment of some coins, where these assortments can potentially overlap with one another. For an assortment \( S \in A \), when a coin \( i \in S \) is flipped it lands on "heads" with probability \( p(i, S) \) and independent from any other coin in any other assortment. Once an assortment is picked all its coins are flipped at once, which is referred to as flipping the assortment. We can flip assortments in any (possibly randomized) order, and must stop once we get a "heads", or have flipped \( \ell \) assortments, where \( \ell \) is a positive integer.

Let \((x(S))_{S \in A}\) be any vector of weights in \([0, 1]^{\left| A \right|}\) satisfying \( \sum_{S \in A} \sum_{i \in S} p(i, S) x(S) \leq 1 \) and \( \sum_{S \in A} x(S) \leq \ell \). Then there exists a randomized procedure for flipping the assortments such that the probability of any assortment \( S \) being flipped, before the process is stopped, is at least

\[
\frac{1 - e^{-w(S)}}{w(S)} \cdot x(S),
\] (52)
where \( w(S) = \frac{1}{1 - \sum_{i \in S} p(i, S)} \sum_{S' \neq S} \sum_{i \in S} p(i, S') x(S') \) (or \( w(S) \) is understood to be 1 if \( \sum_{i \in S} p(i, S) = 1 \)) if \( \sum_{S \in A} \sum_{i \in S} p(i, S) \leq 1 \) and \( w(S) = \frac{1}{1 - \sum_{i \in S} p(i, S) x(S)} \sum_{S' \neq S} \sum_{i \in S} p(i, S') x(S') \) (or \( w(S) \) is understood to be 1 if \( \sum_{i \in S} p(i, S) x(S) = 1 \)) if \( \ell \geq |A| \).

**Proof.** The proof of this lemma is essentially a generalization of that of Lemma 2 to account for assortments of coins. As the elements of \( (x(S))_{S \in A} \) are fractional and do not determine whether an assortment \( S \) should be flipped, we again use the GKPS procedure to round them to 0 and 1. So, assortment \( S \) is among the assortments that can be flipped if and only if \( x(S) \) is rounded up to 1 by the GKPS process. By applying the GKPS rounding, vector \( (x(S))_{S \in A} \) is rounded to \( (X(S))_{S \in A} \), where each \( (X(S))_{S \in A} \) satisfies all three properties mentioned in Theorem 2. We say an assortments comes heads if when the assortment is flipped at least one of its coins comes heads.

**The case of \( \ell \geq |A| \):** Algorithm 4 shows how the randomized algorithm works in this case of the problem. Consider any assortment \( S \in A \) and suppose it is rounded up by the GKPS process. Assuming \( Y_S = y \), an assortment \( S' \) that has also passed through the GKPS process is flipped before \( S \) if \( \frac{Y_{S'}}{1 - \sum_{i \in S'} p(i, S') x(S')} \leq \frac{y}{1 - \sum_{i \in S} p(i, S) x(S)} \). Let us provide an upper-bound on the probability that assortment \( S' \) is flipped before \( S \). To do this, we divide the problem into two cases: \( y \geq 2(1 - \sum_{i \in S} p(i, S) x(S)) \), and otherwise, \( y < 2(1 - \sum_{i \in S} p(i, S) x(S)) \).

\[
1 - \exp\left(-\frac{\sum_{i \in S'} p(i, S') x(S')}{\sum_{i \in S} p(i, S) x(S)} \right) \geq 1 - \exp\left(-\frac{2 \sum_{i \in S'} p(i, S') x(S')}{\sum_{i \in S} p(i, S) x(S)} \right) \geq 2(1 - \sum_{i \in S'} p(i, S') x(S')) \geq 1, \tag{53}
\]

where (53) uses the Taylor expansion of \( \exp(-2 \sum_{i \in S'} p(i, S') x(S')) \) as well as \( \sum_{i \in S'} p(i, S') x(S') \leq 1/2 \), which is implied by \( \sum_{S \in A} \sum_{i \in S} p(i, S) x(S) \leq 1 \) in addition to \( y \geq 2(1 - \sum_{i \in S} p(i, S) x(S)) \) and \( y \leq 1 \). This makes

\[
1 - \exp\left(-\frac{\sum_{i \in S'} p(i, S') x(S')}{\sum_{i \in S} p(i, S) x(S)} \right) \geq (1 - \frac{y}{2(1 - \sum_{i \in S} p(i, S) x(S))}) \cdot \frac{\sum_{i \in S'} p(i, S') x(S')}{\sum_{i \in S} p(i, S) x(S)}, \tag{54}
\]

\[
(1 - \sum_{i \in S'} p(i, S') x(S')) \cdot \frac{\sum_{i \in S'} p(i, S') x(S')}{\sum_{i \in S} p(i, S) x(S)}, \tag{55}
\]

Algorithm 4 Black-box (Assortment Version)

**INPUT:** \( \ell, A, x(S) \) and \( p(i, S) \) for all \( S \in A \) and \( i \in S \)

1. Apply the GKPS rounding to \( (x(S))_{S \in A} \). Let \( \tilde{U} \) be the set of assortments that are rounded up by the GKPS process.
2. For each assortment \( S \in \tilde{U} \) pick a number \( Y_S \) uniformly at random and IID from \([0, 1]\).
3. Flip assortments \( S \in \tilde{U} \) in an increasing order of \( \frac{Y_S}{1 - \sum_{i \in S} p(i, S) x(S)} \) (if \( \sum_{S \in A} \sum_{i \in S} p(i, S) \leq 1 \)) and \( \frac{Y_S}{1 - \sum_{i \in S} p(i, S) x(S)} \) (if \( \ell \geq |A| \)) until a “heads” comes or \( \ell \) assortments are flipped.
where again (54) uses a Taylor expansion similar to the one discussed earlier and assumption $y < 2(1 - \sum_{i \in S} p(i, S)x(S))$ was used in (55). Assortment $S'$ is flipped before $S$ if $Y_{S'} \leq (1 - \sum_{i \in S'} p(i, S')x(S')) \cdot \frac{1}{1 - \sum_{i \in S} p(i, S)x(S)}$ and $Y_{S'}$ is picked uniformly at random in $[0, 1]$; thus, (55) gives the probability of flipping $S'$ before $S$. Therefore, using $I_{S', S}$ to denote the indicator variable that assortment $S'$ was flipped before assortment $S$, we have

$$
\Pr[I_{S', S} | S, S' \in \hat{U}, Y_S = y] \leq \frac{1 - \exp(-y \sum_{i \in S'} p(i, S')x(S'))}{\sum_{i \in S'} p(i, S')x(S')}.
$$

We are now able to bound the probability that an assortment $S$ is flipped before the process is stopped for the case that $\ell \geq |A|$. A lower-bound on the probability that $S$ is flipped given it is rounded up by the GKPS process is provided below:

$$
\Pr[S \text{ flipped} | S \in \hat{U}] \geq \prod_{S' \neq S} \Pr[S' \text{ not heads before } S | S \in \hat{U}]
$$

$$
= \prod_{S' \neq S} \Pr[S' \text{ not heads before } S | S \in \hat{U}]
$$

$$
= \int_0^1 \prod_{S' \neq S} \Pr[S' \text{ not heads before } S | S \in \hat{U}, Y_S = y] dy
$$

$$
= \int_0^1 \prod_{S' \neq S} (1 - \Pr[S' \text{ heads before } S | S \in \hat{U}, Y_S = y]) dy
$$

$$
= \int_0^1 \prod_{S' \neq S} (1 - \Pr[S' \in \hat{U} | S \in \hat{U}, Y_S = y] \cdot \Pr[I_{S', S} | S, S' \in \hat{U}, Y_S = y] \cdot \Pr[S' \text{ flips heads} | S', S' \in \hat{U}, Y_S = y, I_{S', S}]) dy
$$

$$
\geq \int_0^1 \prod_{S' \neq S} (1 - x(S')) \frac{1 - \exp(-y \sum_{i \in S'} p(i, S')x(S'))}{\sum_{i \in S'} p(i, S')x(S')} \sum_{i \in S'} p(i, S')) dy
$$

$$
= \int_0^1 \prod_{S' \neq S} \exp(-y \sum_{i \in S'} p(i, S')x(S')) \frac{1}{1 - \sum_{i \in S} p(i, S)x(S)} dy
$$

$$
= \int_0^1 \exp(-y \sum_{S' \neq S} \sum_{i \in S'} p(i, S')x(S')) \frac{1}{1 - \sum_{i \in S} p(i, S)x(S)} dy
$$

$$
= \frac{1}{\sum_{S' \neq S} \sum_{i \in S'} \frac{p(i, S')x(S')}{1 - \sum_{i \in S} p(i, S)x(S)}} \left(1 - \exp\left(-\sum_{S' \neq S} \sum_{i \in S'} \frac{p(i, S')x(S')}{1 - \sum_{i \in S} p(i, S)x(S)}\right)\right) = \frac{1 - \exp(-w(S))}{w(S)},
$$

where (57) uses the fact that as there is enough patience to flip all assortments when $\ell \geq |A|$ as well as the independence among the events of assortments $S' \neq S$ not being flipped heads. Inequality (58) uses (56) and $\Pr[S' \in \hat{U} | S \in \hat{U}, Y_S = y] \leq \Pr[S' \in \hat{U}] = x(S')$. Therefore,

$$
\Pr[S \text{ flipped} | S \in \hat{U}] = \Pr[S \text{ flipped} | S \in \hat{U}] \cdot \Pr[S \text{ flipped} | S \in \hat{U}] = \Pr[S \text{ flipped} | S \in \hat{U}] \cdot x(S) \geq \frac{1 - \exp(-w(S))}{w(S)} \cdot x(S).
$$

**The case of $\sum_{S \in A} \sum_{i \in S} p(i, S) \leq 1$:** Similar to the previous case, first the GKPS process is performed on $(x(S))_{S \in A}$, and for each assortment $S$ that has rounded up by the GKPS process, i.e., $S \in \hat{U}$, a random variable $Y_S$ is picked IID and uniformly at random in $[0, 1]$, see Algorithm 4. For this case, assortments are ranked in an increasing order of $Y_S/(1 - \sum_{i \in S} p(i, S))$ and they are flipped in this order until the process stops. So, assuming $Y_S = y$, assortment $S'$ is flipped before $S$ if $\frac{Y_{S'}}{1 - \sum_{i \in S'} p(i, S')} \leq \frac{y}{1 - \sum_{i \in S} p(i, S)}$. Again, we divide...
the problem into the following two cases: \( y \geq 2(1 - \sum_{i \in S} p(i, S)) \), and otherwise, \( y < 2(1 - \sum_{i \in S} p(i, S)) \).

Beginning with the former case we have

\[
1 - \exp\left(\frac{-y \sum_{i \in S'} p(i, S')}{1 - \sum_{i \in S'} p(i, S')}\right) \geq 1 - \exp\left(-2 \sum_{i \in S'} p(i, S')\right) \geq 2\left(1 - \sum_{i \in S'} p(i, S')\right).
\]

(59)

We have \( \sum_{i \in S} p(i, S) + \sum_{i \in S'} p(i, S') \leq 1 \) as in this case it is assumed \( \sum_{S \in \mathcal{A}} \sum_{i \in S} p(i, S) \leq 1 \). This in addition to \( 2(1 - \sum_{i \in S} p(i, S)) \leq y \leq 1 \) gives \( \sum_{i \in S'} p(i, S') \leq \frac{1}{2} \). Therefore,

\[
1 - \exp\left(\frac{-y \sum_{i \in S} p(i, S)}{1 - \sum_{i \in S} p(i, S)}\right) \geq 2\left(1 - \sum_{i \in S} p(i, S)\right) \geq 1,
\]

making \( \frac{1 - \exp(-y \sum_{i \in S} p(i, S))}{1 - \sum_{i \in S} p(i, S)} \) a possible upper-bound on the probability that assortment \( S' \) gets flipped before \( S \) for this case of the problem. Next, we consider the case \( y < 2(1 - \sum_{i \in S} p(i, S)) \), for which we have

\[
1 - \exp\left(\frac{-y \sum_{i \in S} p(i, S)}{1 - \sum_{i \in S} p(i, S)}\right) \geq \left(1 - \frac{y \sum_{i \in S} p(i, S)}{2(1 - \sum_{i \in S} p(i, S))}\right) \cdot \frac{y}{1 - \sum_{i \in S} p(i, S)}
\]

(60)

\[
\geq \left(1 - \sum_{i \in S'} p(i, S')\right) \cdot \frac{y}{1 - \sum_{i \in S} p(i, S)}.
\]

(61)

where (60) and (61) use a Taylor expansion similar to the one discussed in the proof of Lemma 2 and \( y < 2(1 - \sum_{i \in S} p(i, S)) \), respectively. The probability of flipping \( S' \) before \( S \) is at most \( \left(1 - \sum_{i \in S'} p(i, S')\right) \cdot \frac{y}{1 - \sum_{i \in S} p(i, S)} \), therefore

\[
\Pr[S' \rightarrow S|S, S' \in \tilde{U}, Y_S = y] \leq \frac{1 - \exp(-y \sum_{i \in S'} p(i, S'))}{1 - \sum_{i \in S} p(i, S)}.
\]

With this, we go forward to provide the lower-bound on the probability that assortment \( S \) is flipped before the process stops.

\[
\Pr[S \text{ flipped}|S \in \tilde{U}] \geq \mathbb{E}_{\tilde{U}}\left[\int_0^1 \prod_{i' \in \tilde{U}\setminus S} \left(1 - \frac{1 - \exp(-y \sum_{i \in S'} p(i, S'))}{1 - \sum_{i \in S} p(i, S)} \right) \sum_{i \in S'} p(i, S') dy|S \in \tilde{U}\right]
\]

\[
= \mathbb{E}_{\tilde{U}}\left[\int_0^1 \exp(-y \sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} \frac{p(i, S')}{1 - \sum_{i \in S} p(i, S)} dy|S \in \tilde{U}\right]
\]

\[
= \mathbb{E}_{\tilde{U}}\left[\frac{1}{\sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S')} \cdot \left|S \in \tilde{U}\right|\right]
\]

\[
\geq \frac{1}{\sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S')} \cdot \left(1 - \exp(-\sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S'))\right)
\]

\[
\geq \frac{1}{w(S)} \cdot \left(1 - \exp(-\sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S'))\right).
\]

The reason behind (62) is that, for \( S \) to get flipped before the process is stopped, all assortment in \( \tilde{U} \) that are flipped before it need to come tails. In (63), Jensen’s inequality and convexity of function \( (1 - \exp(-z))/z \) are used (see Lemma 3). The fact that \( (1 - \exp(-z))/z \) is a decreasing function and \( \mathbb{E}_{\tilde{U}}[\sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S')/(1 - \sum_{i \in S} p(i, S))|S \in \tilde{U}] \leq \sum_{S' \in \tilde{U}\setminus S} \sum_{i \in S'} p(i, S')x(S')/(1 - \sum_{i \in S} p(i, S)) \) are used.
for (64). The former is by Lemma 3 and a similar proof for the latter is provided in the proof of Lemma 2. Therefore, for this case of the problem we have

$$\Pr[\text{S flipped}] \geq 1 - \frac{\exp(-w(S))}{w(S)} \cdot x(S).$$

□

The above lemma addressed the offline subproblem for the case that assortments of items can be offered at each stage of the problem and showing repeated items is allowed. We now shift our focus to the attenuation framework used to address the online subproblem for this problem. Recall that the attenuation framework used in the stochastic matching with timeouts problem consisted of two parts, those are, vertex and edge-attenuation frameworks. In a few words, vertex-attenuation regulated the probability that each item $i$ is available at the beginning of each time-step; whereas, edge-attenuation adjusted the probability each item $i$ is offered to each customer type $j$. For the case that an assortment of items is offered at each stage to a customer, we use a similar attenuation framework to address the online subproblem. The difference is that here the edge-attenuation framework regulates the probability that an item $i$ along with an assortment $S \ni i$ is offered to a customer type $j$.

Let $U_t$ denote the set of items that are still available when customer $t$ arrives. Suppose customer $t$ is of type $j$, then let $S_j(U_t)$ denote the set of assortments in $S$ for which $x^*_j(S) > 0$ with some modifications performed on them: the items that are sold-out are removed from each of these assortments and if an assortment has become empty by doing so, then it is not included in $S_j(U_t)$. Moreover, let $\gamma_t$ be as it was defined in Definition 3. Lastly, we use events $\text{Type}_t(j)$ and $\text{Avail}_t(i)$ as introduced in Definition 2 and we modify the other events discussed there into $\text{Offer}_t(i, S, j)$, $\text{True-offer}_t(i, S, j)$, and $\text{Accept}_t(i, S, j)$. These events are natural extensions of similar events in Definition 2 with the purpose of capturing an item $i$ being offered along an assortment $S$, formally defined below.

DEFINITION 6. For each $t \in [T]$, item $i \in [n]$, assortment $S \in S$ and type $j \in [m]$ let us define the following events:

- $\text{Offer}_t(i, S, j)$: the algorithm (pre-attenuation) intends to offer item $i$ along assortment $S$ to customer $t$ that is of type $j$;
- $\text{True-offer}_t(i, S, j)$: the algorithm (post-attenuation) offers item $i$ along assortment $S$ to customer $t$, whose type is $j$;
- $\text{Accept}_t(i, S, j)$: customer $t$, with type $j$, would have purchased item $i$ if truly offered along assortment $S$ (i.e., offered post-attenuation).

With these in mind let us discuss the overall online algorithm for the multi-stage and multi-customer assortment optimization problem where an item can be offered multiple times to a customer and prove the algorithm’s performance.

LEMMA 13 (Modified Attenuation Framework (Assortment Version)). Consider any time-step $t \in [T]$. Suppose there exist attenuation factors $a^\text{vertex}_t(i), a^\text{edge}_t(i, S, j) \in [0, 1]$ for all items $i \in [n]$, customer types $j \in [m]$, time-steps $t' < t$, and assortments $S \ni i$ and $S \in S$ on which Algorithm 5 can be run until the start of time-step $t$, at which point for all items $i$,

$$\Pr[\text{Avail}_t(i)] = \gamma_t.$$
Algorithm 5 Online Algorithm for the Assortment Optimization with Repeated Offerings

1: Before any customers arrive, solve the MCDLP-R to get \( x^* \).

2: for time-steps \( t = 1, \ldots, T \) do

3: Suppose the arriving customer be of type \( j \). Let \( U^t \) be the set of available items when customer \( t \) arrives and let the set of assortments \( S_j(U^t) \) be defined as \( S_j(U^t) = \{ S' : S' \subseteq U^t, S' \neq \emptyset, \exists S \in S \text{ s.t. } S' \subseteq S \text{ and } x_j^*(S) > 0 \} \). Also, let \( x^*(S_j(U^t)) \) be the elements of \( x^* \) that correspond to customer type \( j \) and assortments in \( S_j(U^t) \).

4: Run the offline black-box (Algorithm 4) on \( x^*(S_j(U^t)) \) as a subroutine and apply edge-attenuation to ensure each item \( i \in U^t \) along with an assortment \( S \ni i \) such that \( S \in S_j(U^t) \) is offered to customer \( t \) with probability \( x_j^*(S) \cdot (1 - e^{-\gamma t})/\gamma t \).

5: Apply vertex-attenuation to each item so that they are available with probability equal to \( \gamma_{t+1} = \gamma_t - (1 - \exp(-\gamma t))/T \) at time-step \( t + 1 \).

Then there exist attenuation factors \( a_i \text{vertex}(i), a_i \text{edge}(i,S,j) \in [0,1] \) for all \( i \in [n], j \in [m] \) and \( S \ni i \) and \( S \in S \) on which Algorithm 5 can be run during time-step \( t \), so that for all items \( i \) and types \( j \),

\[
\Pr[\text{Accept}_i(i,S,j)] = (1 - e^{-\gamma t})q_j p(i,S) x_j^*(S);
\]

\[
\Pr[\text{Avail}_{i+1}(i)] = \gamma_{t+1}.
\]

Proof. Let \( x^* \) denote the optimal solution of the MCDLP-R. For an item \( i \), a customer type \( j \), an assortment \( S \ni i \), and a time-step \( t \), \( \Pr[\text{Accept}_i(i,S,j)] \) can be decomposed as

\[
\Pr[\text{Accept}_i(i,S,j) | \text{True-offer}_i(i,S,j)] \cdot \Pr[\text{True-offer}_i(i,S,j) | \text{Offer}_i(i,S,j)] \times \Pr[\text{Offer}_i(i,S,j) | \text{Avail}_i(i) \cap \text{Type}_j(j)] \cdot \Pr[\text{Avail}_i(i) \cap \text{Type}_j(j)].
\]

(65)

Given that type of customer \( t \) is \( j \) and assortment \( S \) containing \( i \) is offered to her we have \( \Pr[\text{Accept}_i(i,S,j) | \text{True-offer}_i(i,S,j)] = p_j(i,S) \). Also, it can be easily seen that \( \Pr[\text{Avail}_i(i) \cap \text{Type}_j(j)] = \gamma_t q_j \). Furthermore, similar to what discussed earlier in the proof of Theorem 3, the attenuation framework ensures that \( a_i \text{edge}(i,S,j) = \Pr[\text{True-offer}_i(i,S,j) | \text{Offer}_i(i,S,j)] \). Not that we use the notation \( (i,S,j) \) to emphasize that item \( i \) is offered to a type \( j \) customer as a part of assortment \( S \) and we stress that edge attenuation is done on the combination of each item, assortment and customer type.

It remains to analyze \( \Pr[\text{Offer}_i(i,S,j) | \text{Avail}_i(i) \cap \text{Type}_j(j)] \). At time-step \( t \), Algorithm 5 offers assortments by calling the randomized procedure from Lemma 12, with \( \ell = \ell_j, \ A = S_j(U^t) \), and \( p(i',S') = p_j(i',S') x_j^*(S') \) for all \( i' \in A \) and \( S' \in S' \). LP constraints (18b)–(18d) ensure that \( (x(S'))_{S' \in A} \) satisfies the required conditions in Lemma 12. Therefore, by Lemma 12, given \( S \in S_j(U^t) \) and \( i \in S \), the probability that \( S \) is offered to customer \( t \) who is of type \( j \) is at least \( x_j^*(S) \cdot (1 - \exp(-W_i(S \ni i,j))/W_i(S \ni i,j) \), where we use the notation \( S \ni i \) to emphasize that \( i \) is in \( S \) and not removed due to being sold-out. \( W_i(S \ni i,j) \) is a random variable which will be discussed further in the following. If for all types \( j \) we have \( \ell_j \geq |S| \), then for an assortment \( S \in S_j(U^t) \) we have

\[
W_i(S,j) = \frac{1}{1 - \sum_{i' \in S} p_j(i',S)x_j^*(S)} \sum_{S' \neq S} \sum_{i' \in S'} p_j(i',S') x_j^*(S') \mathbb{1}(\text{Avail}_i(i'))
\]

(66)
where \( W_i(S, j) \) is understood to be 1 if \( \sum_{i' \in S} p_j(i', S)x_j^*(S) = 1 \). Furthermore, for the case that \( \sum_{i' \in [n]} \sum_{S' \ni i'} p_j(i', S') \leq 1 \) for all types \( j \) we have

\[
W_i(S, j) = \frac{1}{1 - \sum_{i' \in S} p_j(i', S)} \sum_{S \ni i'} \sum_{S' \ni i'} p_j(i', S')x_j^*(S') I(Avail_i(i')), \tag{67}
\]

where \( W_i(S, j) \) is understood to be 1 if \( \sum_{i' \in S} p_j(i', S) = 1 \). Let us now return to finding the probability of an item \( i \) being offered as part of an assortment \( S \) using a procedure similar to the one used in the proof of Theorem 3

\[
\Pr[Offer_i(i, S, j)|Avail_i(i) \cap Type_t(j)] = \mathbb{E}_{W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)} \left[ \Pr[Offer_i(i, S, j)|Avail_i(i) \cap Type_t(j), W_i(S \ni i,j)] \right] \\
\geq \mathbb{E}_{W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)} \left[ x_j^*(S) \cdot \frac{1 - \exp(-W_i(S \ni i,j))}{W_i(S \ni i,j)} \right] \\
= x_j^*(S) \cdot \mathbb{E}_{W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)} [ f(W_i(S \ni i,j))]. \\
\geq x_j^*(S) f(\mathbb{E}[W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)]). 	ag{68}
\]

where (68) is due to the fact that \( f(z) = (1 - e^{-z})/z \) is convex and decreasing as well as Jensen’s inequality.

Below we bound \( \mathbb{E}[W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)] \). Events \( Type_t(j) \) and \( Avail_i(i') \) are independent for any \( i' \neq i \); thus, taking conditional expectation of (66) provides

\[
\mathbb{E}[W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)] = \frac{\sum_{S' \ni i,j} \sum_{i' \in S'} p_j(i', S')x_j^*(S') \mathbb{E}[I(Avail_i(i'))|Avail_i(i)]}{1 - \sum_{i' \in S} p_j(i', S)x_j^*(S)}. 
\]

By (Brubach et al. 2017, Lemma 3.1) we have that at time-step \( t \), the availability of items \( i \) and \( i' \) are negatively correlate; thus,

\[
\mathbb{E}[W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)] \leq \frac{\sum_{S' \ni i,j} \sum_{i' \in S'} p_j(i', S')x_j^*(S') \Pr[Avail_i(i')]}{1 - \sum_{i' \in S} p_j(i', S)x_j^*(S)} \\
= \frac{\sum_{S' \ni i,j} \sum_{i' \in S'} p_j(i', S')x_j^*(S') \cdot \gamma_t}{1 - \sum_{i' \in S} p_j(i', S)x_j^*(S)} \leq (1) \cdot \gamma_t, \tag{69}
\]

with the final inequality coming from LP constraint (18b). Similarly, for the case that \( \sum_{i' \in S} \sum_{S' \ni i'} p_j(i', S') \leq 1 \):

\[
\mathbb{E}[W_i(S \ni i,j)|Avail_i(i) \cap Type_t(j)] \leq \frac{\sum_{S' \ni i,j} \sum_{i' \in S'} p_j(i', S')x_j^*(S') \Pr[Avail_i(i')]}{1 - \sum_{i' \in S} p_j(i', S)x_j^*(S)} \\
= \frac{\sum_{S' \ni i,j} \sum_{i' \in S'} p_j(i', S')x_j^*(S') \cdot \gamma_t}{1 - \sum_{i' \in S} p_j(i', S)} \leq (1) \cdot \gamma_t. \tag{70}
\]

Combining (68) with (69) and (70), for both cases of \( \ell_j \geq |S| \) and \( \sum_{i' \in S} \sum_{S' \ni i'} p_j(i', S') \leq 1 \), we have

\[
\Pr[Offer_i(i, S, j)|Avail_i(i) \cap Type_t(j)] \geq x_j^*(S) \cdot (1 - e^{-\gamma_t})/\gamma_t; \text{ therefore,}
\]

\[
\Pr[Offer_i(i, S, j)|Avail_i(i) \cap Type_t(j)] \cdot \Pr[Avail_i(i) \cap Type_t(j)] \geq (x_j^*(S) \cdot \frac{1 - e^{-\gamma_t}}{\gamma_t}) \gamma_t q_j \\
= (1 - e^{-\gamma_t}) q_j x_j^*(S).
\]

Moreover, there exists an attenuation factor \( \alpha_t \) such that \( \alpha_t (i, S, j) \cdot \Pr[Offer_i(i, S, j)|Avail_i(i) \cap Type_t(j)] \cdot \Pr[Avail_i(i) \cap Type_t(j)] = (1 - e^{-\gamma_t}) q_j x_j^*(S) \), which when plugged in (65) provides

\[
\Pr[Accept_i(i, S, j)] = (1 - e^{-\gamma_t}) q_j x_j^*(S).
\]
Defining such an attenuation factor for all \( i \in [n], j \in [m] \) and \( S \in \mathcal{S} \) such that \( S \ni i \) in addition to LP constraint (18a), it can be concluded that for any item \( i \)

\[
Pr[\text{Avail}_{t+1}(i)] = Pr[\text{Avail}_t(i)] - \sum_{j=1}^{m} \sum_{S \ni j} Pr[\text{Accept}_t(i, S, j)] \\
= \gamma_t - (1 - e^{-\gamma_t}) \sum_{j=1}^{m} \sum_{S \ni j} q_j p_j(i, S) x_j^*(S) \geq \gamma_t - \frac{1 - e^{-\gamma_t}}{T}.
\]

Thus, there exists an attenuation factor \( a_{i,\text{vertex}}(i) \in [0,1] \) such that

\[
Pr[\text{Avail}_{t+1}(i)] = \gamma_t - \frac{1 - e^{-\gamma_t}}{T} = \gamma_{t+1}.
\]

\[\square\]

**Proof.** [Proof of Theorem 4] The proof of Lemma 13 showed that at time-step \( t \), \( Pr[\text{Accept}_t(i, S, j)] = (1 - e^{-\gamma_t}) q_j p_j(i, S) x_j^*(S) \). Let \( E[\text{ALG}] \) denote the total expected revenue of Algorithm 5; therefore,

\[
E[\text{ALG}] = \sum_{t=1}^{T} \sum_{j=1}^{m} \sum_{S \in \mathcal{S}} \sum_{i \in S} r_{ij} x_j^*(S) r_{ij} p_j(i, S).
\]

Moreover,

\[
\text{OPT} = \sum_{j=1}^{m} T q_j \sum_{S \in \mathcal{S}} x_j^*(S) r_{ij} p_j(i, S).
\]

Therefore, the approximation ratio of Algorithm 5 is \( \frac{\sum_{t=1}^{T} (1 - e^{-\gamma_t})}{T} \). Theorem 1 showed that \( \sum_{t=1}^{T} (1 - e^{-\gamma_t}) \) is lower-bounded by \( (1 - \ln(2 - \frac{1}{e})) \); therefore, the multi-customer and multi-stage assortment optimization problem when items can be offered multiple times to customers has an approximation ratio of \( (1 - \ln(2 - \frac{1}{e})) \approx 0.51 \). \[\square\]

**B.2. Deferred Proofs from Section 4**

**Proof of Lemma 7.** The proof can be easily obtained by extending that of Lemma 1 to the case the probability of purchasing an item is also a function of the assortment within which that item was presented, i.e., \( p_j(i, S) \). Moreover, compared to the MCDLP-R, there is an additional constraint in the MCDLP-NR, that is, constraint (19d). As any valid algorithm does not show a previously seen item to a customer, taking the expectation, we have that any valid assortment offering strategy satisfies constraint (19d). \[\square\]

**Proof of Theorem 6.** First, note that in the discussed non-deterministic version of the multi-stage multi-customer assortment optimization problem a slight change in the patience constraint of MCDLP-NR, i.e., constraint (19c), is required. Namely, we replace the patience level of each customer \( j \), i.e., \( \ell_j \), by its expected patience level \( 1/p^{\text{out}} \). The modified MCDLP-NR is demonstrated below:

\[
\max \sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \in \mathcal{S}} x_j(S) \sum_{i \in S} r_{ij} p_j(i, S) \\
\sum_{t=1}^{T} \sum_{j=1}^{m} q_{tj} \sum_{S \in \mathcal{S}; S \ni j} x_j(S) p_j(i, S) \leq 1 \quad \forall i = 1, \ldots, n \tag{71a}
\]

\[
\sum_{S \in \mathcal{S}} x_j(S) \sum_{i \in S} p_j(i, S) \leq 1 \quad \forall j = 1, \ldots, m \tag{71b}
\]
\[ \sum_{S \in \mathcal{S}} x_j(S) \leq \frac{1}{p^\text{out}_j} \quad \forall j = 1, \ldots, m \quad (71c) \]
\[ \sum_{S \in \mathcal{S}} x_j(S) \leq 1 \quad \forall i = 1, \ldots, n; \forall j = 1, \ldots, m \quad (71d) \]
\[ x_j(S) \geq 0 \quad \forall j = 1, \ldots, m; \forall S \in \mathcal{S} \quad (71e) \]

To prove the theorem, we need to show that Lemmas 9-11 for this variant of the problem. This can be easily checked for Lemma 9 as it provides an upper-bound on the probability that an item \( i \) is already matched when a customer of type \( j \) arrives for the first time and the proof of the lemma does not depend on patience level of customers. Similarly, Lemma 10 stays untouched for non-deterministic patience levels as it does not depend on patience levels either.

The only difficulty comes from Lemma 11. In the proof of this lemma two cases are considered for a customer of type \( j \), those are, (i) patience level of 1, and (ii) patience level of greater than 1. In non-deterministic patience levels, these two cases translate to (i) type \( j \) customer leaves the platform if no purchase is made from the first offered assortment, (ii) type \( j \) customer sees the second assortment if she does not make a purchase from the first assortment. Case (i) takes place with probability \( p^\text{out}_j \) and with an argument similar to the one provided in the proof of Lemma 11 it can easily be seen that the probability of this case is upper-bounded by \( 1/(2\alpha) \). Case (ii) takes place with probability \( 1 - p^\text{out}_j \). In this case, by union bound we have

\[ \Pr[\text{Timeout}_{S}(j) \cup \text{CMatch}_{S}(j)|\text{Type}(j)] \leq \Pr[\text{CMatch}_{S}(j)|\text{Type}(j)] + \Pr[\text{Timeout}_{S}(j)|\text{Type}(j)] \]

and similar to the proof of Lemma 11 it can be seen that \( \Pr[\text{CMatch}_{S}(j)|\text{Type}(j)] \leq 1/(2\alpha) \). In the remainder of this proof we show that

\[ \Pr[\text{Timeout}_{S}(j)|\text{Type}(j)] \leq (1 - p^\text{out})2/(3\alpha^2) \]

providing \( \Pr[\text{Timeout}_{S}(j) \cup \text{CMatch}_{S}(j)|\text{Type}(j)] \leq 1/(2\alpha) + 2/(3\alpha^2) \) similar to Lemma 11.

To see this, suppose customer type \( j \) has a patience level of \( a \), that is she sees up to \( a \) assortment. This happens with probability \( (1 - p^\text{out}_j)^{a-1} \cdot p^\text{out}_j \) as all customers see at least one assortment. Therefore we have

\[ \Pr[\text{Timeout}_{S}(j)|\text{Type}(j)] = \sum_{a=2}^{\infty} \Pr[\text{Timeout}_{S}(j)|\text{Type}(j), l_j = a] \Pr[l_j = a] \]
\[ \leq \sum_{a=2}^{\infty} \frac{1}{(a + 1)!} \left( \frac{a}{\alpha} \right)^a (1 - p^\text{out}_j)^{a-1} p^\text{out}_j \]
\[ \leq \frac{2}{3\alpha^2} \cdot p^\text{out}_j \sum_{a=2}^{\infty} (1 - p^\text{out}_j)^{a-1} \]
\[ \leq \frac{2}{3\alpha^2} (1 - p^\text{out}_j), \]

where (72) uses (31), (73) uses bound \( \frac{1}{(a + 1)!} \left( \frac{a}{\alpha} \right)^a \leq \frac{2}{3\alpha^2} \) discussed in the proof of Lemma 11, and finally (74) uses sum of geometric series. In conclusion, here we showed that proofs of Lemmas 9-11 hold in case of non-deterministic patience levels, hence Algorithm 3 has the same approximation ratio for this variant of the problem. □

**Proof of Theorem 7.** Let \( x^* \) be the optimal solution to the MCDLP-NRS. As customers come their patience levels are revealed to the platform and up to their patience level or once a purchase is made by them (whichever comes first), the algorithm offers them a random permutation of assortments. For an assortment \( S \) to be offered to customer \( t \), items that are already seen by that customer in the previous offering stages
and those that are sold-out are first removed from it and then with probability \(x^*(S)/\alpha\) it is shown to the customer. Theorem 5 shows that for \(\alpha = (3 + \sqrt{11})/2\) Algorithm 3 has a 0.093 approximation guarantee for the case that items can have different revenues across different customer types. Here, we show that the modified Algorithm 3 has an improved approximation guarantee of 0.15 when for each item all customer types are associated with the same revenue.

Let \(U_t\) denote the set of available items at the beginning of time-step \(t\). The modified Algorithm 3 offers an item \(i \in U_t\) along with assortment \(S \ni i\) to customer \(t\) with probability at least \((1 - 3/(2\alpha))/\alpha \cdot \sum_{j=1}^{m} q_{i,j} x_j^*(S)\). Note that this is different from the guarantee presented in Lemma 8 by an additive term of \((-2/(3\alpha^2))\) as here it is assumed that item \(i\) is not sold-out yet. The term \((1 - 3/(2\alpha))/\alpha\) is a concave function of \(\alpha\) and by simple calculus it can be seen that it is maximized for \(\alpha = 3\), which makes \((1 - 3/(2\alpha))/\alpha\) equal to 1/6.

Therefore, the modified Algorithm 3 offers an item \(i \in U_t\) along assortment \(S \ni i\) with probability at least \(\sum_{j=1}^{m} q_{i,j} x_j^*(S)/6\).

Let \(p_{it}\) denote the probability that item \(i\) is purchased by customer \(t\) by the optimal solution of the MCDLP-NRS, that is, \(p_{it} = \sum_{j=1}^{m} q_{i,j} \sum_{S \ni i} x_j^*(S)p_j(i,S)\). Constraint (19a) states that for each item \(i\) we have \(\sum_{j=1}^{T} \sum_{q \in S \ni i} x_j^*(S)p_j(i,S) \leq 1\). In other words,

\[
\sum_{t=1}^{T} p_{it} \leq 1, \tag{75}
\]

which will be later used to establish how far the total expected revenue of the modified Algorithm 3 is from the optimal value of the MCDLP-NRS.

Let \(\text{Accept}_t(i)\) be the event that item \(i\) is purchased by a customer \(t\) and \(\text{Accept}(i)\) be the event that \(i\) is sold to a customer while performing the modified Algorithm 3. We have

\[
\Pr\{\text{Accept}(i)\} \geq 1 - \prod_{t=1}^{T} (1 - \Pr\{\text{Accept}_t(i)\}). \tag{76}
\]

Since the modified Algorithm 3 offers an item \(i \in U_t\) to customer \(t\) as a part of an assortment \(S\) with probability at least \(\sum_{j=1}^{m} q_{i,j} x_j^*(S)/6\), we have \(\Pr\{\text{Accept}_t(i)\} \geq \sum_{j=1}^{m} q_{i,j} \sum_{S \ni i} x_j^*(S)\cdot p_j(i,S)/6\), that is, \(p_{it}/6\). Thus,

\[
1 - \prod_{t=1}^{T} (1 - \Pr\{\text{Accept}_t(i)\}) = 1 - \prod_{t=1}^{T} (1 - \frac{p_{it}}{6}) \geq 1 - \prod_{t=1}^{T} \exp(-\frac{p_{it}}{6}) \tag{77}
\]

\[
= 1 - \exp(-\sum_{t=1}^{T} \frac{p_{it}}{6}), \tag{78}
\]

where (77) uses the bound \(1 - p_{it}/6 \leq \exp(-p_{it}/6)\) resulted from the Taylor expansion of \(\exp(-p_{it}/6)\). Combining (76) and (78) we have

\[
\Pr\{\text{Accept}(i)\} \geq 1 - \exp(-\sum_{t=1}^{T} \frac{p_{it}}{6}). \tag{79}
\]

Now, let \(E[\text{ALG}(i)]\) and \(\text{OPT}(i)\) denote the expected revenue that the modified Algorithm 3 and the optimal solution of the MCDLP-NRS collect from item \(i\), respectively. Moreover, let \(E[\text{ALG}]\) and \(\text{OPT}\) denote the total expected revenue of the algorithm and the optimal value of the MCDLP-NRS, respectively. Therefore,
\[ \mathbb{E}[\text{ALG}] = \sum_{i=1}^{n} \mathbb{E}[\text{ALG}(i)] \text{ and } \text{OPT} = \sum_{i=1}^{n} \text{OPT}(i). \]

If we show that there exists a positive constant \( c \leq 1 \) such that for all \( i = 1, \ldots, n \) we have \( \mathbb{E}[\text{ALG}(i)] \geq c \cdot \text{OPT}(i) \), then it implies that \( \mathbb{E}[\text{ALG}] \geq c \cdot \text{OPT} \), establishing an approximation guarantee of \( c \) for the modified Algorithm 3. We show that such \( c \) exists and it is equal to 1/6. To see that, observe that \( \mathbb{E}[\text{ALG}(i)] = r_i \cdot \Pr[\text{Accept}(i)] \) and \( \text{OPT}(i) = r_i \sum_{t=1}^{T} p_{it} \). This in addition to (79) gives

\[ \frac{\mathbb{E}[\text{ALG}(i)]}{\text{OPT}(i)} \geq \frac{r_i (1 - \exp(-\sum_{t=1}^{T} \frac{p_{it}}{6}))}{r_i \sum_{t=1}^{T} p_{it}} = \frac{1 - \exp(-\sum_{t=1}^{T} \frac{p_{it}}{6})}{\sum_{t=1}^{T} \frac{p_{it}}{6}} = \frac{\sum_{t=1}^{T} \frac{p_{it}}{6}}{6}, \]

where, \( f(z) = (1 - \exp(-z))/z \). By Lemma 3, \( f(z) \) is a decreasing function. Furthermore, (75) states that \( \sum_{t=1}^{T} p_{it} \leq 1 \) for all items \( i = 1, \ldots, n \). Therefore

\[ \frac{\mathbb{E}[\text{ALG}(i)]}{\text{OPT}(i)} \geq \frac{f(\frac{1}{6})}{6} = 1 - \exp(-\frac{1}{6}) \approx 0.15, \]

that is, \( c = 0.15 \) and the modified Algorithm 3 has an approximation guarantee of at least 0.15. \( \square \)

**Proof of Theorem 8.** Fix a large even integer \( M \), and consider a family of assortments \( S' \) with \( |S'| = M \), constructed as follows. Every pair of assortments in \( S' \) share exactly 1 item, so that there are \( \frac{M(M-1)}{2} \) items in total. Each assortment \( S \in S' \) contains exactly \( M - 1 \) items, since it shares exactly one item with each of exactly \( M - 1 \) other assortments. The patience level of the customer is \( \ell = \frac{M}{2} \). The price of each item is 1.

The feasible family \( S \) of assortments that can be offered is the downward closure of \( S' \). That is, an assortment \( S \) can be offered if an only if \( S \subseteq S' \) for some \( S' \in S' \). When offered any assortment \( S \in S \), the customer chooses each item in \( S \) with probability \( \frac{2}{M(M-1)} \), and chooses to purchase nothing with probability \( 1 - \frac{2}{M(M-1)} \). We remark that if \( |S| = M - 1 \) (the maximum possible size of an assortment \( S \in S \)), then the customer makes a purchase with probability \( \frac{2}{M(M-1)} \). It is easily checked that these choice probabilities satisfy the substitutability assumption.

Set \( x(S) = 1/2 \) for each assortment \( S \in S' \). We claim that this is a feasible solution to the single-customer MCDLP-NRS. Indeed, constraint (32a) holds because the LHS equals \( M \times \frac{1}{2} \times \frac{2}{M} \) by the remark above; constraints (32b) and (32d) hold trivially; while constraint (32c) holds because each item appears in exactly two assortments. Therefore, the optimal objective value of the MCDLP-NRS is at least the value of this solution, which is 1.

Meanwhile, consider any collection \( S^{\text{Alg}} \) of assortments which an online algorithm could plan to offer the customer. \( S^{\text{Alg}} \) cannot contain more than \( M/2 \) assortments (because the patience level is \( M/2 \)), and the order in which the assortments in \( S^{\text{Alg}} \) are offered does not affect the revenue (because all of the items have a price of 1). Moreover, the assortments in \( S^{\text{Alg}} \) cannot overlap, by the constraint that the same item cannot be shown twice. For any assortment \( S \in S \), let \( \overline{S} \) denote any \( S' \supseteq S \) with \( S' \in S' \) (i.e. \( |\overline{S}| = |S'| = M - 1 \)). Then, we have

\[ \sum_{S \in S^{\text{Alg}}} |\overline{S}| \leq \bigcup_{S \in S^{\text{Alg}}} |\overline{S}| \]  

(80)

where we note that the assortments \( \overline{S} \) could overlap. Now, let \( M' \) denote the number of unique values of \( \overline{S} \) over \( S \in S^{\text{Alg}} \), with \( M' \leq |S^{\text{Alg}}| \leq M/2 \). By the principle of inclusion-exclusion, the right-hand side of (80)
equals $M'(M - 1) - \binom{M'}{2}$, because $|\mathcal{S}| = M - 1$, the intersection of every two assortments has cardinality exactly 1, and the intersection of 3 or more assortments has cardinality 0. This value is at most
\[
\frac{M(M - 1)}{2} - \frac{M/2(M/2 - 1)}{2} \leq \frac{3M^2}{8},
\]
and hence $\sum_{S \in \mathcal{S}^\mathcal{S}_k} |S| \leq \frac{3M^2}{8}$.

Now, since the probability of an individual assortment $S$ earning a sale is $\frac{2|S|}{M(M - 1)}$, the total probability of the online algorithm earning a sale is
\[
1 - \prod_{S \in \mathcal{S}^\mathcal{S}_k} \left(1 - \frac{2|S|}{M(M - 1)}\right) \leq 1 - \prod_{S \in \mathcal{S}^\mathcal{S}_k} \left(\exp\left(-\frac{2|S|}{M(M - 1)}\right) - \frac{1}{2}\left(\frac{2|S|}{M(M - 1)}\right)^2\right)
\leq 1 - \prod_{S \in \mathcal{S}^\mathcal{S}_k} \exp\left(-\frac{2|S|}{M(M - 1)}\right) + \sum_{S \in \mathcal{S}^\mathcal{S}_k} \frac{1}{2}\left(\frac{2|S|}{M(M - 1)}\right)^2
\leq 1 - \exp\left(-\frac{2\sum_{S \in \mathcal{S}^\mathcal{S}_k} |S|}{M(M - 1)}\right) + \sum_{S \in \mathcal{S}^\mathcal{S}_k} \frac{2}{M^2}
\leq 1 - \exp\left(-\frac{3M^2}{4M(M - 1)}\right) + \left(\frac{M}{2}\right)^2 \frac{2}{M^2}
\]
where the first inequality holds because $e^{-x} \leq 1 - x + \frac{x^2}{2}$ for $x \geq 0$, the second inequality holds because each term in the product is at most 1, the third inequality holds because $|S| \leq M - 1$, and the final inequality holds because $\sum_{S \in \mathcal{S}^\mathcal{S}_k} |S| \leq \frac{3M^2}{8}$ and $|\mathcal{S}^\mathcal{S}_k| \leq M/2$. The final expression can be checked to be approach $1 - \exp(-\frac{3}{4})$ as $M \to \infty$, completing the proof. \(\square\)

Appendix C: Supplements to Section 5

In the following, we discuss the column generation technique for the case of the multi-stage and multi-customer assortment optimization problem without repeated item offerings. We later finish this section by discussing how to tackle the problem when repeated offerings of items is allowed. Suppose $\zeta$, $\gamma$, $\beta$ and $\sigma$ are the dual variables for constraints (19a)-(19d), respectively. We refer to the dual of the MCDLP-NR by MCDLP-NR-D and for both MCDLP-NR and MCDLP-NR-D, we use the notations MCDLP-NR($\mathcal{S}'$) and MCDLP-NR-D($\mathcal{S}'$) to specify that only assortments from set $\mathcal{S}'$ are considered for these linear programs.

Below we present MCDLP-NR-D($\mathcal{S}'$) for a set $\mathcal{S}' \subseteq \mathcal{S}$.

\[
\min \sum_{i=1}^{n} \zeta_i + \sum_{j=1}^{m} (\gamma_j + \ell_j \beta_j) + \sum_{i=1}^{n} \sum_{j=1}^{m} \sigma_{ij}
\]
\[
\gamma_j \sum_{i \in \mathcal{S}} p_j(i, S) + \beta_j \sum_{i \in \mathcal{S}} \sigma_{ij} + \sum_{i \in \mathcal{S}} \sum_{t=1}^{T} \zeta_i \sum_{j \in \mathcal{M}} q_{ij} p_j(i, S) \geq \sum_{i \in \mathcal{S}} \sum_{t=1}^{T} r_{ij} \sum_{j \in \mathcal{M}} q_{ij} p_j(i, S) \quad \forall j \in [m]; \forall S \in \mathcal{S}'
\]
\[
\zeta, \gamma, \beta, \sigma \geq 0
\]

The column generation technique starts with a set of assortments $\mathcal{S}^1$ and solves MCDLP-NR-D($\mathcal{S}^1$) to find the optimal dual variables ($\eta^1, \gamma^1, \beta^1, \sigma^1$) for this set assortments. We then check whether the dual variables corresponding to MCDLP-NR-D($\mathcal{S}^1$) are also feasible for MCDLP-NR-D($\mathcal{S}$). In other words, we need to specify whether there exists any assortment in $\mathcal{S} \setminus \mathcal{S}^1$ for which constraint (81a) is violated, i.e., has
a positive reduced cost. To find any such assortment we can first solve the following problem, called the column generation subproblem for any customer type $j$:

$$
\max_{i \in S} \sum_{i \in S} \left( r_{ij}^1 p_j(i, S) - \zeta_{ij} \right),
$$

(82)

where $w_{ij}^1 = r_{ij} \sum_{t=1}^T g_{ij} - \zeta_{ij} \sum_{t=1}^T g_{ij} - \gamma_{ij}$. Let $S^1$ denote the solution of the above problem. Note that $\sum_{i \in S} \left( w_{ij}^1 p_j(i, S^1) - \sigma_{ij}^1 \right) \geq 0$ by setting $S^1 = \emptyset$. The goal is to check whether $\sum_{i \in S} \left( w_{ij}^1 p_j(i, S^1) - \sigma_{ij}^1 \right) > \beta_j^1$. In this case, we set $S^2 = S^1 \cup \{S^1\}$ and repeats this process again; otherwise, the process stops. Suppose the process stops at iteration $\tau$. By solving MCDLP-NR(S$^\tau$) we get its optimal solution $\bar{x}$ that has the same objective value as that of MCDLP-NR(S).

In any $\tau$-th round of this procedure, by constraint (81b) of MCDLP-NR-D(S$^\tau$), we have $\beta_j^\tau, \sigma_{ij}^\tau \geq 0$. Therefore, subproblem (82) is only interesting when $w_{ij}^\tau \geq 0$. If for the solution of this subproblem’s, denoted by $S^\tau$, we have $\sum_{i \in S} \left( w_{ij}^\tau p_j(i, S^\tau) - \sigma_{ij}^\tau \right) \leq \beta_j^\tau$, then $(\zeta^\tau, \gamma^\tau, \beta^\tau, \sigma^\tau)$ are dual feasible for MCDLP-NR-D(S) and the current solution to the reduced MCDLP-NR is in fact optimal. Otherwise, $S^\tau$ has a positive reduced cost. By adding $S^\tau$ to $S^\tau$ we resolve the reduced MCDLP-NR of $S^\tau$, which results in a larger value of objective function of the MCDLP-NR. This process is continued until subproblem (82)’s solution is the empty set, at which point the MCDLP-NR gives us the optimal solution.

In general solving column generation subproblems are NP-hard (Liu and Van Ryzin 2008). However, there is hope to find approximate solutions for it for specific choice models. Combining an approximation algorithm with the aforementioned column generation framework provides the approximate column generation technique. In other words, we want to find a set $S^* \in S \setminus S^\tau$ such that

$$
\sum_{i \in S} \left( w_{ij}^* p_j(i, S^*) - \sigma_{ij}^* \right) \geq \alpha \cdot \max_{S \in S} \sum_{i \in S} \left( w_{ij}^1 p_j(i, S) - \sigma_{ij}^1 \right).
$$

(83)

In approximate column generation, we still terminate when the subproblem algorithm fails to find a constraint violating dual feasibility. Since the subproblem algorithm is $\alpha$-approximate, we can guarantee that when it terminates, the dual constraints are within an $\alpha$-factor of being feasible.

Since the empty set is a feasible solution to problem (88), the optimal value of problem (88) is always non-negative. Moreover, the problem at hand is a maximization problem; thus, we have $\alpha \in [0, 1]$. Before discussing any such approximation algorithm, let us introduce a few notations. For a set of assortments $S' \subseteq S$, let $OPT(MCDLP-NR(S'))$ and $OPT(MCDLP-NR-D(S'))$ denote the optimal values of MCDLP-NR(S') and MCDLP-NR-D(S'), respectively. Moreover, we denote the optimal solution of the overall MCDLP-NR, that is MCDLP-NR(S), by $x^*$ and the optimal solution of the overall dual of MCDLP-NR, that is MCDLP-NR-D(S), by $(\zeta^*, \gamma^*, \beta^*, \sigma^*)$. Furthermore, for $\bar{x}$, a feasible solution to MCDLP-NR(S), let $Val(\bar{x}) = \sum_{j=1}^m \sum_{i \in S} \bar{x}_j(S) \sum_{r_{ij}} r_{ij} p_j(i, S)$. Lemma 14 discusses how the approximation ratio of an approximation algorithm for the column generation subproblem and that of the MCDLP-NR(S) when the approximate column generation technique is used are related. The proof of this lemma is similar to the proof of Lemma 3.5 in (Cheung and Simchi-Levi 2016) and effectively extends the result from CDLP’s to our MCDLP’s.
Lemma 14. Consider the multi-stage choice-based deterministic linear program with no repeated item offerings (MCDLP-NR) and suppose an algorithm \( A \) with approximation guarantee of \( \alpha \) for the column generation subproblem (i.e., subproblem (82)) on the set of assortments \( S \) and the underlying choice model \( p_j(\ldots) \) for all customer types \( j \). Then, the approximate column generation technique returns a solution \( \bar{x} \) to MCDLP-NR(S) whose objective value is at least \( \alpha \) times away from \( \text{OPT}(\text{MCDLP-NR}(S)) \), that is,

\[
\text{Val}(\bar{x}) \geq \alpha \text{Val}(x^*) \quad (84)
\]

Proof. First of all, note that the approximate column generation technique halts in finite amount of time, after at most \( |S| \) iterations. In any iteration \( \tau \) of the approximate column generation technique, other than its last iteration, an assortment \( S^r \in S \setminus S^r \) is added to \( S^r \) to make \( S^{r+1} \). To see that \( S^r \) is in fact in \( S \), note that in the \( \tau \)-th iteration MCDLP-NR-D(\( S^r \)) is solved. Therefore, for a feasible solution to MCDLP-NR-D(\( S^r \)) constraint (81a) is satisfied for all assortments in \( S^r \); thus, on set \( S^r \) subproblem (82) has no solution \( S \).

We now show that if subproblem (82) can be solved with an \( \alpha \) approximation guarantee, then the approximate column generation technique returns a solution for MCDLP-NR(S) whose value is within an \( \alpha \) factor of \( \text{OPT}(\text{MCDLP-NR}(S)) \). Suppose that the approximate column generation technique halts in the \( \tau \)-th iteration, providing \((\zeta^r, \gamma^r, \beta^r, \sigma^r)\), the optimal dual variables for MCDLP-NR-D(\( S^r \)). By constraint (81a), for all customer types \( j \) and assortments \( S^r \), the assortment returned by Algorithm \( A \) in the \( \tau \)-th iteration, we have

\[
\beta^r_{ij} \geq \sum_{i \in S^r} (w^r_{ij} p_j(i, S^r) - \sigma^r_{ij}),
\]

where \( w^r_{ij} = r_{ij} - \zeta^r - \gamma^r \). Moreover, since algorithm \( A \) has an \( \alpha \) approximation ratio, combing with the above it gives

\[
\beta^r_{ij} \geq \sum_{i \in S^r} (w^r_{ij} p_j(i, S^r) - \sigma^r_{ij}) \geq \alpha \cdot \max_{S \subseteq S} \sum_{i \in S} (w^r_{ij} p_j(i, S) - \sigma^r_{ij}).
\]

Observe that not only \((\zeta^r, \gamma^r, \beta^r, \sigma^r)\) are the optimal decision variables for MCDL-NRP-D(\( S^r \)), they are also feasible decision variables for MCDL-NRP-D(S) as constraints (81a) and (81b) are satisfied for them. Otherwise, the approximate column generation technique would have not stopped at the \( \tau \)-th iteration. Feasibility of \((\zeta^r, \gamma^r, \beta^r/\alpha, \sigma^r)\) and the fact that \( \alpha \leq 1 \) implies that \((\zeta^r, \gamma^r, \beta^r/\alpha, \sigma^r)\) is also a feasible to MCDLP-NR-D(S).

Thus,

\[
\sum_{i=1}^{n} \zeta^r_i + \sum_{j=1}^{m} (\gamma^r_j + \ell_j \beta^r_{ij}) + \sum_{i=1}^{n} \sum_{j=1}^{m} \sigma^r_{ij} \geq \sum_{i=1}^{n} \zeta^r_i + \sum_{j=1}^{m} (\gamma^r_j + \ell_j \beta^r_{ij}) + \sum_{i=1}^{n} \sum_{j=1}^{m} \sigma^r_{ij} \geq \sum_{j=1}^{m} \sum_{S \subseteq S} x^*_j(S) \sum_{i \in S} r_{ij} p_j(i, S),
\]

where (85) is due to the fact that the dual of the MCDLP-NR is a minimization problem and \((\zeta^r, \gamma^r, \beta^r/\alpha, \sigma^r)\) is feasible solution for MCDLP-NR-D(S), and (86) holds by strong duality for MCDLP-NR(S) and MCDLP-NR-D(S). Furthermore, using strong duality for MCDLP-NR(\( S^r \)) and MCDLP-NR-D(\( S^r \)) we have

\[
\frac{1}{\alpha} \sum_{j=1}^{m} \sum_{S \in S^r} \bar{x}^*_j(S) \sum_{i \in S} r_{ij} p_j(i, S) = \frac{1}{\alpha} \sum_{i=1}^{n} \zeta^r_i + \sum_{j=1}^{m} (\gamma^r_j + \ell_j \beta^r_{ij}) + \sum_{i=1}^{n} \sum_{j=1}^{m} \sigma^r_{ij} \geq \sum_{i=1}^{n} \zeta^r_i + \sum_{j=1}^{m} (\gamma^r_j + \ell_j \beta^r_{ij}) + \sum_{i=1}^{n} \sum_{j=1}^{m} \sigma^r_{ij},
\]

(87)
where $\tilde{x}^*$ is the optimal solution for MCDLP-NR($S'$) that is padded with zeros for the assortments that in $S \backslash S'$; therefore, it is a feasible solution for MCDLP-NR($S$). Combining (86) and (87) and using the fact that as $\tilde{x}^*$ is a feasible solution to a maximization problem we have

$$\frac{\text{Val}(x^*)}{\alpha} = \frac{1}{\alpha} \sum_{j=1}^{m} \sum_{S \in S} x_j^*(S) \sum_{i \in S} r_{ij} p_j(i, S) \geq \frac{1}{\alpha} \sum_{j=1}^{m} \sum_{S \in S'} x_j^*(S) \sum_{i \in S} r_{ij} p_j(i, S) = \frac{\text{Val}(*\tilde{x})}{\alpha} \geq \sum_{j=1}^{m} \sum_{S \in S} x_j^*(S) \sum_{i \in S} r_{ij} p_j(i, S) = \text{Val}(x^*),$$

concluding that $\text{Val}(\tilde{x}^*) \geq \alpha \text{Val}(x^*)$, as desired. □

Knowing how the approximation ratio of the column generation subproblem translates to that of the MCDLP-NR, it is then of interest to find an approximation algorithm for this subproblem. In the next section, we discuss an approximation algorithm for this problem in the case of the MNL choice model.

C.1. Specific Results for MNL Choice Model

As discussed earlier, it is known to be NP-hard to solve the column generation subproblem for general choice models. For some restricted choice models, such as multinomial logit (MNL), the column generation subproblem has been shown to be polynomially solvable for many different CDLP’s, see Liu and Van Ryzin (2008) and Talluri and Van Ryzin (2004). Therefore, it is of interest to determine whether for the MCDLP-NR the column generation subproblem can be efficiently solved. Subproblem (82) for multinomial logit choice model and a fixed customer type $j$ is presented below

$$\max_{S \in S} \sum_{i \in S} \left( \frac{w_{ij} v_{ij}}{\sum_{k \in S} v_{kj} + 1} - \sigma_{ij}^* \right). \quad (88)$$

Lemma 15 addresses whether problem (88) can be solved in polynomial-time, i.e., whether the column generation subproblem is polynomially solvable when the choice model is multinomial logit. To evaluate the complexity of an optimization problem we need to study its decision problem as the complexity class NP is defined for decision problems. A decision problem is a problem with “Yes” or “No” answers. Thus, the decision version of problem (88) can be

$$\max_{S \in S} \sum_{i \in S} \left( \frac{w_{ij} v_{ij}}{\sum_{k \in S} v_{kj} + 1} - \sigma_{ij}^* \right) \geq \lambda, \quad (89)$$

where $\lambda$ is a real positive number.

**Lemma 15.** Problem (89) is an NP-complete problem.

**Proof.** First of all, note that problem (88) is in complexity class NP as a “Yes” certificate for it can be verified in polynomial-time. A “Yes” certificate is an assortment $S \in S$ for which $\sum_{i \in S} \left( w_{ij} v_{ij} / (\sum_{k \in S} v_{kj} + 1) - \sigma_{ij}^* \right) \geq \lambda$. Hence, it remains to show that problem (89) is an NP-hard problem. Motivated by (Liu et al. 2019a, Theorem 1), we prove the NP-hardness of the problem via a reduction from the partition problem, a well-known NP-hard problem (Garey and Johnson 2002).

Let us start by introducing the partition problem. In this problem we have $M$ integers, $c_1, \ldots, c_M$, such that $\sum_{k=1}^{M} c_k = 2t$. The goal of the partition problem is to find a subset of these integers, denoted by $S$, such that the sum of integers in this set is half of the total sum of all the integers at hand, that is, $\sum_{k \in S} c_k = t$.
We provide a reduction from the partition problem to problem (89) by constructing an instance of the problem (89) for any instance of the partition problem such the instance of problem (89) has a solution if and only if the instance of the partition problem is partitionable.

For a given instance of the partition problem we construct the instance of problem (89) as follows. Since problem (89) is for a fixed customer type $j$ and iteration $\tau$, we drop the indices $j$ and $\tau$ for the rest of this proof. We have $M$ items such that for each item $1 \leq i \leq M$ we have $w_i = 1$, $v_i = c_i/t$ and $\sigma_i = c_i/(4t)$; moreover, we set $\lambda = 1/4$. Therefore, problem (89) becomes

$$\max_{\mathcal{S} \in \mathcal{S}} \sum_{i \in \mathcal{S}} \left( \frac{c_i}{4} \frac{1}{t} + 1 - \frac{c_i}{4t} \right) \geq \frac{1}{4}.$$  

After multiplying both sides of the above by $\sum_{k \in \mathcal{S}} \frac{c_i}{4} + 1$ and simplification we have

$$\max_{\mathcal{S} \in \mathcal{S}} \frac{3}{4} \sum_{i \in \mathcal{S}} c_i - \frac{(\sum_{i \in \mathcal{S}} c_i)^2}{4t} - \frac{1}{2} \sum_{i \in \mathcal{S}} c_i \geq \frac{t}{4}. \tag{90}$$

The left-hand side of (90) is quadratic in $\sum_{i \in \mathcal{S}} c_i$; thus, it can be bounded from above. Setting $y = \sum_{i \in \mathcal{S}} c_i$, we can write the left-hand side of (90) as $3y/4 - y^2/(4t) - y/4$. Taking derivative with respect to $y$, it can be seen that $3y/4 - y^2/(4t) - y/4$ achieves its maximum at $y = t$ and the maximum at this $y$ is $t/4$. Therefore, we have the left-hand side of (90) is also upper-bounded by $t/4$. Since the upper and lower-bound match, for this instance of problem (89) it has to be that $\frac{3}{4} \sum_{i \in \mathcal{S}} c_i - \frac{(\sum_{i \in \mathcal{S}} c_i)^2}{4t} - \frac{1}{2} \sum_{i \in \mathcal{S}} c_i = \frac{1}{4}$, which happens only if $\sum_{i \in \mathcal{S}} c_i = t$. Recall that the goal of the partition problem was to find a subset of integers $c_1, \ldots, c_M$ that sum to $t$. In other words, this instance of problem (89) is solvable if and only if we can also solve the partition problem on integers $c_1, \ldots, c_M$. As partition problem is NP-hard, this proves that problem (89) is also an NP-hard problem. \(\square\)

Despite the column generation subproblem being NP-hard even for the basic MNL choice model, we derive an FPTAS (the best possible approximation for an NP-hard problem) for it via the knapsack problem. Using the knapsack problem to design an FPTAS approximation algorithms for assortment optimization problems under the MNL choice model has been utilized previously, e.g. Désir et al. (2014) and Liu et al. (2019a). We in fact use the algorithm by Désir et al. (2014) as a subroutine in our algorithm.

As problem (88) is solved for each iteration $\tau$ and customer type $j$, in the following we omit these two indices to refer to a general instance of this problem. Suppose $S^*$ is the optimal solution to problem (88). Our algorithm is based on constructing a polynomial number of knapsack problems where for each one of them a polynomial number of dynamic programs are solved. More precisely, we first guess the value of $\sum_{i \in S^*} \sigma_i$ within a factor of $(1 + \epsilon)$ for some small $\epsilon > 0$. Using this we then reformulate the problem as a capacity constrained assortment optimization problem. Désir et al. (2014) have shown that the latter problem is known to have an FPTAS approximation algorithms by using the FPTAS algorithm for the knapsack problem, which we later discuss how to utilize.

Let $\sigma$ (resp. $\bar{\sigma}$) be the minimum (resp. maximum) among all $\sigma_i$’s. For a given $\epsilon$ we use the following guesses for $\sum_{i \in S^*} \sigma_i$

$$\Phi^\epsilon = \{ \sigma(1 + \epsilon)^{l_1} : l_1 = 0, \ldots, L_1 \}, \tag{91}$$
where $L_1 = O(\log(n\sigma/\epsilon)/\epsilon)$, ensuring that the maximum value in $\Phi$ is $n\sigma$, which is an upper-bound on $
_{i\in S^*} \sigma_i$; hence, we cover the whole search space. Note that the number of guesses for $\n_{i\in S^*} \sigma_i$ is polynomial in the input size $n$ and $1/\epsilon$. For a fixed $\epsilon$ and each $\phi \in \Phi_\epsilon$, problem (88) turns into

$$
\max_{S \in S} \sum_{i \in S} \frac{w_i v_i}{\sum_{k \in S} v_k + 1} - \sum_{i \in S} \sigma_i \leq \phi.
$$

(92)

For problem (92), we use the FPTAS for the capacity constrained assortment optimization with MNL choice model problem by Désir et al. (2014), presented in Algorithm 7. We then pick the $\phi$ to which an assortment $S$ that maximizes $\frac{\sum_{i \in S} w_i v_i}{\sum_{i \in S} v_i + 1} - \sum_{i \in S} \sigma_i$ is associated, see Algorithm 6.

**Algorithm 6** FPTAS for Problem (88)

**INPUT:** $w_i, v_i, \sigma_i$ for $i = 1, \ldots, n$ and $\epsilon$

1: for $\phi \in \Phi_\epsilon$ do

2: Solve Algorithm 7 with $\phi$ as its input and let the solution be $S_{\gamma,\delta,\phi}$.

3: return assortment $S$ that maximizes $\frac{\sum_{i \in S} w_i v_i}{\sum_{i \in S} v_i + 1} - \sum_{i \in S} \sigma_i$ over all assortments $\{S_{\gamma,\delta,\phi} : \gamma \in \Gamma_\epsilon, \delta \in \Delta_\epsilon, \phi \in \Phi_\epsilon\}$.

Here, we briefly discuss the mechanism of the FPTAS for problem (92), see Désir et al. (2014) for a thorougher discussion. For a given value of $\phi$, the algorithm first makes guesses for the values of $\sum_{i \in S^*} w_i v_i$ and $\sum_{i \in S^*} v_i$. Let $\tilde{w}$ (resp. $\tilde{v}$) be the minimum (resp. maximum) among all $w_i$’s and $v$ (resp. $V$) be the minimum (resp. maximum) among all $v_i$’s. For a choice of $\epsilon$ we use the following guesses for $\sum_{i \in S^*} w_i v_i$ and $\sum_{i \in S^*} v_i$

$$
\Gamma_\epsilon = \left\{\tilde{w}v(1+\epsilon)^2 : l_2 = 0, \ldots, L_2\right\}, \quad \text{and} \quad \Delta_\epsilon = \left\{\tilde{v}(1+\epsilon)^3 : l_3 = 0, \ldots, L_3\right\},
$$

(93)

where $L_2 = O(\log(n\tilde{w}v^2)/(\epsilon\tilde{w}))$ and $L_3 = O(\log(n\tilde{v}^3)/(\epsilon\tilde{v}))$, that is, the number of guesses for $\sum_{i \in S^*} w_i v_i$ and $\sum_{i \in S^*} v_i$ is polynomial in the input size $n$ and $1/\epsilon$. Moreover, the choice of $L_2$ and $L_3$ ensures that the maximum values in $\Gamma_\epsilon$ and $\Delta_\epsilon$ are $n\tilde{w}v^2$ and $n\tilde{v}^3$, respectively, which upper-bound $\sum_{i \in S^*} w_i v_i$ and $\sum_{i \in S^*} v_i$; hence, the whole search space is covered. For each $\gamma \in \Gamma_\epsilon$ and $\delta \in \Delta_\epsilon$, the following discretizations are implemented

$$
\tilde{w}_i = \left\lfloor \frac{nw_i v_i}{\epsilon \gamma} \right\rfloor, \quad \text{and} \quad \tilde{v}_i = \left\lfloor \frac{nv_i}{\epsilon \delta} \right\rfloor.
$$

(94)

For each $a \in [J]$ and $b \in [J]$, a dynamic program is then used to find an assortment $S$ such that

$$
\sum_{i \in S} \tilde{w}_i \geq a, \quad \sum_{i \in S} \tilde{v}_i \leq b, \quad \text{and} \quad \sum_{i \in S} \sigma_i \leq \phi.
$$

(95)
Algorithm 7 FPTAS for Problem (92) (Désir et al. 2014)

INPUT: \( w_i, v_i, \sigma_i \) for \( i = 1, \ldots, n \), and \( \epsilon \) and \( \phi \)

1: for \( \gamma \in \Gamma_{\epsilon} \) do
2: for \( \delta \in \Delta_{\epsilon} \) do
3: Use (94) to compute \( \hat{w}_i \) and \( \hat{v}_i \).
4: for \( (a, b, c) \in [I] \times [J] \times [n] \) do
5: Compute \( V(a, b, c) \).
6: if \( V(a, b, c) \leq \phi \) then
7: Let \( S_{\gamma, \delta} \) be the corresponding assortment.
8: return assortment \( S \) with maximum \( \frac{\sum_{i \in \delta} w_i v_i}{\sum_{i \in \delta} v_i + 1} \) over all assortments \( \{S_{\gamma, \delta} : \gamma \in \Gamma_{\epsilon}, \delta \in \Delta_{\epsilon}\} \).

where \( I = \lfloor \frac{n}{2} \rfloor - n \) and \( J = \lceil \frac{n}{2} \rceil + n \). To do this, referring to \( \sigma_i \) as the “mass” of an item \( i \), for each \( (a, b, c) \in [I] \times [J] \times [n] \), \( V(a, b, c) \) is used to denote the minimum mass subset of \( \{1, \ldots, c\} \) such that (95) holds. For each \( (a, b, c) \in [I] \times [J] \times [n] \), \( V(a, b, c) \) can be calculated using the following recursion

\[
V(a, b, c) = \begin{cases} 
  w_i v_i & \text{if } 0 \leq a \leq \hat{w}_i \text{ and } b \geq \hat{v}_i \\
  0 & \text{if } a \leq 0 \text{ and } b \geq 0 \\
  \infty & \text{otherwise}
\end{cases}
\]

\[
V(a, b, c + 1) = \min\{V(a, b, c), \sigma_{c+1} + V(a - \hat{w}_{c+1}, b - \hat{v}_{c+1}, c)\}.
\]

The following lemma discusses the performance of Algorithm 7 and the reader is referred to Désir et al. (2014) for its details.

**Lemma 16 (Désir et al. (2014)).** For a given \( \phi \in \Phi_{\epsilon} \), Algorithm 7 returns an \( (1 - \epsilon) \)-approximation solution to problem (92) in \( O(\log(n\hat{w}v/(\hat{w}v))\log(n\hat{w}v)n^3/\epsilon^4) \) number of steps.

Let \( S^* \) be the optimal solution to problem (88), we define \( f^* = \sum_{i \in S^*} w_i v_i / (\sum_{i \in S^*} v_i + 1) \) and \( h^* = \sum_{i \in S^*} \sigma_i \). Using the above lemma, we discuss the performance and run-time of Algorithm 6 in the following.

**Lemma 17.** If there exists a constant \( \alpha > 0 \) such that for all instance of problem (88) we have \( f^* \geq (1 + 2/\alpha) \cdot h^* \) then there is an FPTAS for problem (88).

**Proof.** Suppose for some \( \phi \in \Phi_{\epsilon} \) we have \( \phi/(1 + \epsilon) \leq h^* \leq \phi \). We denote the assortment returned by Algorithm 7 when \( \phi \) is set as its input by \( S(\phi) \) and define \( f(\phi) = \sum_{i \in S(\phi)} w_i v_i / (\sum_{i \in S(\phi)} v_i + 1) \) and \( h(\phi) = \sum_{i \in S(\phi)} \sigma_i \). Furthermore, let \( S^*(\phi) \) be optimal solution to problem (92) and \( f^*(\phi) = \sum_{i \in S^*(\phi)} w_i v_i / (\sum_{i \in S^*(\phi)} v_i + 1) \) and \( h^*(\phi) = \sum_{i \in S^*(\phi)} \sigma_i \). If \( h^* \leq \phi \), then problem (92) is less constrained than the optimal solution; therefore, \( f^*(\phi) \geq f^* \).

By Lemma 16 we have \( f/g \geq (1 - \epsilon)f^*(\phi) \geq (1 - \epsilon)f^* \). Furthermore, we have \( h^* \geq \phi/(1 + \epsilon) \); thus, overall we have

\[
f - h \geq (1 - \epsilon)f^* - (1 + \epsilon)h^*.
\]
Suppose there exists a constant $0 < \alpha < 1/\epsilon - 1$ such that $f^* \geq (1 + 2/\alpha)h^*$. Then, it can be easily checked that $f - h \geq (1 - (\alpha + 1)\epsilon)(f^* - h^*)$. Furthermore, Algorithm 6 makes $O(\log(n\sigma/\epsilon))$ calls to Algorithm 7 and by Lemma 16 the running time of Algorithm 7 is $O(\log(n\sigma/(\epsilon \mu)) \log(n\sigma/\epsilon)n^3/\epsilon^4)$. Therefore it takes Algorithm 6 $O(\log(n\sigma/\epsilon) \log(n\sigma/(\epsilon \mu)) \log(n\sigma/\epsilon)n^3/\epsilon^4)$ runs to find an approximate solution for problem (88), which is polynomial in the input size and $1/\epsilon$. Therefore, if there exist an $0 < \alpha < 1/\epsilon - 1$ such that $f^* \geq (1 + 2/\alpha)h^*$ then Algorithm 6 provides an FPTAS for problem (88). □

Remark 6. The MCDLP-R is similar to the MCDLP-NR except that constraint $\sum_{S \ni i} x_j(S) \leq 1$ exists in the latter but not in the former. The dual variable associated to this constraint is $\sigma_{ij}$, which means that the dual of the MCDLP-R is similar to MCDLP-NR-D with $\sigma_{ij} = 0$ for all $i$ and $j$. As a result, Lemma 14 holds for MCDLP-R as well. Moreover, for the case of MNL choice model, when offering repeated items to customers is allowed, the column generation subproblem becomes

$$\max_{S \in \mathcal{S}} \sum_{i \in S} w_{ij} v_{ij} \sum_{k \in S} v_{kj} + 1.$$  

Liu and Van Ryzin (2008) provided a polynomial time algorithm for the above optimization problem. Consequently, when offering repeated items is permitted the column generation subproblem can solved efficiently.

Appendix D: Supplements to Numerical Experiments in Section 6

Figure 2 shows the performance of Greedy, Conservative, Algorithm 3 (i.e., the 9% algorithm) and Modified Algorithm 3 (i.e., the 15% algorithm) on hotel data set with heterogeneous room fares for different patience levels, sizes of permissible assortments and loading factors. However, unlike Figure 1 where the scale factor was 2, here the scale factor is set as 4. In other words, there is a wider difference between the high and low fares and different room categories.

Similar to Figure 1, we here see that in some cases for low loading factors the Greedy algorithm outperform the other algorithms; whereas, for large loading factors the Conservative algorithm outperform the other methods. However, in the region in-between these two extremes, the Modified Algorithm 3 outperforms the Greedy and Conservative algorithms.
Figure 2: Performance of Greedy, Conservative, Algorithm 3 (i.e., the 9% algorithm) and Modified Algorithm 3 (i.e., the 15% algorithm) on hotel data set with heterogeneous room fares for different patience levels, sizes of permissible assortments for scale factor=4 over different loading factors.