2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Online Stochastic Allocation of Reusable Resources

Xilin Zhang and Wang Chi Cheung

Abstract— **We study a multi-objective model on the allocation of reusable resources under model uncertainty. Heterogeneous customers arrive sequentially according to a latent stochastic process, request for certain amounts of resources, and occupy them for random durations of time. The decision maker's goal is to simultaneously maximize multiple types of rewards generated by the customers, while satisfying the resource capacity constraints in each time step. We develop models and algorithms for deciding on the allocation actions. We show that when the usage duration is relatively small compared with the length of the planning horizon, our policy achieves $1 - O(\epsilon)$ fraction of the optimal expected rewards, where ϵ decays to zero at a near optimal rate as the resource capacities grow.**

I. Introduction

In the online optimization framework, information is revealed sequentially in time. The decisions are made without knowledge of the future information, but they can depend on past observations. In this work, we study online optimization algorithms in *reusable* resource allocation applications, where a resource unit is returned to the system after a period of usage duration, and can be further assigned to another customer. The decision-maker (DM) assigns limited inventories of reusable resources to sequentially arriving customers. In each time step, the DM's decision leads to a set of allocation outcomes, consisting of the amounts of rewards earned, the amounts of resources consumed and the usage durations of the assigned resources. Our model captures a diversity of real-life applications include hotel booking, rental of cars and fashion items and cloud computing services. Our problem instance incorporates the following features:

1) *Multiple objectives.* The DM's goal is to maximize multiple types of rewards.
2) *Customer heterogeneity.* The customers are associated with different customer types.
3) *Online setting.* In each time step, the arriving customer's type is drawn independently and identically from an *unknown* probability distribution.
4) *Reusability.* Each type of resource is endowed with a stochastic usage duration, whose probability distribution is known to the DM. However, the DM does not know the the realized usage duration of an allocation before the resource is returned.

Features 1-3 are shared by both non-reusable and reusable resource allocation problems, while feature 4 is a distinct feature of reusable resource allocation problems. Without considering feature 4, our problem reduces to the non-reusable setting as in [1]. On feature 3, we remark that in many applications, given a customer type, its mean allocation outcome is accessible by machine learning (ML) approaches in a data-efficient manner ([2], [3], [4]). In many existing resource allocation research ([21], [5], [6], [7], [8]), the mean allocation outcomes are assumed to be prior-knowledge acquired through ML models. However, in applications where customer types are represented as high-dimensional feature vectors, the number of types can be exponential in the dimension of the feature vectors or even unbounded. Such a curse of dimensionality hinders the estimation on the proportion of each customer type. Therefore, we treat the probability distribution of each customer type as the unknown object. We further remark on feature 4 that our usage duration is defined in the same manner as [6] and [9], which are recent works on offline reusable resource allocation problems. We highlight that the probability distribution of each usage duration can be arbitrarily defined, which means our result does not depend on specific structures of certain usage distributions, such as the exponential distribution.

Traditional resource allocation problems [10], [11] concern the allocation of non-reusable resources. Online algorithms for allocating non-reusable resources have been extensively studied in [12], [13], [14], [1], [15], [16], [17]. These algorithms involve adaptive weighing processes that balance the trade-off between the rewards earned and the resources consumed. Most of their analysis largely depend on the monotonically-decreasing inventories. However, in our reusable model, the effect of an allocation may be different for each future time step, contingent on whether the allocated resources are returned, causing fluctuating resource consumption amount across consecutive time steps.

To our knowledge, this is the first paper to address reusable resource allocation problems in an online stochastic setting and demonstrate a near-optimal performance guarantee. Some studies focus on assortment planning problems in adversarial settings ([20], [18], [19]), and achieve non-trivial competitive ratios. Offline pricing and assortment planning problems have been studied in [6], [9], [20], where near-optimality is achieved in the form of approximation ratios under full model certainty. The main contribution of our paper can be summarized as follows.

- *Model generality.* We propose a general reusable resource allocation model which allows for various decision settings (such as admission control, matching, pricing and assortment planning), multiple objectives (such as revenue, market share and service level), and large numbers of customer types or allocation types (the algorithm's performance is independent of these sizes).
- *Near-optimal algorithm performance.* We develop an

Both authors are with Department of Industrial and Systems Engineering, Faculty of Engineering, National University of Singapore. (emails: e0408730@u.nus.edu and isecwc@nus.edu.sg)

adaptive weighing algorithm that trades-off among not only the resources occupied and the rewards earned, but also the usage durations. In the regime where each usage duration is short compared with the length of the planning horizon, our algorithm achieves matching near-optimal performance as the online non-reusable resource setting ([1]), as well as the the state-of-art offline reusable setting ([20]).

The remainder of paper is organized as follows. In Section II, we present our model and highlight some of its applications. An online algorithm and the corresponding performance analysis are proposed in Section III. In Section IV, numerical experiments are provided.

II. Model

Notation. The reward types and the resource types are respectively indexed by two finite sets $\mathcal{I}_r$ and $\mathcal{I}_c$. A generic reward type or resource type is denoted as i. For each $i \in \mathcal{I}_c$, the DM has $c_i \in \mathbb{R}_{>0}$ units of resource i for allocation. Each customer is associated with a customer type $j \in \mathcal{J}$, which reflects the customer's features. We denote the set of possible allocation actions as $\mathcal{K}$, and each element $k \in \mathcal{K}$ as an action. The action set can model a broad range of decisions, which is elaborate in the end of Section II.

The DM allocates the resources in T discrete time steps. In time step $t \in \{1, \ldots, T\}$, at most one customer arrives. We denote the customer type of the arrival at time t as $j(t)$. In particular, we designate the type $j_{\mathsf{null}} \in \mathcal{J}$ to represent the case of no arrival. We assume that $j(1), \ldots, j(T)$ are independently and identically distributed (i.i.d.) random variables over $\mathcal{J}$. We denote $p_j = \Pr(j(1) = j)$, and $\mathbf{p} = \{p_j\}_{j\in\mathcal{J}}$.

When a customer (denote his type as j) arrives, the DM chooses an action $k \in \mathcal{K}$. The choice leads to an array of stochastic outcomes, consisting of the amount of rewards earned $W_{jk} = (W_{ijk})_{i\in\mathcal{I}_r}$, the amount of resources occupied $A_{jk} = (A_{ijk})_{i\in\mathcal{I}_c}$, and the usage durations $\{D_i\}_{i\in\mathcal{I}_c}$. For the no arrival customer type j_{null}, we stipulate that $\Pr(W_{i',j_{\mathsf{null}},k} = A_{i,j_{\mathsf{null}},k} = 0) = 1$ for all $i' \in \mathcal{I}_r, i \in \mathcal{I}_c, k \in \mathcal{K}$, since there should be no reward earned and no resource occupied in the case of no arrival. To ensure feasibility in our resource constrained model, we assume that there exists a null action $k_{\mathsf{null}} \in \mathcal{K}$ that satisfies $\Pr(W_{i',j,k_{\mathsf{null}}} = A_{i,j,k_{\mathsf{null}}} = 0) = 1$ for all $i' \in \mathcal{I}_r, i \in \mathcal{I}_c, j \in \mathcal{J}$. Selecting the null action is equivalent to rejecting a customer.

For each j, k, the stochastic outcomes follow the joint distribution $\mathcal{O}_{jk}$, namely $(W_{jk}, A_{jk}) \sim \mathcal{O}_{jk}$. We allow W_{jk}, A_{jk} to be arbitrarily correlated. For each $i \in \mathcal{I}_c$, the random usage duration D_i is independent of W_{jk}, A_{jk}. This assumption is also made in related works on offline reusable resource allocation, such as [6] and [9], since the usage duration reflects more of a customer's intrinsic needs on each resource. We assume that $W_{ijk} \in [0, w_{\max}]$ for each $i \in \mathcal{I}_r, j \in \mathcal{J}, k \in \mathcal{K}$, $A_{ijk} \in [0, a_{\max}]$ almost surely for each $i \in \mathcal{I}_c, j \in \mathcal{J}, k \in \mathcal{K}$, and $D_i \in \{0, 1, \ldots, d_{\max}\}$ almost surely for each $i \in \mathcal{I}_c$. Additionally, we denote $w_{ijk} = \mathbb{E}[W_{ijk}]$, $a_{ijk} = \mathbb{E}[A_{ijk}]$, and $d_i = \mathbb{E}[D_i]$.

Model uncertainty and dynamics. We assume that the DM knows the horizon length T, the values of $w_{\max}$, $a_{\max}$, $d_{\max}$, as well as $\Pr(D_i \geq t)$ for every $i \in \mathcal{I}_c, t \in \{1, \ldots, T\}$. However, the DM does not know the probability distribution $\mathbf{p}$ over customer types. At each time step $t \in \{1, \ldots, T\}$, the DM observes the type $j(t) \sim \mathbf{p}$ of the arriving customer, and the mean outcomes $\{(w_{j(t),k}, a_{j(t),k})\}_{k\in\mathcal{K}}$ specific to the type $j(t)$. Then, the DM chooses an action $k(t) \in \mathcal{K}$, and observes the stochastic outcomes of rewards $\{W_{i,j(t),k(t)}(t)\}_{i\in\mathcal{I}_r}$ and resources $\{A_{i,j(t),k(t)}(t)\}_{i\in\mathcal{I}_c}$ at time t. Our model uncertainty scenario included the case when the DM knows the mean outcomes a_{ijk}, w_{ijk} in advance. For example, the DM could have estimates on $a_{ijk}, w_{ijk}, \Pr(D_i \geq t)$ by constructing supervised learning models [2], [3], [4] on a pool of customer demand data.

An integer programming formulation. We let binary decision variables $X_k^\pi(t)$ be the DM's decision under a non-anticipatory algorithm π, where $X_k^\pi(t) = 1$ if action k is taken at time t, and $X_k^\pi(t) = 0$ otherwise. Under a non-anticiaptory algorithm, $\{X_k^\pi(t)\}_{k\in\mathcal{K}}$ depends on historical observations $\{j(s)\}_{s=1}^t \cup \{W_{i,j(s),k(s)}(s)\}_{i\in\mathcal{I}_r, 1\leq s\leq t-1} \cup \{A_{i,j(s),k(s)}(s)\}_{i\in\mathcal{I}_c, 1\leq s\leq t-1}$. The DM aims to maximize $\mathbb{E}[\min_{i\in\mathcal{I}_r} \sum_{t=1}^T \sum_{k\in\mathcal{K}} W_{i,j(t),k}(t) X_k^\pi(t)]$, which achieves the simultaneous maximization of all the reward types by ensuring max-min fairness. Here we maximize the rewards to keep in line with the resource allocation literature instead of minimize the regret as in the classical online convex optimization literature, but we remark that they are essentially eqivalent. For each $i \in \mathcal{I}_c$ and $t \in \{1, \ldots, T\}$, we require that the resource constraint

$$\sum_{\tau=1}^{t} \sum_{k\in\mathcal{K}} \mathbf{1}(D_i(\tau) \geq t - \tau + 1) A_{i,j(\tau),k}(\tau) X_k^\pi(\tau) \leq c_i \quad (1)$$

holds with certainty. The left hand side in (1) represents the amount of type i resources occupied at time step t. Our goal can be formulated as the following binary integer program

$$\begin{aligned}
\text{(IP-C)} \quad & \max_{\text{non-anticipatory } \pi} \quad \mathbb{E}[\hat{\lambda}] \\
\text{s.t.} \quad & \sum_{t=1}^{T} \sum_{k\in\mathcal{K}} W_{i,j(t),k}(t) X_k^\pi(t) \geq T\hat{\lambda} \quad \forall i \in \mathcal{I}_r \\
& \sum_{\tau=1}^{t} \sum_{k\in\mathcal{K}} \mathbf{1}(D_i(\tau) \geq t - \tau + 1) A_{i,j(\tau),k}(\tau) X_k^\pi(\tau) \leq c_i \\
& \qquad\qquad\qquad \forall i \in \mathcal{I}_c,\ t \in \{1, \ldots T\} \\
& \sum_{k\in\mathcal{K}} X_k^\pi(t) = 1 \quad \forall\, t \in [T] \\
& X_k^\pi(t) \in \{0, 1\} \quad \forall k \in \mathcal{K},\ t \in [T].
\end{aligned}$$

We remark that the term $\mathbf{1}(D_i(\tau) \geq t - \tau + 1)$ induces non-stationarity in resource consumption, since even when a DM selects the same action k at $\tau_1 < \tau_2$, their amounts of resource consumption $\mathbf{1}(D_i(\tau_1) \geq t - \tau_1 + 1) A_{i,j(\tau_1),k}(\tau_1), \mathbf{1}(D_i(\tau_2) \geq t - \tau_2 + 1) A_{i,j(\tau_2),k}(\tau_2)$ at time t are differently distributed. Existing works on non-reusable resources crucially hinges on model stationarity, which does not hold true in our setting.

A tractable benchmark. The goal of constructing a non-anticipatory algorithm that achieves the optimal value of (IP-C) is analytically intractable due to the curse of dimensionality. The intractability motivates us to consider an alternative linear program (LP), dubbed (LP-E), where the realization of the customer arrivals, their usage duration and outcomes exactly follow the expectation:

$$\begin{aligned}
\text{(LP-E)}\ & \max\ \lambda \\
\text{s.t.} & \sum_{t=1}^{T}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} p_j w_{ijk} y_{jk}(t) \geq T\lambda \quad \forall i \in \mathcal{I}_r \\
& \sum_{\tau=1}^{t}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} p_j \Pr(D_i \geq t-\tau+1) a_{ijk} y_{jk}(\tau) \leq c_i \\
& \qquad\qquad \forall i \in \mathcal{I}_c,\ t \in \{1,\dots T\} \\
& \sum_{k\in\mathcal{K}} y_{jk}(t) \leq 1 \quad \forall j \in \mathcal{J},\ t \in \{1,\dots,T\} \\
& y_{jk}(t) \geq 0 \quad \forall j \in \mathcal{J},\ k \in \mathcal{K},\ t \in [T].
\end{aligned}$$

Define the optimal objective value of (LP-E) to be λ^*, and let the optimal objective of (IP-C) be $\hat{\lambda}^*$. The following lemma shows that λ^* is a tractable upper bound for the expected reward of any online algorithms.

Lemma 1. $\lambda^* \geq \mathbb{E}[\hat{\lambda}^*]$.

For the algorithm design, we further introduce a "steady state" benchmark, assuming the decision variables are invariant across time:

$$\begin{aligned}
\text{(LP-SS):}\ & \max_{x_{jk}}\ \tilde{\lambda} \\
\text{s.t.} & \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} p_j w_{ijk} x_{jk} \geq \tilde{\lambda} && \forall i \in \mathcal{I}_r \\
& \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} p_j a_{ijk} d_i x_{jk} \leq c_i && \forall i \in \mathcal{I}_c \\
& \sum_{k\in\mathcal{K}} x_{jk} \leq 1 && \forall j \in \mathcal{J} \\
& x_{jk} \geq 0 && \forall j \in \mathcal{J},\ k \in \mathcal{K}.
\end{aligned}$$

We denote an optimal solution of (LP-SS) as x^*_{jk}, and the optimal value of (LP-SS) as $\tilde{\lambda}^*$. We further define

$$\gamma = \min\left\{\min_{i\in\mathcal{I}_c}\left\{\frac{c_i}{a_{\max}}\right\}, \frac{T\tilde{\lambda}^*}{w_{\max}}\right\}.$$

Assumption 1. There exists $\delta \in (0,1)$ and $\bar{d}(\delta) \leq T$ such that $\sum_{t=\bar{d}(\delta)+1}^{\infty} \Pr(D_i \geq t) \leq \delta$, $\forall i, j, k$.

This assumption indicates that our algorithm does not apply to large D_i, say for non-reusable resources where $D_i = T$ with certainty. In the next lemma, we show that $\tilde{\lambda}^*$ is close to λ^*.

Lemma 2. $\left(1 - \frac{\delta}{\gamma}\right)\left(T\lambda^* - \bar{d}(\delta) w_{\max}\right) \leq T\tilde{\lambda}^* \leq T\lambda^*$.

We remark that under a wide range of usage durations, we can use (LP-SS) as a benchmark to gauge the performance of our algorithm. For instance, for light tailed D_i (for example, there exists $u > 0$ such that $\lim_{t\to\infty} \Pr(D_i \geq t)t^u = 0$), we can take $\delta = \epsilon/T$, $\bar{d}(\delta) = d_{max} \log(d_{\max}T/\epsilon)$. If D_i has bounded support, i.e. $D_i \in [0, d_{\text{UB}}]$ almost surely, we can take $\delta = 0$ and let $\bar{d}(\delta) = d_{\text{UB}}$.

Applications. Before proceeding to our algorithm development, we highlight the generality of our model by discussing some of its applications, where the reward type set $\mathcal{I}_r$, the customer type set $\mathcal{J}$ and the action set $\mathcal{K}$ can be chosen to model a variety of decisions.

- **Admission control.** In this setting, the DM is to either admit or reject each arriving customer [21]. Real life examples include patient inflow control in an emergency department or an ICU. The admission control setting can be modeled by letting action set $\mathcal{K} = \{\text{accept}, \text{reject}\}$. The reward of a resource is fixed at r_i for $i \in \mathcal{I}_c$. Upon taking an action k for a type j customer, an array of stochastic demands $\{A_{ijk}\}_{i\in\mathcal{I}_c}$ is generated. Our model captures different reward settings. We list some of the examples: for simultaneously maximizing the revenue/social welfare for each type of resource, define $\mathcal{I}_r = \mathcal{I}_c$ and $W_{ijk} = r_i A_{ijk}$. For maximizing the total revenue/social welfare of all resources, let $\mathcal{I}_r = \{1\}$ be a singleton, and define $W_{1jk} = \sum_{i\in\mathcal{I}_c} r_i A_{ijk}$. For maximizing the service level of each resource, we define $\mathcal{I}_r = \mathcal{I}_c$ and $W_{ijk} = A_{ijk}$. We remark that multiple kinds of rewards can be modeled simultaneously.
- **Assortment Planning.** In assortment planning problems, one unit of resource i is associated with a fixed price r_i. The DM influences the customers' demands through offering different assortments of resources. Real life assortment planning examples with reusable resources include renting of fashion items and vehicles. Contingent upon the arrival of a customer, say of type j, the DM decides the assortment $k \in \mathcal{K}$ to display, where $\mathcal{K}$ is a collection of subsets of $\mathcal{I}_c$ ([9], [22]). Let q_{ijk} denote the probability for customer type j to choose product i in assortment k. In the revenue management literature, the probability q_{ijk} is modeled by a random utility choice model. The assortment planning problem (simultaneously maximizing revenue of each resource) can be incorporated in our model by setting $\mathcal{I}_r = \mathcal{I}_c$, setting A_{ijk} to be the Bernoulli random variable with mean q_{ijk}, and setting $W_{ijk} = r_i A_{ijk}$.

III. Online algorithm and performance analysis

Main results. In this section, we propose a multi-stage adaptive weighing algorithm (dubbed Algorithm A as in "Adaptive", and displayed in Algorithm 1). We first provide our main results. We assume there exists ϵ satisfying $\gamma = \Omega(\log(|\mathcal{I}|T/\epsilon)/\epsilon^2)$, where $\mathcal{I} = \mathcal{I}_c \cup \mathcal{I}_r$. Let

$$W_i(t) = \sum_{k\in\mathcal{K}} W_{i,j(t),k}(t) X_k^A(t)$$

denote the type $i \in \mathcal{I}_r$ reward achieved by Algorithm A, our main result is shown in the following theorem.

Theorem 1. *Let $\epsilon > 0$ be an arbitrary constant satisfying $\gamma = \Omega(\log(|\mathcal{I}|T/\epsilon)/\epsilon^2)$. Without knowing $\mathbf{p}$,*

$$\sum_{t=1}^{T} W_i(t) \geq \left(1 - \frac{\delta}{\gamma}\right) T\lambda^*(1 - O(\epsilon)) - \bar{d}(\delta)\tilde{O}(\epsilon)$$

for every $i \in \mathcal{I}_r$ with probability at least $1 - \epsilon(1+\epsilon)^\delta$.

We remark that if D_i has bounded support, i.e. $D_i \in [0, d_{\text{UB}}]$ almost surely for all $i \in \mathcal{I}_c$, then the above reward can be simplified as $\sum_{t=1}^{T} W_i(t) \geq T\lambda^*(1 - O(\epsilon)) - d_{\text{UB}}\tilde{O}(\epsilon)$ for every $i \in \mathcal{I}_r$ with probability at least $1 - \epsilon$. In the case when $\delta \leq \log(|\mathcal{I}|T)/\epsilon$ and $\bar{d}(\delta) = o(T)$, our algorithm achieves a reward at least $T\lambda^*(1 - O(\epsilon))$. This result nearly matches the [1] in studying an online non-reusable resource allocation problem, as well as [20] in the state-of-art work on the offline assortment planning with reusable resources.

The assumption $\gamma = \Omega(\log(|\mathcal{I}|T/\epsilon)/\epsilon^2)$ means that the amount of resource c_i for each i is sufficiently large, and the planning horizon T is sufficiently long. Crucially, the performance guarantee in Theorem 1 does not deteriorate even when $|\mathcal{J}|$ or $|\mathcal{K}|$ grows. Consequently, our Algorithm A is applicable to complex scenarios when $|\mathcal{J}| > T$.

High-level description and comparison against [1]. Our Algorithm A extends [1]'s idea of adaptive weighing from the non-reusable setting to the reusable setting. [1] proposes a multi-stage adaptive weighing algorithm to trade-off between the rewards and the resources. For each reward and resource constraint of a fluid LP (corresponding to our (LP-E)), a penalty weight is defined. The weight on each constraint progressively gets larger as the reward generated or the resource consumed gets closer to their total capacity (the capacity of each reward is approximated). [1] does not apply to the reusable setting, as their penalty weights and algorithm design depend on the monotonic-decreasing resources.

Somewhat surprisingly, we can still utilize their adaptive weighing idea. we approximate (LP-E) with a knapsack-constrained (LP-SS), and define penalty weights that incorporates the usage duration as well as the resource consumption. In a series of Lemmas that eventually leads to Theorem 1, we show that our algorithm effectively capitalizes the reusability of the resources to maximize the total rewards. We overcome the technical difficulty in non-monotonic and time-correlated resource levels, by achieving near-optimal performance. Nevertheless, we remark that the closeness of (LP-E) and (LP-SS) builds upon Assumption 1, and hence our algorithm is not applicable to the non-reusable setting.

A multistage online algorithm. In Algorithm 1, we divide the time horizon into l stages $\{-1, 0, 1, \ldots, l-1\}$ where l satisfies $\epsilon 2^l = 1$ for some $\epsilon \in [\bar{d}(\delta)/T, 1/2]$. Stage -1 consists of $t^{(-1)} = \epsilon T$ time periods. This stage is solely for exploration on the latent $\{p_j\}_{j\in\mathcal{J}}$, and the first ϵT customers are served with random actions. Stage $r \in \{0, \ldots, l-1\}$ consists of $t^{(r)} = \epsilon T 2^r$ time periods. The assumption $\epsilon \in [\bar{d}(\delta)/T, 1/2]$ ensures that $l \geq 1$ (there is at least 1 stage) and $\epsilon T \geq \bar{d}(\delta)$ (each stage consists of at least $\bar{d}(\delta)$ time periods). We denote $j^{(r)}(s)$ as the type of the customer who arrives at the s-th time step in stage r (where $s \in \{1, \ldots, t^{(r)}\}$), meaning that $j^{(r)}(s) = j(t^{(r)} + s)$.

In each stage $r \geq 0$, Algorithm A consists of two steps. In **Step 1**, we estimate the optimum of (LP-SS). In **Step 2**, we define "penalty weights" on constraints of (LP-SS), and choose the action that balances between each constraint.

Step 1: *Estimate the value of $\tilde{\lambda}^*$ (Line 3 of Algorithm 1).* We first derive $\mu^{(r)*}$, which is the optimal objective value of the linear program (LP-RSS)$^{(r)}$:

$$\begin{aligned}
&\max_{x_{jk}^{(r)}} \ \mu^{(r)} \\
&\text{s.t.} \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{p}_j^{(r)} w_{ijk} x_{jk}^{(r)} \geq \mu_r && \forall i \in \mathcal{I}_r \\
&\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{p}_j^{(r)} a_{ijk} d_i x_{jk}^{(r)} \leq c_i && \forall i \in \mathcal{I}_c \\
&\sum_{k\in\mathcal{K}} x_{jk}^{(r)} \leq 1 && \forall j \in \mathcal{J} \\
&x_{jk}^{(r)} \geq 0 && \forall j \in \mathcal{J},\ k \in \mathcal{K},
\end{aligned}$$

where $\hat{p}_j^{(r)} = \frac{1}{t^{(r-1)}}\sum_{t=1}^{t^{(r-1)}} \mathbf{1}(j^{(r-1)}(t) = j)$, denoting the empirical customer distribution based on customer arrivals in stage $r-1$. (LP-RSS)$^{(r)}$ is a sample average approximation (SAA) of (LP-SS). It is worth mentioning that both (LP-SS) and (LP-RSS)$^{(r)}$ are highly tractable, even in assortment planning application when $|\mathcal{K}|$ is exponential in $|\mathcal{I}_c|$ ([23]). In addition, the knapsack-type constraints in (LP-RSS)$^{(r)}$ allows us to apply the adaptive weighing in Step 2 in a similar manner to the non-reusable setting. Given that $\mu^{(r)*}$ is easily obtained in Step 1, we define $\lambda^{(r)}$ in the following lemma, and show it is a progressively more accurate estimate of $\tilde{\lambda}^*$ as r grows.

Lemma 3. *Define $\epsilon_x^{(r)} = \sqrt{\frac{4T\log\frac{2|\mathcal{I}|}{\eta}}{t^{(r)}\gamma}}$ for $r \in \{-1, 0, 1, \ldots, l-1\}$. For any $\eta \in (0,1)$, with probability at least $1 - 2\eta$,*

$$\tilde{\lambda}^*(1 - 3\epsilon_x^{(r-1)}) \leq \lambda^{(r)} \leq \tilde{\lambda}^*$$

where $\lambda^{(r)} = \frac{\mu^{(r)*}}{1+\epsilon_x^{(r-1)}}$.

Step 2: *Run an online algorithm given $\lambda^{(r)}$ (Line 4 - Line 10 of Algorithm 1).* With slight abuse of notation, we write $A_{i,j(t^{(r)}+s),k}(t^{(r)}+s)$ as $A_{i,j^{(r)}(s),k}^{(r)}(s)$, $W_{i,j(t^{(r)}+s),k}(t^{(r)}+s)$ as $W_{i,j^{(r)}(s),k}^{(r)}(s)$, and $D_{i,j(t^{(r)}+s),k}(t^{(r)}+s)$ as $D_i^{(r)}(s)$. In addition, we denote $X_{j(t^{(r)}+s),k}^{(r)}(t^{(r)}+s)$ as $X_{j^{(r)}(s),k}^{(r)}(s)$. Define, at time t in stage r, $Y_{i\tau t}^{(r)} = \sum_{k\in\mathcal{K}} \mathbf{1}(D_i^{(r)}(\tau) \geq t - \tau + 1) A_{i,j^{(r)}(\tau),k}^{(r)}(\tau) X_{j^{(r)}(\tau),k}^{(r)}(\tau)$ and $Z_{it}^{(r)} = \sum_{k\in\mathcal{K}} W_{i,j^{(r)}(t),k}^{(r)}(t) X_{j^{(r)}(t),k}^{(r)}(t)$ respectively as resource i consumed by customer τ, and reward i earned.

At the s-th time step of stage r, after observing the customer type $j^{(r)}(s)$ we take action $k^{(r)}(s)$ according to Line 7. The parameter $\phi_{i,s,t}^{(r)}$ represents a "penalty weight" for the resource constraint $i \in \mathcal{I}_c$ in (LP-SS). If the allocation decisions during $1, \ldots, s-1$ in stage r leads to a high amount of resource i occupation at time t, the penalty $\phi^{(r)}_{i,s,t}$ would also be high. Similarly, a lower amount of accrued reward type $i \in \mathcal{I}_r$ during $1, \dots, s-1$ in stage r leads to a higher value of the weight $\psi^{(r)}_{i,s}$. Both weights quantify the DM's emphasis on resources and rewards.

Performance guarantee of Algorithm A. Our analysis involves bounding the total probability of violating each constraint in (LP-C) in all time steps. Each violation leads to unserved customers and lost rewards. We show that under Algorithm A, all constraints of (LP-C) are satisfied with high probability. To understand the choice of $k^{(r)}(s+1)$ in Line 7 of Algorithm 1, we introduce an auxiliary offline static algorithm (dubbed Algorithm S as in "Static"). Algorithm S requires knowing $x^* = \{x^*_{jk}\}_{jk}$, an optimal solution to (LP-SS), and a tuning parameter $\epsilon \in (0,1)$ for preserving capacities in anticipation of any constraint violation. At a time t, if a customer of type j arrives, the DM selects action $k \in \mathcal{K} \backslash k_{\text{null}}$ with probability $\frac{x^*_{jk}}{1+\epsilon}$, and selects the null action k_{null} with probability $\frac{\epsilon}{1+\epsilon} + \frac{x^*_{j,k_{\text{null}}}}{1+\epsilon}$. We denote $X^S_{j(t^{(r)}+s),k}(t^{(r)}+s)$ as $\tilde{X}^{(r)}_{j^{(r)}(s),k}(s)$. Define $\tilde{Y}^{(r)}_{i\tau t} = \sum_{k\in\mathcal{K}} \mathbf{1}(D^{(r)}_i(\tau) \geq t - \tau + 1) A^{(r)}_{i,j^{(r)}(\tau),k}(\tau) \tilde{X}^{(r)}_{j^{(r)}(\tau)k,r}(\tau)$ and $\tilde{Z}^{(r)}_{it} = \sum_{k\in\mathcal{K}} W^{(r)}_{i,j^{(r)}(t),k}(t) \tilde{X}^{(r)}_{j^{(r)}(t),k}(t)$ where $\Pr(\tilde{X}^{(r)}_{j^{(r)}(\tau),k}(\tau) = 1) = \frac{x^*_{jk}}{1+\epsilon}$ for each τ in stage r. A performance guarantee of Algorithm S is provided in the following lemma.

Lemma 4. *Let $\eta = \epsilon/(5l)$, Algorithm S achieves a total reward of at least*

$$\sum_{r=0}^{l-1}\sum_{t=1}^{t^{(r)}}\sum_{k\in\mathcal{K}} W^{(r)}_{i,j^{(r)}(t),k}(t)\tilde{X}^{(r)}_{j^{(r)}(t),k}(t) \geq T\tilde{\lambda}^*(1-O(\epsilon)) - \bar{d}(\delta) w_{\max} O\left(\epsilon + \log\frac{1}{\epsilon}\right)$$

for every $i \in \mathcal{I}_r$ with probability at least $1 - \epsilon(1+\epsilon)^{\delta}$.

For analysis sake, we consider a hybrid Algorithm $A_s S_{t^{(r)}-s}$ in each stage r. For Algorithm $A_s S_{t^{(r)}-s}$, the DM makes allocation decisions based on Algorithm A in time step $\{1, \dots, s\}$, and based on Algorithm S in time step $\{s+1, \dots, t^{(r)}\}$. We show that $A_{s+1} S_{t^{(r)}-s-1}$ outperforms $A_s S_{t^{(r)}-s}$, which inductively leads to the conclusion that the performance of the online adaptive Algorithm A is no worse than the offline static Algorithm S (see Lemma 4). This induction technique is introduced in [1] on the non-reusable setting. We need more refined techniques to disentangle the time-correlation between the resources constraints at each time step, and finally prove the following lemma.

Lemma 5. *Algorithm A achieves a total reward of at least*

$$\sum_{r=0}^{l-1}\sum_{t=1}^{t^{(r)}}\sum_{k\in\mathcal{K}} W_{ij(t)k,r}(t)\tilde{X}^{(r)}_{j(t)k,r}(t) \geq T\tilde{\lambda}^*(1-O(\epsilon)) - \bar{d}(\delta) w_{\max} O\left(\epsilon + \log\frac{1}{\epsilon}\right)$$

for every $i \in \mathcal{I}_r$ with probability at least $1 - \epsilon(1+\epsilon)^{\delta}$.

Algorithm 1 Online Algorithm A

Input: the number of time periods T, the capacities for each resource c_i, the values of γ and $\epsilon \in [d_i/T, 1/2]$.
Output: actions to take $k^{(r)}(t)$, for $r = 0, \dots, l-1, t = 1, \dots, t^{(r)}$.

```
1:  Set $l = \log(1/\epsilon)$. Initialize $t^{(-1)} = \epsilon T$.
2:  for $r = 0, \dots, l-1$ do
3:      Compute $\lambda^{(r)}$ by solving (LP-RSS)$^{(r)}$.
4:      Set
        $$\epsilon_x^{(r-1)} = \sqrt{\frac{4T\log\frac{2|\mathcal{I}|}{\eta}}{t^{(r-1)}\gamma}},\ \epsilon_z^{(r)} = \sqrt{\frac{2w_{\max}(1+\epsilon)\log\frac{2|\mathcal{I}|l}{\eta}}{t^{(r)}\lambda^{(r)}}}.$$
5:      Set
        $$\phi^{(r)}_{i,1,t} = \begin{cases} \frac{\epsilon\gamma}{c_i(1+\epsilon)^{\gamma-\delta}},\ t=1 \\ \frac{\epsilon\gamma}{c_i(1+\epsilon)^{\gamma-\delta}} \prod_{\tau=2}^{t}\left(1+\epsilon\frac{\gamma\Pr(D_i \geq t-\tau+1)}{d_i(1+\epsilon)}\right), \\ \qquad t = 2, \dots, t^{(r)} \end{cases}$$
        for each $i \in \mathcal{I}_c$, and
        $$\psi^{(r)}_{i,1} = -\frac{\epsilon_z^{(r)}\prod_{\tau=2}^{t^{(r)}}\left(1-\epsilon_z^{(r)}\frac{\lambda^{(r)}}{w_{\max}(1+\epsilon)}\right)}{w_{\max}\left(1-\epsilon_z^{(r)}\right)^{\frac{\left(1-\epsilon_z^{(r)}\right)t^{(r)}\lambda^{(r)}}{w_{\max}}}}.$$
        for each $i \in \mathcal{I}_r$.
6:      for $s = 1, \dots, t^{(r)}$ do
7:          Observe customer type $j^{(r)}(s)$, take action:
            $$k^{(r)}(s) \in \arg\min_{k\in\mathcal{K}} \left\{\sum_{t=s}^{t^{(r)}}\sum_{i\in\mathcal{I}_c} a_{i,j^{(r)}(s),k}\Pr(D_i \geq t-s+1)\phi^{(r)}_{i,s,t} + \sum_{i\in\mathcal{I}_r} w_{i,j^{(r)}(s),k}\psi^{(r)}_{i,s}\right\}.$$
8:          Set $Y^{(r)}_{ist} = a_{i,j^{(r)}(s),k^{(r)}(s)}\Pr(D_i \geq t-s+1)$ for each $i \in \mathcal{I}_s$ and $t \in \{s, \dots, t^{(r)}\}$, and $Z^{(r)}_{is} = w_{i,j^{(r)}(s),k^{(r)}(s)}$ for each $i \in \mathcal{I}_r$.
9:          Set for each $i \in \mathcal{I}_c$
            $$\phi^{(r)}_{i,s+1,t} = \frac{\phi^{(r)}_{i,s,t}(1+\epsilon)^{\frac{\gamma}{c_i}Y^{(r)}_{ist}}}{\left(1+\epsilon\frac{\gamma\Pr(D_i\geq t-s)}{d_i(1+\epsilon)}\right)},\ t = s+1, \dots, t^{(r)},$$
            and for each $i \in \mathcal{I}_r$,
            $$\psi^{(r)}_{i,s+1} = \frac{\psi^{(r)}_{i,s}\left(1-\epsilon_z^{(r)}\right)^{\frac{1}{w_{\max}}Z^{(r)}_{is}}}{\left(1-\epsilon_z^{(r)}\frac{\lambda^{(r)}}{w_{\max}(1+\epsilon)}\right)}.$$
10:     end for
11: end for
```

Putting Lemmas 2 and 5 together, we have Theorem 1. The proofs and some supplement experiments can be found at https://arxiv.org/abs/2308.00281.

IV. Numerical experiments

We consider an assortment planning problem. One unit of resource i is associated with a fixed price r_i. Contingent upon the arrival of a customer, say of type j, the DM decides the assortment $k \in \mathcal{K}$ to display, where $\mathcal{K}$ is a collection of subsets of $\mathcal{I}_c$. Let q_{ijk} denote the probability for customer type j to choose product i in assortment k. Therefore, the assortment planning problem (simultaneously maximizing the revenue of each resource) can be incorporated in our model by setting $\mathcal{I}_r = \mathcal{I}_c$, setting A_{ijk} to be the Bernoulli random variable with mean q_{ijk}, and setting $W_{ijk} = r_i A_{ijk}$. In our test, the probability q_{ijk} is modeled by the multinomial logit (MNL) choice model. Each resource $i \in \mathcal{I}_c$ is associated with a feature vector $\boldsymbol{f}_i \in \mathbb{R}^m$, and each customer type $j \in \mathcal{J}$ is associated with a set of feature vectors $\{\boldsymbol{b}_{ij}\}_{i \in \mathcal{I}_c}$, where $\boldsymbol{b}_{ij} \in \mathbb{R}^m$ for each $i \in \mathcal{I}_c$. The feature vector $\boldsymbol{f}_i$ could involve the fixed price r_i, and $q_{ijk} = \frac{\exp(\boldsymbol{b}_{ij}^\top \boldsymbol{f}_i)}{1+\sum_{\ell \in k} \exp(\boldsymbol{b}_{\ell j}^\top \boldsymbol{f}_\ell)}$ if $i \in k$, and $q_{ijk} = 0$ if $i \notin k$. In complement, the probability of no purchase is $\frac{1}{1+\sum_{\ell \in k} \exp(\boldsymbol{b}_{\ell j}^\top \boldsymbol{f}_\ell)}$. Notice that the size of the action set $\mathcal{K}$ scales exponentially with the number of products. Nevertheless, for the MNL models, $k^{(r)}(t)$ can be computed efficiently by solving a simple LP whose computational time is polynomial in $|\mathcal{I}_c|$ [24].

We consider a synthetic data-set with 14 types of resources indexed by $\mathcal{I}_c = \{1, 2, \dots, 14\}$, and 1000 types of customers indexed by $\mathcal{J} = \{1, 2, \dots, 1000\}$. We allow offering any assortment of fewer than 5 products, and hence the assortment set is of size $\sum_{i=1}^{5} C_i^{14} = 3472$. We let $\boldsymbol{p}$ follow a discrete distribution with a support of $|\mathcal{J}|$. For any $n \geq 1$, we set $T = 1000n$, $c_i = 20n$ and D_i follow a randomly generated probability distribution with a bounded support of $[1, 200n]$. Theoretically, the regret of both Algorithms A and S should grow sublinearly against the scale n, roughly at the rate of $\tilde{O}(\sqrt{n})$. For each T value, we run 10 simulations using a column generation approach and take the average as well as the standard deviation.

T	ϵ	UB	Total Revenue				% Gap from UB	
			Algo S		Algo A		Algo S	Algo A
			Mean	Std	Mean	Std		
1000	0.3	644.90	507.06	42.1689	331.03	2.531798	21.37%	48.67%
2000	0.22	1289.80	1097.47	50.88593	887.28	7.681146	14.91%	31.21%
3000	0.185	1934.70	1702.97	75.61902	1419.54	8.537564	11.98%	26.63%
4000	0.162	2579.60	2327.51	86.79659	1966.83	6.663332	9.77%	23.75%
5000	0.148	3224.50	2936.00	98.77243	2522.00	10.44462	8.95%	21.79%
6000	0.136	3869.40	3557.05	74.11313	3086.28	20.91315	8.07%	20.24%
7000	0.127	4514.30	4200.24	67.91298	3647.95	9.508417	6.96%	19.19%
8000	0.12	5159.20	4804.85	68.66264	4217.17	14.06983	6.87%	18.26%

TABLE I

Results with 14 resource types and 1000 customer types.

In Table I, the third column of upper bounds are the optimal values of (LP-SS). The offline Algorithm S performs better than the online Algorithm A. Algorithm A achieves rewards within $1 - 2\epsilon$ fraction of the upper bounds.

References

[1] N. R. Devanur, K. Jain, B. Sivan, and C. A. Wilkens, "Near optimal online algorithms and fast approximation algorithms for resource allocation problems," *Journal of the ACM (JACM)*, vol. 66, no. 1, pp. 1–41, 2019.

[2] X. Chen, Z. Owen, C. Pixton, and D. Simchi-Levi, "A statistical learning approach to personalization in revenue management," *Management Science*, vol. 68, no. 3, pp. 1923–1937, 2022.

[3] Y. Han, F. C. Pereira, M. Ben-Akiva, and C. Zegras, "A neural-embedded choice model: Tastenet-mnl modeling taste heterogeneity with flexibility and interpretability," *arXiv preprint arXiv:2002.00922, 2020.*

[4] A. Aouad and A. Désir, "Representing random utility choice models with neural networks," *arXiv preprint arXiv:2207.12877, 2022.*

[5] Y. Chen, R. Levi, and C. Shi, "Revenue management of reusable resources with advanced reservations," *Production and Operations Management*, vol. 26, no. 5, pp. 836–859, 2017.

[6] Y. Lei and S. Jasin, "Real-time dynamic pricing for revenue management with reusable resources, advance reservation, and deterministic service time requirements," *Operations Research*, vol. 68, no. 3, pp. 676– 685, 2020.

[7] J. Baek and W. Ma, "Bifurcating constraints to improve approximation ratios for network revenue management with reusable resources," *Operations Research*, 2022.

[8] O. Besbes, A. N. Elmachtoub, and Y. Sun, "Static pricing: Universal guarantees for reusable resources," *Operations Research*, 2021.

[9] P. Rusmevichientong, M. Sumida, and H. Topaloglu, "Dynamic assortment optimization for reusable prod- ucts with random usage durations," *Management Science*, vol. 66, no. 7, pp. 2820–2844, 2020.

[10] D. Adelman, "Dynamic bid prices in revenue management," *Operations Research*, vol. 55, no. 4, pp. 647– 661, 2007.

[11] S. Alaei, M. Hajiaghayi, and V. Liaghat, "Online prophet-inequality matching with applications to ad allocation," in *Proceedings of the 13th ACM Conference on Electronic Commerce*, pp. 18–35, 2012.

[12] S. Agrawal and N. R. Devanur, "Fast algorithms for online stochastic convex programming," in *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1405–1424, SIAM, 2014.

[13] S. Agrawal, Z. Wang, and Y. Ye, "A dynamic near-optimal algorithm for online linear programming," *Operations Research*, vol. 62, no. 4, pp. 876–890, 2014.

[14] S. Balseiro, H. Lu, and V. Mirrokni, "Dual mirror descent for online allocation problems," in *International Conference on Machine Learning*, pp. 613–628, PMLR, 2020.

[15] J. Feldman, M. Henzinger, N. Korula, V. S. Mirrokni, and C. Stein, "Online stochastic packing applied to display ad allocation," in *Algorithms – ESA 2010* (M. de Berg and U. Meyer, eds.), (Berlin, Heidelberg), pp. 182–194, Springer Berlin Heidelberg, 2010.

[16] H. Yu, M. Neely, and X. Wei, "Online convex optimization with stochastic constraints," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] J. Yuan and A. Lamperski, "Online convex optimization for cumulative constraints," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[18] X. Y. Gong, V. Goyal, G. Iyengar, D. Simchi-Levi, R. Udwani, and S. Wang, "Online assortment optimization with reusable resources," *Available at SSRN 3334789*, 2019.

[19] V. Goyal, G. Iyengar, and R. Udwani, "Asymptotically optimal competitive ratio for online allocation of reusable resources," *arXiv preprint arXiv:2002.02430*, 2020.

[20] Y. Feng, R. Niazadeh, and A. Saberi, "Near-optimal bayesian online assortment of reusable resources," *Proceedings of the 23rd ACM Conference on Economics and Computation*, 2022..

[21] R. Levi and A. Radovanović, "Provably near-optimal lp-based policies for revenue management in systems with reusable resources," *Operations Research*, vol. 58, no. 2, pp. 503–507, 2010.

[22] Y. Feng, R. Niazadeh, and A. Saberi, "Linear programming based online policies for real-time assortment of reusable resources," *Available at SSRN 3421227*, 2019.

[23] J. J. M. Bront, I. Méndez-Díaz, and G. Vulcano, "A column generation algorithm for choice-based network revenue management," *Operations research*, vol. 57, no. 3, pp. 769–784, 2009.

[24] J. Davis, G. Gallego, and H. Topaloglu, "Assortment planning under the multinomial logit model with totally unimodular constraint structures," *Work in Progress*, 2013.