Dynamic Stochastic Multiple Knapsack Problem with Multi-type Items

December 2, 2025

Abstract

We examine the dynamic stochastic multi-type multiple knapsack problem. Its dynamic programming formulation is computationally prohibitive to solve directly. To address this, we first develop a canonical bid-price control policy, a threshold policy that accepts items above a certain type-value and rejects those below. However, since this policy provides a uniform bid-price across all knapsacks, it cannot distinguish between them and requires a feasibility check for each item. We then develop a pattern-based bid-price control, which assigns a bid-price to an entire item configuration rather than to individual size. This policy offers a superior approximation of the dynamic program's value function. Finally, we propose a dynamic primal policy to capture information potentially absent from the dual formulation. The numerical results show that the performances of the pattern-based bid-price control and dynamic primal policy are better than the traditional bid-price control.

Keywords: multi-type multiple knapsack problem, revenue management, bid-price control.

1 Introduction

The dynamic stochastic multiple knapsack problem with multi-type items extends the classical knapsack problem to a dynamic and multi-knapsack setting, where items of distinct types arrive sequentially, and decisions to accept or reject them must be made immediately without knowledge of future arrivals. Each item is characterized by a type-dependent size and value, and must be placed into one of multiple knapsacks with non-identical capacities. This framework captures critical resource allocation challenges across various domains, including online advertising, cloud resource and energy management.

The business strategy of cloud providers offering fixed-term, pre-paid virtual machine instances presents a classic and impactful resource allocation challenge. This operational challenge can be effectively modeled using the DSMKP, providing a foundational framework for optimizing revenue and resource utilization. Consider the case of Tencent Cloud, a representative provider that sells 1-month reserved instance contracts. These contracts, such as small (Type-S), medium (Type-M), and large (Type-L) instances, are characterized by their specific resource requirements (size) and a fixed, pre-paid price (value).

In this formalization, the provider's infrastructure is abstracted as a set of multiple knapsacks. Each knapsack represents a distinct, homogeneous resource pool–for instance, a cluster of servers with identical hardware–with a capacity defined by its total available standardized "compute units" (e.g., a combination of CPU cores and memory). Customer orders for reserved instances arrive sequentially and stochastically in an online fashion. Upon each arrival, the platform must make an immediate and irrevocable decision: to either reject the request or accept it and assign it to a specific resource pool, thereby consuming a portion of that pool's capacity.

In online advertising, our work employs a foundational framework for ad display allocation. In this model, a platform faces dynamically arriving ad campaigns from a finite set of types. Upon each arrival, it must make an immediate and irrevocable decision: to either reject the campaign or accept it by assigning it to an advertising slot inventory, thereby consuming a portion of that inventory's capacity. This application is aligned with the traditional method of fulfilling guaranteed contracts, as seen in yield optimization literature (Balseiro et al., 2011).

In this formalization, an ad slot inventory is analogous to a "knapsack". It represents a homogeneous pool of future advertising impressions (e.g., the banner on a specific website section). The knapsack's capacity corresponds to the total number of available impressions for that inventory over a planned future period (e.g., the next day or week). For example, the platform manages two distinct inventories with different impression capacities. An item to be allocated is a dynamically arriving ad campaign request. Requests are grouped into a finite set of types, where all campaigns of the same type have the identical contractual requirement and value.

The DSMKP provides a unified formalism to study this essential trade-off between utilizing current capacity and reserving it for future, potentially more valuable opportunities, making it a critical tool for optimizing online decision-making under uncertainty.

These applications underscore the broad applicability of the DSMKP in real-world systems where heterogeneous resources must be allocated dynamically under uncertainty. Developing efficient algorithms for this problem–with guarantees on scalability and robustness–is therefore of significant theoretical and practical interest. This paper aims to address this gap by proposing novel strategies for the DSMKP and validating them in one or more of the above domains.

We develop several policies for DSMKP. The rest of this paper is structured as follows. We review the relevant literature in Section 2. Section 3 presents the bid-price and resolving dynamic primal policies to assign the incoming requests. Section 6 presents the experimental results and provides insights gained from implementing dynamic primal. Conclusions are shown in Section 7.

2 Literature Review

The Multiple Knapsack Problem (MKP) (Martello and Toth, 1990) is a well-studied combinatorial optimization problem with broad practical implications. Existing literature has largely concentrated on establishing bounds or competitive ratios for general MKP instances Khuri et al. (1994); Ferreira et al. (1996); Pisinger (1999); Chekuri and Khanna (2005). In contrast, our work focuses on analyzing the specific solution characteristics of the multi-type multiple knapsack problem, providing a more tailored understanding of its dynamic form.

A closely related stream of research addresses the Dynamic Stochastic Knapsack Problem, as examined in Kleywegt and Papastavrou (1998, 2001) and Papastavrou et al. (1996). These studies typically consider a single knapsack setting where requests arrive sequentially, with resource requirements and rewards revealed only upon arrival. However, they do not incorporate multiple knapsacks, which limits their applicability to more complex resource allocation settings.

In comparison, the DSMKP generalizes this framework by introducing multiple knapsacks, significantly increasing the complexity of real-time decision-making. Moreover, in DSMKP settings, additional information–such as the arrival probabilities of different item types–is often available, enabling more informed and potentially more effective allocation strategies.

Research on dynamic and stochastic variants of the Multiple Knapsack Problem (MKP) remains limited and can be divided into two main streams. The first addresses dynamic online settings where items arrive sequentially without prior information, as seen in studies on the online multiple knapsack problem (Bienkowski et al., 2020; Sun et al., 2020), which aim to develop competitive algorithms. The second investigates stochastic models, such as the two-stage stochastic MKP where the availability of knapsacks can vary across different scenarios (Tönissen et al., 2017). Departing from these, our work considers a setting with sequentially arriving items of known types and known arrival probabilities, enabling the design of proactive and more effective allocation strategies.

Generally speaking, the DSMKP problem is related to the Revenue Management (RM) problem. RM has been extensively studied in industries such as airlines, hotels, and car rentals, where the core challenge is to dynamically allocate perishable inventory to maximize revenue (van Ryzin and Talluri, 2005). Network revenue management (NRM) extends this framework by considering multiple interconnected resources (e.g., flight legs, hotel nights) and interdependent demands (Williamson, 1992).

The standard NRM problem is typically formulated as a dynamic programming (DP) model, where decisions involve accepting or rejecting requests based on their revenue contribution and remaining capacity (Talluri and van Ryzin, 1998).

A fundamental challenge, however, is the "curse of dimensionality", as the state space grows exponentially with problem size, making exact DP solutions computationally intractable. Consequently, various approximate control policies have been developed, including bid-price controls (Adelman, 2007; Bertsimas and Popescu, 2003), booking limits (Gallego and van Ryzin, 1997), and dynamic programming decomposition methods (Talluri and van Ryzin, 2006; Liu and van Ryzin, 2008).

However, a significant challenge arises because the number of states grows exponentially with the problem size, rendering direct solutions computationally infeasible. To address this, various control policies have been proposed, such as bid-price (Adelman, 2007; Bertsimas and Popescu, 2003), booking limits (Gallego and van Ryzin, 1997), and dynamic programming decomposition (Talluri and van Ryzin, 2006; Liu and van Ryzin, 2008).

In our problem, items often request multiple units simultaneously, requiring decisions that must be made on an all-or-none basis for each request. This requirement introduces significant complexity in managing group arrivals (Talluri and van Ryzin, 2006).

A notable study addressing group-like arrivals in revenue management examines hotel multi-day stays (Bitran and Mondschein, 1995; Goldman et al., 2002; Aydin and Birbil, 2018). While these works focus on customer classification and room-type allocation, they do not prioritize real-time assignment. The work of Zhu et al. (2023), which addresses the high-speed train ticket allocation and processes individual seat requests and implicitly accommodates group-like traits through multi-leg journeys (e.g., passengers retaining the same seat across connected segments).

3 The Model

We consider a system with a set $\mathcal{N} = \{1, 2, \ldots, N\}$ of knapsacks, where each knapsack j has a capacity $c_j \in \mathbb{Z}^+$. A finite set of item types $\mathcal{M} = \{1, 2, \ldots, M\}$ is given, where any item of type i is characterized by a size $w_i \in \mathbb{Z}^+$ and a profit $r_i \in \mathbb{Z}^+$. In the context of online advertising, for instance, size and profit correspond to the impressions required by an ad campaign and the revenue it generates, respectively. The item types are ordered such that their profit-to-weight ratios, r_i/w_i, are monotonically increasing in i.

Requests for these items arrive sequentially. Upon the arrival of a request (which specifies its type), the seller must immediately decide whether to accept or reject it. If accepted, the seller must also assign it to a specific knapsack with sufficient remaining capacity. Each item must be placed whole into a single knapsack; partial assignments or reassignments are not permitted.

To model this problem, we adopt a dynamic programming framework based on discrete time periods $t = 1, 2, \ldots, T$. In each period, at most one request arrives. Let $\lambda_i(t)$ denote the probability that a request for an item of type $i \in \mathcal{M}$ arrives at time t. These probabilities satisfy $\sum_{i=1}^{M} \lambda_i(t) \leq 1$ for all t, and we define $\lambda_0(t) = 1 - \sum_{i=1}^{M} \lambda_i(t)$ as the probability of no arrival in period t. Arrival events are assumed to be independent across time periods.

The system state is the vector of remaining capacities $\mathbf{C} = (c_1, \ldots, c_N)$. When an item of type i arrives, the seller chooses a decision variable $u_{i,j}(t) \in \{0, 1\}$ for each knapsack j. The feasible set $U(\mathbf{C}(t))$ is defined by:

$$U(\mathbf{C}(t)) = \left\{ u_{i,j}(t) \in \{0,1\} \;\middle|\; \begin{array}{ll} \text{(a)} & \sum_{j=1}^{N} u_{i,j}(t) \leq 1 \quad \forall i \in \mathcal{M} \\ \text{(b)} & w_i u_{i,j}(t) \leq c_j \quad \forall i \in \mathcal{M}, \forall j \in \mathcal{N} \end{array} \right\}.$$

Constraint (a) ensures the item is assigned to at most one knapsack. Constraint (b) ensures that if the item is assigned to knapsack j ($u_{i,j}(t) = 1$), its weight w_i does not exceed c_j. The original vector form of this constraint, $w_i u_{i,j}(t)\mathbf{e}_j \leq \mathbf{C}$ (where $\mathbf{e}_j$ is the j-th standard basis vector), is mathematically equivalent to (b).

Let $v_t(\mathbf{C})$ denote the value function, representing the maximum expected revenue obtainable from period t onward, given the current remaining capacity vector $\mathbf{C}$. The Bellman equation is given by:

$$v_t(\mathbf{C}(t)) = \max_{u_{i,j}(t) \in U(\mathbf{C}(t))} \left\{ \sum_{i=1}^{M} \lambda_i(t) \big(\sum_{j=1}^{N} r_i u_{i,j}(t) + v_{t+1}(\mathbf{C}(t) - w_i u_{i,j}(t) \mathbf{e}_j) \big) + \lambda_0(t) v_{t+1}(\mathbf{C}(t)) \right\} \quad (1)$$

The boundary condition is $v_{T+1}(\mathbf{C}) = 0$ for all $\mathbf{C} \geq \mathbf{0}$, indicating that no more revenue can be earned after the final period T.

Let $\mathbf{C}(0) = (C_1, C_2, \ldots, C_N)$ be the initial capacity vector. The objective is to compute $v_1(\mathbf{C}(0))$, the maximum total expected revenue over the entire horizon from $t = 1$ to $t = T$, and to find the policy of item assignments that achieves this value.

Solving the dynamic programming problem presented in Equation (1) is computationally intractable for realistic problem sizes due to the curse of dimensionality inherent in the large state space.

To overcome this challenge, we propose a heuristic assignment policy. We first outline a traditional bid-price control policy. We then enhance this approach by introducing a novel bid-price control policy that leverages patterns.

3.1 BPC Policy

Bid-price control is a classical and widely studied methodology in network revenue management. The core idea is to set thresholds, known as bid prices, that represent the opportunity cost of consuming one unit of capacity. An item is accepted only if its revenue exceeds the estimated opportunity cost of the capacity it requires.

Typically, these bid prices are derived from the shadow prices of the capacity constraints in a deterministic approximation of the underlying stochastic problem. In this section, we detail the implementation of a bid-price control policy for our model.

We begin by formulating a deterministic linear programming (LP) approximation, specifically the LP relaxation of a multi-type multiple knapsack problem. This model uses expected demand over the horizon. Let x_{ij} denote the number of type i items assigned to knapsack j, and let $d_i = \sum_{t=1}^{T} \lambda_i(t)$ represent the expected number of requests for type i. The formulation is as follows:

$$\begin{aligned}
\max \quad & \sum_{i=1}^{M} \sum_{j=1}^{N} r_i x_{ij} && (2)\\
\text{s.t.} \quad & \sum_{j=1}^{N} x_{ij} \leq d_i, \quad i \in \mathcal{M}, && (3)\\
& \sum_{i=1}^{M} w_i x_{ij} \leq c_j, j \in \mathcal{N}, && (4)\\
& x_{ij} \geq 0, \quad i \in \mathcal{M}, j \in \mathcal{N}.
\end{aligned}$$

The objective (2) is to maximize total expected revenue. Constraint (3) ensures the total number of accepted type i items does not exceed its expected demand. Constraint (4) ensures the total weight in each knapsack j does not exceed its initial capacity C_j.

The monotonic increase of the profit-to-weight ratio r_i/w_i with type index i implies that items with a higher index are more profitable per unit of capacity. Consequently, the optimal solution to the LP relaxation exhibits a greedy structure, preferentially utilizing higher-indexed item types. This structural property is formalized in Proposition 1.

Lemma 1. *For the LP relaxation of the* MKP *problem, there exists an index $\tilde{i}$ such that the optimal solutions satisfy the following conditions: $x^*_{ij} = 0$ for all j, $i = 1, \ldots, \tilde{i} - 1$; $\sum_{j=1}^N x^*_{ij} = d_i$ for $i = \tilde{i} + 1, \ldots, M$; $\sum_{j=1}^N x^*_{ij} = \frac{L - \sum_{i=\tilde{i}+1}^M d_i w_i}{w_{\tilde{i}}}$ for $i = \tilde{i}$.*

The dual of LP relaxation of the MKP problem is:

$$\begin{aligned} \min \quad & \sum_{i=1}^M d_i z_i + \sum_{j=1}^N c_j \beta_j \\ \text{s.t.} \quad & z_i + \beta_j w_i \geq r_i, \quad i \in \mathcal{M}, j \in \mathcal{N} \\ & z_i \geq 0, i \in \mathcal{M}, \beta_j \geq 0, j \in \mathcal{N}. \end{aligned} \tag{5}$$

In (5), β_j can be interpreted as the bid-price for one size in knapsack j. A request is only accepted if the revenue it generates is no less than the sum of the bid prices of the sizes it uses. Thus, if $r_i - \beta_j w_i \geq 0$, meanwhile, the capacity allows, we will accept the item type i. And choose knapsack $j^* = \arg\max_j \{r_i - \beta_j w_i\}$ to allocate that item.

Proposition 1. *The optimal solution to problem (5) is given by $z_1 = \ldots = z_{\tilde{i}} = 0$, $z_i = \frac{r_i w_{\tilde{i}} - r_{\tilde{i}} w_i}{w_{\tilde{i}}}$ for $i = \tilde{i} + 1, \ldots, M$ and $\beta_j = \frac{r_{\tilde{i}}}{w_{\tilde{i}}}$ for all j.*

According to Proposition 1, the decision inequality becomes $r_i - \beta_j w_i = r_i - \frac{r_{\tilde{i}}}{w_{\tilde{i}}} w_i \geq 0$. This establishes the threshold policy: reject item type $i, i < \tilde{i}$ and accept item type $i, i \geq \tilde{i}$.

Algorithm 1: Bid-Price Control

```
1  for $t = 1, \ldots, T$ do
2      Observe a request of item type $i$;
3      Solve problem (5) with $\boldsymbol{d}^{[t,T]}$ and $\mathbf{C}(t)$;
4      Obtain $\tilde{i}$ such that the aggregate optimal solution is $x e_{\tilde{i}} + \sum_{i=\tilde{i}+1}^M d_i(t) e_i$;
5      if $i \geq \tilde{i}$ and $\max_{j \in \mathcal{N}} c_j(t) \geq w_i$ then
6          Set $k = \arg\min_{j \in \mathcal{N}} \{c_j(t) | c_j(t) \geq w_i\}$ and break ties;
7          Assign the item to knapsack $k$, let $c_k(t+1) \leftarrow c_k(t) - w_i$ ;
8      else
9          Reject the request;
10     end
11 end
```

Let $z(\text{BPC})$, $z(\text{DLP})$ denote the optimal value of (5) and the LP relaxation of the MKP problem with expected demand, respectively. Then we have $z(\text{BPC}) = z(\text{DLP})$.

However, the BPC policy has two drawbacks. First, the capacity feasibility need to be checked when to accept a request. Second, when capacities permit, the policy treats all knapsacks as equally preferable, making no distinction among them.

Example 1. *Consider an instance with $M = 3$, $N = 4$, $w_1 = 3, w_2 = 4, w_3 = 5$, $r_1 = 4, r_2 = 6, r_3 = 8$, $\boldsymbol{C} = [7, 8, 8, 4]$ and $T = 8$. The stationary arrival probability is $\lambda_1 = \lambda_3 = \frac{1}{4}$, $\lambda_2 = \frac{1}{2}$, giving an expected demand vector $\boldsymbol{d} = (2, 4, 2)$.*

Under traditional bid-price control, the bid-price for each knapsack j is $\beta_j^* = \frac{4}{3}$, yielding a total expected reward $z(BPC) = \frac{124}{3}$. This solution exhibits two major drawbacks:

It fails to differentiate between knapsacks, assigning them a uniform bid-price. It permits potentially infeasible assignments. For instance, even though $r_3 - \beta_4^* * w_3 > 0$, assigning a type 3 item (with weight 5) to knapsack 4 (with capacity 4) is infeasible.

3.2 BPC Policy Based on Patterns (BPP)

In the BPC policy, we treat each size in each knapsack separately. However, a request always occupies a series of consecutive sizes within a knapsack. In order to capture the value of an item, it would be helpful to consider the sizes of an item jointly. In the following, we consider an alternative bid-price formulation based on the idea of pattern and we derive a different set of bid-prices and hence a better-performing control policy. The core idea, as we explain next, is to reformulate the dynamic system with patterns.

Let $\boldsymbol{h} = [h_1, \ldots, h_M]$ denote a pattern for a knapsack, where each h_i denotes the number of type-i items assigned. A pattern $\boldsymbol{h}$ is feasible for a knapsack with capacity c_j if it satisfies $\sum_{i=1}^{M} w_i h_i \leq c_j$. For any knapsack j, we construct a set $S(c_j)$ which contains all feasible patterns. For a more general case where $\boldsymbol{h}$ is subject to additional constraints, we consider the extension in Section 4.2.

A request of type i can be accepted only if there exists at least one knapsack j with sufficient remaining capacity $c_j \geq w_i$. This is equivalent to the existence of a feasible pattern $\boldsymbol{h} \in S(c_j)$ for that knapsack that includes at least one item of type i (i.e., $h_i \geq 1$).

To formalize this, we define the maximum value obtainable from accepting a type i request. For a given knapsack j and pattern $\boldsymbol{h}$ that can accommodate the item ($\boldsymbol{h} \in S(c_j)$, $h_i \geq 1$), the value of placing it in j is the sum of the immediate reward r_i and the future value $v_{t+1}(\boldsymbol{C} - e_j^T w_i)$ starting from the updated capacity.

We therefore define q_i as the maximum of this value over all feasible placement options:

$$q_i = \begin{cases} \max_{j:\exists \boldsymbol{h} \in S(c_j), h_i \geq 1} \left\{ v_{t+1} \left(\boldsymbol{C} - e_j^T w_i \right) + r_i \right\} & \text{if such a knapsack } j \text{ exists,} \\ 0 & \text{otherwise} \end{cases}$$

In this definition, the maximization evaluates the optimal placement of a type-i item by considering all knapsacks j for which there exists at least one feasible pattern $\boldsymbol{h} \in S(c_j)$ capable of holding it ($h_i \geq 1$).

Now we present the DP formulation from the pattern perspective. Recall that $v_t(\boldsymbol{C})$ denotes the maximal expected value-to-go at time t, given the remaining capacity $\boldsymbol{C}$. The dynamic programming can be reformulated as follows:

$$\begin{aligned} & v_t(\boldsymbol{C}) \geq \mathbb{E}_{i\sim\lambda(t)}\Bigg[\max\{q_i, v_{t+1}(\boldsymbol{C})\}\Bigg], \forall t, \\ & v_{T+1}(\boldsymbol{C}) = 0 \end{aligned} \tag{6}$$

The maximization compares the value of accepting i (via the maximization of q_i) and rejecting i (retaining $v_{t+1}(\boldsymbol{C})$). If no such j exists, the request i is rejected.

We can solve the following program to compute $v_1(\boldsymbol{C})$ for any given capacity $\boldsymbol{C}$:

$$\begin{aligned} \min \quad & v_1(\boldsymbol{C}) \\ \text{s.t.} \quad & v_t(\boldsymbol{C}) \geq \mathbb{E}_{i\sim\lambda(t)}\Bigg[\max\Big\{q_i, v_{t+1}(\boldsymbol{C})\Big\}\Bigg], \\ & v_{T+1}(\boldsymbol{C}) \geq 0. \end{aligned} \tag{7}$$

Solving (7) remains computationally intractable. We therefore adopt an Approximate Dynamic Programming (ADP) approach (Adelman, 2007) to approximate the value function $v_t(\boldsymbol{C})$. Our proposed approximation is:

$$\hat{v}_t(\boldsymbol{C}) = \theta_t + \sum_{j=1}^{N} \max_{\boldsymbol{h}\in S(c_j)} \{\sum_{i=1}^{M} \beta_{ij}^{\dagger} h_i\}. \tag{8}$$

The term $\beta_{ij}^{\dagger}$ can be regarded as the approximated value for each type i in knapsack j. Unlike traditional linear approximations, our approach retains the linear term θ_t but introduces a nonlinear component for each knapsack j. Specifically, we maximize the linear combination $\sum_{i=1}^{M} \beta_{ij}^{\dagger} h_i$ over the feasible set $S(c_j)$.

Substituting (8) into (7), we have:

$$\theta_t - \theta_{t+1} = \hat{v}_t(\boldsymbol{C}) - \hat{v}_{t+1}(\boldsymbol{C}) \geq \sum_i \lambda_i(t) \max\{q_i - v_{t+1}(\boldsymbol{C}), 0\} \tag{9}$$

For the case where there exists a knapsack j satisfying both conditions: $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h}\in S(c_j)} \sum_i \beta_{ij}^{\dagger} h_i$ and $h_i^* \geq 1$, we establish the value difference:

$$\begin{aligned} & v_{t+1}(\boldsymbol{C} - e_j^T w_i) - v_{t+1}(\boldsymbol{C}) \\ = & \max_{\boldsymbol{h}\in S(c_j - w_i)} \{\sum_i \beta_{ij}^{\dagger} h_i\} - \max_{\boldsymbol{h}\in S(c_j)} \{\sum_i \beta_{ij}^{\dagger} h_i\} \\ = & -\beta_{ij}^{\dagger} \leq 0 \end{aligned}$$

The acceptance threshold is then defined as:

$$\alpha_i = \max\left\{\max_j \left\{r_i - \beta_{ij}^{\dagger}\right\}, 0\right\},$$

with $\alpha_i = 0$ when no qualifying knapsack j exists.

Let $\gamma_j = \max_{\boldsymbol{h} \in S(c_j)}\{\sum_i \beta_{ij}^{\dagger} h_i\}$. This yields:

$$\begin{aligned} \theta_1 &= \sum_{t=1}^{T}(\theta_t - \theta_{t+1}) \geq \sum_t \sum_i \alpha_i \lambda_i(t) = \sum_i d_i \alpha_i \\ \hat{v}_1(\boldsymbol{C}) &= \sum_i d_i \alpha_i + \sum_j \gamma_j \end{aligned}$$

Since $\hat{v}_1(\boldsymbol{C})$ constitutes a feasible solution to (7), we have $\hat{v}_1(\boldsymbol{C}) \geq v_1(\boldsymbol{C})$.

If all j exist for $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger} h_i, h_i^* \geq 1$, the corresponding bid-price problem can be expressed as:

$$\begin{aligned} \min \quad & \sum_{i=1}^{M} \alpha_i d_i + \sum_{j=1}^{N} \gamma_j \\ \text{s.t.} \quad & \alpha_i + \beta_{ij}^{\dagger} \geq r_i, \quad \forall i, j, \\ & \sum_{i=1}^{M} \beta_{ij}^{\dagger} h_i \leq \gamma_j, \quad \forall j, \boldsymbol{h} \in S(c_j), \\ & \alpha_i \geq 0, \forall i, \quad \beta_{ij}^{\dagger} \geq 0, \forall i, j \\ & \gamma_j \geq 0, \quad \forall j. \end{aligned} \tag{10}$$

α_i represents marginal revenue for type i. $\beta_{ij}^{\dagger}$ represents the cost for type i assigned in knapsack j. γ_j represents the capacity cost associated with knapsack j.

If for a given item type i, no knapsack j contains an optimal allocation $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger} h_i$ with $h_i^* \geq 1$, then the variable α_i must be zero. This indicates that the constraint associated with item type i and knapsack j is redundant. Although these constraints appear difficult to enforce in advance, the following lemma reveals that in practice we can avoid imposing additional restrictions. Simply solving problem (10) will inherently satisfy all required conditions.

Lemma 2. *Define $\mathcal{J}_i = \{j \in \mathcal{N} \mid r_i - \beta_{ij}^{\dagger *} \geq r_i - \beta_{ik}^{\dagger *}, \forall k \in \mathcal{N}, r_i - \beta_{ij}^{\dagger *} > 0\}$. If $\mathcal{J}_i = \varnothing$, the first set of constraints for i is redundant. If $\mathcal{J}_i \neq \varnothing$, then there exists a $j' \in \mathcal{J}_i$ such that:*

$$\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_{j'})} \sum_i \beta_{ij'}^{\dagger *} h_i, h_i^* \geq 1.$$

This lemma guarantees that when such a j' exists, the first set of constraints for i becomes active; otherwise, it remains inactive. Consequently, we have $z(\text{BPP}) = \hat{v}_1(\boldsymbol{C})$.

The control policy is then defined as follows. For the arriving type-i item, if $\alpha_i > 0$, accept the type i. Assign the type i to a knapsack $j \in \mathcal{J}_i$ satisfying:

$$\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger *} h_i, h_i^* \geq 1.$$

If $\alpha_i = 0$ and there exists a knapsack j such that: $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger *} h_i, h_i^* \geq 1$, accept the type

i and assign it to knapsack j; otherwise, reject the type i.

Let $z(\text{BPC})$ and $z(\text{BPP})$ denote the expected optimal value of (5) and (10), respectively.

Proposition 2. *We have $z(BPC) \geq z(BPP)$.*

Both bid-price approaches give upper bounds on the value function at any state, meanwhile it follows that BPP provides a tighter approximation to the value function more accurately than BPC does.

Under the approximation (8), the BPP policy operates as follows:

Algorithm 2: Bid-Price Control Based on Patterns

```
1  for $t = 1, \dots, T$ do
2      Observe a request of item type $i$;
3      Solve problem (10) with $\boldsymbol{d}^{[t,T]}$;
4      if $\alpha_i > 0$ then
5          Assign the type $i$ to a knapsack $j \in \mathcal{J}_i$ satisfying:
               $$\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger *} h_i, h_i^* \geq 1.$$
           Let $c_j(t+1) \leftarrow c_j(t) - w_i$;
6      else
7          if There exists $j$ such that $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta_{ij}^{\dagger} h_i, h_i^* \geq 1$ then
8              Assign the item in knapsack $j$, let $c_j(t+1) \leftarrow c_j(t) - w_i$;
9          else
10             Reject the request;
11         end
12     end
13 end
```

Unlike the traditional BPC, the BPP does not require explicit feasibility checks on capacity. However, it still needs to verify whether the optimal pattern contains the given request. This introduces a new challenge, as bid-price policies are inherently derived from a dual formulation, which may inherently omit key information preserved in the primal problem.

Example 2. *Continue with the above example:*

For the BPP, the optimal bid prices are $\boldsymbol{\beta}_1^* = [4, 4, 4, 6]$, $\boldsymbol{\beta}_2^* = [6, 6, 6, 6]$, and $\boldsymbol{\beta}_3^* = [10, 8, 8, 8]$. The resulting objective value is $z(BPP) = 40$, which is less than $z(BPC) = \frac{124}{3}$. We use item type 3 to demonstrate the assignment analysis. The reduced reward vector for this type, $r_3 - \boldsymbol{\beta}_3^* = [-2, 0, 0, 0]$, has a negative first element, indicating that assignment to knapsack 1 is unprofitable. Furthermore, from the knapsack's perspective, the patterns generated for knapsack 4 in the optimal solution are $[0, 1, 0]$ and $[1, 0, 0]$, confirming that type 3 is also excluded from this knapsack. Conversely, this item type can be assigned to knapsacks 2 and 3, as evidenced by the non-zero third elements in the respective generated patterns.

While this verification is cumbersome under pure bid-price frameworks, the primal problem offers a more direct way to guarantee the existence of such a request-pattern match. To address this gap, we propose the dynamic primal formulation.

4 Dynamic Primal Based on Patterns

In Section 3.2, we proposed a bid-price control policy based on the patterns. However, the bid-price policies are established via a dual formulation, which might lose some information contained in the primal problem. In this section, we propose a policy called dynamic primal based on patterns (DPP).

Let $y_{j\boldsymbol{h}}$ denote the proportion of pattern $\boldsymbol{h}$ used in knapsack j. The primal problem can be formulated as:

$$\begin{aligned} \max \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} r_i x_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{N} x_{ij} \le d_i, \quad i \in \mathcal{M}, \\ & x_{ij} \le \sum_{\boldsymbol{h}\in S(c_j)} h_i y_{j\boldsymbol{h}}, \quad i \in \mathcal{M}, j \in \mathcal{N}, \\ & \sum_{\boldsymbol{h}\in S(c_j)} y_{j\boldsymbol{h}} \le 1, \quad j \in \mathcal{N}. \end{aligned} \tag{11}$$

The first set of constraints demonstrate that for each item type i, the sum of assigned items and unassigned items equals the total demand. The second set of constraints shows that the number of items of type i assigned in knapsack j is not larger than the sum of h_i (the count of type i items in pattern $\boldsymbol{h}$) weighted by the pattern proportions $y_{i\boldsymbol{h}}$. The total proportion of patterns uesd in knapsack j cannot exceed 1.

Proposition 3. *If the optimal solution x_{ij}^* to (11) satisfies $x_{ij}^* > 0$ for a pair (i, j), then for that knapsack j, we have*

$$\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h}\in S(c_j)} \sum_i \beta_{ij}^{\dagger} h_i, h_i^* \ge 1.$$

In contrast to Lemma 2, this lemma ensures the existance of a knapsack j to accommodate item type i when $x_{ij}^* > 0$. This eliminates the need for explicit existance verification.

4.1 Solve the Dynamic Primal

The pattern $\boldsymbol{h}$ is efficient for knapsack j if and only if, for some $(\alpha_1, \ldots, \alpha_M, \gamma_j)$ (except that $\alpha_i = r_i, \forall i$), $\boldsymbol{h}$ is the optimal solution to

$$\max_{\boldsymbol{h}} \sum_{i=1}^{M} (r_i - \alpha_i) h_i - \gamma_j$$

To generate all efficient patterns, we need to solve the subproblem for each knapsack j:

$$\begin{aligned} \max \quad & \sum_{i=1}^{M} (r_i - \alpha_i) h_i - \gamma_j \\ \text{s.t.} \quad & \sum_{i=1}^{M} w_i h_i \leq c_j, \\ & h_i \in \mathbb{N}, \quad i \in \mathcal{M}. \end{aligned} \tag{12}$$

If the optimal value of (12) is larger than 0, the primal (11) reaches the optimal. Otherwise, a new pattern can be generated.

One important fact is that only efficient sets are used in the solution to (11). Therefore, if $y^*_{j\boldsymbol{h}} > 0$ is the optimal solution to (11), then $\boldsymbol{h}$ is an efficient pattern.

A pattern $\boldsymbol{h}$ is dominant if there is no distinct pattern $\boldsymbol{h}'$ where every component of $\boldsymbol{h}'$ is greater than or equal to the corresponding component of $\boldsymbol{h}$. The efficient pattern is a dominating pattern. (If $\alpha_i = r_i$, (11) reaches the optimal and no pattern will be generated.)

The relation between the capacity and the demand shows the different structure of the optimal solution.

Lemma 3. *When $\sum_{i=1}^{M} d_i w_i < \sum_{j=1}^{N} c_j$, we have $\gamma^*_j = 0, \forall j$, $\beta^{\dagger *}_{ij} = 0, \forall i, j$ and $\alpha^*_i = r_i, \forall i$. There exists at least one knapsack j such that $\sum_{\boldsymbol{h} \in S(c_j)} y^*_{j\boldsymbol{h}} < 1$.*

When $\sum_{i=1}^{M} d_i w_i \geq \sum_{j=1}^{N} c_j$, we have $\sum_{\boldsymbol{h} \in S(c_j)} y^*_{j\boldsymbol{h}} = 1, \forall j$.

Algorithm 3: Dynamic Primal

```
1  for $t = 1, \dots, T$ do
2      Observe a request of type $i$;
3      if $c_j(t) = w_i, \exists j$ then
4          Assign the item to knapsack $j$;
5          continue
6      end
7      Solve problem (11) with $\boldsymbol{d}^{[t,T]}$ ;
8      Obtain an optimal solution $x_{ij}$ ;
9      if $\max_j \{x_{ij}\} > 0$ then
10         Set $k = \arg\max_j \{x_{ij}\}$ and break ties;
11         Assign the item to knapsack $k$, let $c_k(t+1) \leftarrow c_k(t) - w_i$;
12     else
13         Reject the request;
14     end
15 end
```

Meanwhile, it guarantees feasible placement. Once a request is accepted, the policy ensures it can be assigned to a suitable knapsack without additional feasibility checks.

Example 3. *For the dynamic primal, the optiaml solution is given by* $\boldsymbol{x}_1^* = [1, 1, 0, 0]$*,* $\boldsymbol{x}_2^* = [1, 0, 2, 1]$*,* $\boldsymbol{x}_3^* = [0, 1, 0, 0]$*. This indicates that type 1 can be assigned to knapsacks 1 or 2, type 2 cannot be assigned to knapsack 2, type 3 can only be assigned to knapsack 2.*

In the optimal solution, the efficient patterns for each knapsack are $[1, 1, 0]$*,* $[1, 0, 1]$*,* $[0, 2, 0]$*,* $[0, 1, 0]$*. For instance,* $[1, 1, 0]$ *shows that knapsack 1 can hold one item of type 1 and one of type 2.*

Using the decision variable x_{ij} *to represent these assignments is a straightforward and easily implementable approach.*

4.2 Extension

Sometimes, we do not need to generate all feasible patterns. The generated patterns may need to satisfy certain restrictions, such as certain items not being allowed in certain knapsacks, or relationships between items.

For example, we can add constraints like: $h_i = 0$ (Item i is excluded), $h_i + h_k \leq 1$ (Items i and k are mutually exclusive). $h_i \leq \max \text{count}_i$ (An upper bound on the number of item i).

To represent these scenarios in a general form, we consider the following extended subproblem for knapsack j:

$$\begin{aligned}
\max & \sum_{i=1}^{M} (r_i - \alpha_i)\, h_i - \gamma_j \\
\text{s.t.} & \sum_{i=1}^{M} w_i h_i \leq c_j \\
& \mathbf{A}_j \mathbf{h} \leq \mathbf{b}_j \\
& \mathbf{h} \in \mathbb{N}^M \\
& h_i = 0, \quad \forall i \notin \mathcal{F}_j
\end{aligned}$$

Here, the second constraint $\mathbf{A}_j \mathbf{h} \leq \mathbf{b}_j$ encompasses various linear restrictions on the pattern, such as mutual exclusion, dependency, and quantity limits. The set $\mathcal{F}_j$ contains the items permitted for knapsack j.

5 Evaluation on the Asymptotic Loss

In this section, we examine the notion of loss in revenue management. To establish benchmarks, we define two reference values for the base problem instance: the offline optimal value V^{off}, which represents the maximum revenue attainable with perfect knowledge of all future demand realizations, and the offline relaxed optimal value V^{HO}, which is the optimal value of the offline problem with integrality constraints relaxed. By definition, $V^{\text{HO}} \geq V^{\text{off}}$. The expected loss of a policy π is bounded by $E[\text{loss}]^{\pi} = V^{\text{off}} - V^{\pi} \leq V^{\text{HO}} - V^{\pi}$. To obtain a tractable upper bound, we will therefore measure the loss of policy π through the difference $V^{\text{HO}} - V^{\pi}$.

Direct analysis of this loss term, however, is often intractable. To evaluate policy performance systematically, we adopt the standard asymptotic scaling framework prevalent in revenue management literature (see Gallego and van Ryzin (1997)). Let $\theta \in \mathbb{N}$ be a scale parameter. For a given base instance, we generate a sequence of scaled problems. In the θ-th problem, the resource capacity vector is scaled to $\boldsymbol{C}(\theta) = \theta \boldsymbol{C}$, and the selling horizon is extended to θT periods. The arrival rates are defined as $\lambda_i^t(\theta) = \lambda_i^{\lceil t/\theta \rceil}$ for $t \in [\theta T]$.

Let V_θ^π be the expected revenue of policy π in the scaled problem. Within this framework, we will study the asymptotic behavior of the relative loss $(V_\theta^{\text{HO}} - V_\theta^\pi)/V_\theta^{\text{HO}}$ as the scale parameter $\theta \to \infty$. This provides the core metric for assessing the asymptotic optimality of policy π. For the DPP policy, we have Proposition 4.

Proposition 4. *For the scaled problem, we have* $V_\theta^{HO} - V_\theta^{DPP} = O(1)$.

Here we present a brief roadmap for the proof of Proposition 4. First, we consider a variant of (11) by imposing lower bounds for $\boldsymbol{x}$. For each sample path, we decompose the total loss between DPP and HO into θT increments, where each increment is characterized by the gap between two objective values of the variants with different lower bounds for $\boldsymbol{x}$. Then we upper bound each increment. We show that each increment is uniformly upper bounded by a constant that depends only on $\{r_i\}$. As a result, the sum of the increments converges to a constant.

Proof of Proposition 4

Let γ_{ij}, γ_{i0} denote the number of type i accepted in capacity j and rejected by DPP, respectively. We consider the following variant of (11).

$$
\begin{aligned}
OPT(\boldsymbol{C}, d, \gamma): \quad \max \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} r_i x_{ij} \\
\text{s.t.} \quad & \sum_{j=1}^{N} x_{ij} + x_{i0} = d_i, \quad i \in \mathcal{M}, \\
& x_{ij} \le \sum_{\boldsymbol{h} \in S(c_j)} h_i y_{j\boldsymbol{h}}, \quad i \in \mathcal{M}, j \in \mathcal{N}, \\
& \sum_{\boldsymbol{h} \in S(c_j)} y_{j\boldsymbol{h}} \le 1, \quad j \in \mathcal{N} \\
& x_{ij} \ge \gamma_{ij}, \quad i \in \mathcal{M}, j \in \mathcal{N} \cup \{0\}.
\end{aligned} \tag{13}
$$

Note that comparing to the primal, we add a set of constraints $x_{ij} \ge \gamma_{ij}$ for all $i \in \mathcal{M}, j \in \mathcal{N} \cup \{0\}$. It is clear that (11) equals $OPT(\boldsymbol{C}, d, 0)$.

We use superscripts to denote realized values from a sample path. Specifically, for $t_1 \le t_2$, let $d_i^{[t_1,t_2]}$ be the realized demand for type i during $[t_1, t_2]$, and let $\gamma_{ij}^{[1,t)}$ be the number of type i requests allocated to knapsack j by the policy during $[1, t)$. The total realized demand over the horizon is therefore $d^{[1,T]}$.

$OPT(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t+1)})$ can be interpreted as the total reward obtained under a virtual policy where we first follow the heuristic policy during $[1, t+1)$ and then from time $t+1$ we follow the optimal solution assuming that we know the future demands. Let $x^*_{i^t j^t}(t)$ denote the optimal solution for $\text{OPT}(\boldsymbol{C}(t), d^{[t,T]}, 0)$ at time t, where i^t indicates the arriving type at time t, j^t indicates the assigned knapsack at time t.

The following lemma shows an important property of (13).

Lemma 4. *Given $\hat{d}$, for any $1 \leq t_1 \leq t_2 \leq T+1$, we have*

$$OPT(\boldsymbol{C}(1), \hat{d} + d^{[1,t_2)}, \gamma^{[1,t_2)}) = \sum_i r_i \sum_j \gamma_{ij}^{[1,t_1)} + OPT(\boldsymbol{C}(t), \hat{d} + d^{[t_1,t_2)}, \gamma^{[t_1,t_2)})$$

Proof of Lemma 4

For any optimal solution x^* of $OPT(\boldsymbol{C}(t), \hat{d} + d^{[t_1,t_2)}, \gamma^{[t_1,t_2)})$, $x^* + \gamma^{[1,t_1)}$ is a feasible solution of $OPT(\boldsymbol{C}(1), \hat{d} + d^{[1,t_2)}, \gamma^{[1,t_2)})$. For any optimal solution x^* of $OPT(\boldsymbol{C}(1), \hat{d} + d^{[1,t_2)}, \gamma^{[1,t_2)})$, $x^* - \gamma^{[1,t_1)}$ is a feasible solution of $OPT(\boldsymbol{C}(t), \hat{d} + d^{[t_1,t_2)}, \gamma^{[t_1,t_2)})$ because $x^* - \gamma^{[1,t_1)} \geq \gamma^{[1,t_2)} - \gamma^{[1,t_1)} = \gamma^{[t_1,t_2)}$. ■

For one sample path ω, the HO under ω is $OPT(\boldsymbol{C}, d^{[1,T]}, 0)$. From Lemma 4, by setting $\hat{d} = 0$ and $t_1 = t_2 = T+1$, we can see that the total revenue collected under ω is $OPT(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,T]})$. Therefore, the loss incured by DPP for ω can be written as

$$\begin{aligned}
&OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, 0) - OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,T]}) \\
=&\sum_{t=1}^{T} [OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t)}) - OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t+1)})]
\end{aligned}$$

The revenue loss between HO and DPP can be decomposed into T increments. Each increment represents the gap between two "adjacent" OPTs. The gap can be uniformly bounded by l related to r_M. For any ω, $OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t)}) - OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t+1)}) \leq l$.

The expected revenue loss can be upper bounded:

$$\begin{aligned}
&E_{\omega}[OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, 0) - OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,T]})] \\
\leq& l \sum_{t=1}^{T} P(OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t)}) - OPT^{\omega}(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,t+1)}) > 0) \\
=& l \sum_{t=1}^{T} P(OPT^{\omega}(\boldsymbol{C}(t), d^{[t,T]}, 0) - OPT^{\omega}(\boldsymbol{C}(t), d^{[t,T]}, \gamma^{[t,t+1)}) > 0) \\
\leq& l \sum_{t=1}^{T} P(x^*_{i^t j^t}(t) < 1)
\end{aligned}$$

The first inequality results from $E[A] \leq lE[\mathbf{1}_{A>0}] = lP(A > 0)$.

The second equation is as follows. If $x^*_{i^t j^t}(t) \geq 1$, then $x^*(t)$ is still feasible for $OPT(\boldsymbol{C}(t), d^{[t,T]}, \gamma^{[t,t+1)})$. (Because the optimal policy)

Now we consider $P(x^*_{i^t j^t}(t) < 1)$. Let $x^{\text{DPP}}_{i^t j^t}(t)$ denote the number of type i^t assigned to knapsack j^t for DPP. At time period t, after realization of i^t, based on the maximum choice of j^t in DPP, we have

$$x^{\text{DPP}}_{i^t j^t}(t) = \max_j x^{\text{DPP}}_{i^t j}(t) \geqslant \frac{\sum_j x^{\text{DPP}}_{i^t j}(t)}{\sum_j 1} = \frac{\lambda_{i^t}^{[t,\theta T]}}{N}$$

The inequality holds because the maximum value over j is always greater than or equal to the average value.

$$\begin{aligned}
&\mathbb{P}\left(x^*_{i^t j^t}(t) < 1\right) \\
\leqslant &\mathbb{P}\left(x^*_{i^t j^t}(t) < x^{\mathrm{DPP}}_{i^t j^t}(t) + 1 - \frac{\lambda^{[t,\theta T]}_{i^t}}{N}\right) \\
= &\mathbb{P}\left(x^{\mathrm{DPP}}_{i^t j^t}(t) - x^*_{i^t j^t}(t) > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{N}\right) \\
\overset{\text{(a)}}{\leqslant} &\mathbb{P}\left(\max_{i'} \left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N}\right) \\
= &\mathbb{P}\left(\bigcup_{i'} \left\{\left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N}\right\}\right) \\
\leqslant &\sum_{i'} \mathbb{P}\left(\left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N}\right) \\
\overset{\text{(b)}}{=} &\sum_{i'} \sum_{i} \mathbb{P}\left(\left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N} \,\middle|\, (i^t = i)\right) \mathbb{P}\left(i^t = i\right) \\
\leqslant &\sum_{i} \sum_{i'} \mathbb{P}\left(\left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N} \,\middle|\, (i^t = i)\right) \mathbf{1}\left\{t \leqslant \theta T\right\} \\
\overset{\text{(c)}}{=} &\sum_{i} \sum_{i'} \mathbb{P}\left(\left|\mathbb{E}\left[d^{[t,\theta T]}_{i'}\right] - d^{[t,\theta T]}_{i'}\right| > \frac{\mathbb{E}\left[d^{[t,\theta T]}_{i^t}\right] - N}{\delta N}\right) \mathbf{1}\left\{t \leqslant \theta T\right\}.
\end{aligned}$$

(a) is obtained from the well-known Lipschitz property of optimal solutions to linear programs with respect to perturbations in the right-hand side, as established by Mangasarian and Shiau (1987).

For each t, consider $OPT(A^t, d, 0)$. In the DPP, $d = \lambda^{[t,\theta T]}$, while in the sample path HO, $d = d^{[t,\theta T]}$. We choose $x^*(t)$ be an optimal solution of $OPT(A^t, d^{[t,\theta T]}, 0)$ such that

$$\left\|x^*(t) - x^{\mathrm{DPP}}(t)\right\|_\infty \leqslant \delta \left\|d^{[t,\theta T]} - \lambda^{[t,\theta T]}\right\|_\infty = \delta \left\|d^{[t,\theta T]} - \mathbb{E}\left[d^{[t,\theta T]}\right]\right\|_\infty .$$

(b) follows from Bayes formula. (c) follows because of the arrival independence between different time periods.

We consider $\inf_{t,i:t\in[\theta T]} \frac{\lambda_i^{(t,\theta T]}(\theta)}{\theta T - t}$ is lower bounded by some positive constant $\lambda_{\min} \triangleq \inf_i \frac{\lambda_i^T}{T}$.

We have $\mathbb{E}\left[d_i^{(t,\theta T]}\right] = \lambda_i^{(t,\theta T]} \geqslant \lambda_{\min}\left(\theta T - t\right)$.

Let $T_0 = \left\lceil \frac{N}{\lambda_{\min}} \right\rceil < \frac{\theta}{2}$. For $t \leq \theta T - T_0$, $\frac{\mathbb{E}[d_i^{(t,\theta T]}] - N}{\delta N} \geqslant \frac{\lambda_{\min}(\theta T - t) - \lambda_{\min} T_0/2}{\delta N} \geqslant \frac{\lambda_{\min}(\theta T - t)}{2\delta N}$.

Thus,

$$
\begin{aligned}
&E[OPT(\boldsymbol{C}, d^{[1,T]}, 0) - OPT(\boldsymbol{C}, d^{[1,T]}, \gamma^{[1,T]})] \\
\leqslant & 2l \sum_{t=1}^{\theta T} \sum_{i} \sum_{i'} \mathbb{P}\left(\left| \mathbb{E}\left[d_{i'}^{(t,\theta T]} \right] - d_{i'}^{(t,\theta T]} \right| > \frac{\mathbb{E}\left[d_{i^t}^{[t,\theta T]} \right] - N}{\delta N} \right) \mathbf{1}\left\{ t \leqslant \theta T \right\} \\
\leqslant & 2l \sum_{i} \sum_{t=1}^{\theta T} \sum_{i'} \mathbb{P}\left(\left| \mathbb{E}\left[d_{i'}^{(t,\theta T]} \right] - d_{i'}^{(t,\theta T]} \right| > \frac{\lambda_{\min}(\theta T - t)}{2\delta N} \right) \\
\overset{(a)}{\leqslant} & 2l \sum_{i} \sum_{t=1}^{\theta T - T_0} \sum_{i'} 2\exp\left(-2 \frac{\left(\lambda_{\min}(\theta T - t)\right)^2}{(2\delta N)^2 (\theta T - t)} \right) + 2l \sum_{i} \sum_{t=\theta T - T_0 + 1}^{\theta T} \sum_{i'} 1 \\
= & 2l \sum_{i} \sum_{t=1}^{\theta T - T_0} \sum_{i'} 2\exp\left(-\frac{\lambda_{\min}^2}{2\delta^2 N^2} \cdot (\theta T - t) \right) + O(1) \\
\leqslant & 2lN \sum_{i} \sum_{t=1}^{\theta T - T_0} 2\exp\left(-\frac{\lambda_{\min}^2}{2\delta^2 N^2} \cdot (\theta T - t) \right) + O(1).
\end{aligned}
$$

(a) holds from Hoeffding's inequality

$$
\mathrm{P}\left(|S_n - \mathrm{E}[S_n]| \geq t \right) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n} (b_i - a_i)^2} \right)
$$

In this case, $S_n = d_{i'}^{(t,\theta T]}$, $a_i = 0, b_i = 1$.

Let $\theta \to \infty$, we can further upper bound by

$$
\begin{aligned}
& 2lN \sum_{i} \sum_{t=T_0}^{+\infty} 2\exp\left(-\frac{\lambda_{\min}^2 t^2}{2\delta^2 N^2 t} \right) + O(1) \\
\leqslant & 2lN^2 \sum_{t=T_0}^{+\infty} 2\exp\left(-\left(\frac{\lambda_{\min}^2}{2\delta^2 N^2} \right) \cdot t \right) + O(1) \\
= & O(1)
\end{aligned}
$$

■

6 Computational Experiments

6.1 Performances

$N = 10,\ L_j = 20 \forall j,$

$D_1 = [0.2, 0.5, 0.3]$

$w = [3, 4, 5],\ r = [4, 6, 8],\ r_i/w_i > 1 \forall i$

$w = [3, 4, 5] = r,\ r_i/w_i = 1 \forall i$

$w = [8, 6, 4]\ r = [5, 4, 3],\ r_i/w_i < 1 \forall i$

$D_2 = [0.25, 0.25, 0.25, 0.25]$

$w = [3, 5, 7, 9],\ r = [2, 4, 6, 8].$

Table 1: Performance of Policies

Distribution	T	r_i/w_i	Primal (%)	BPC (%)	BPP (%)
	60	>1	99.15	95.30	97.98
	80	>1	99.37	95.70	98.18
D_1	60	=1	99.68	95.66	98.88
	80	=1	99.80	96.26	99.44
	60	<1	98.56	95.86	97.66
	80	<1	97.00	95.67	97.31
	60	<1	99.65	94.63	99.84
	80	<1	99.81	96.89	99.91
D_2	100	<1	99.94	99.41	99.97

6.2 Loss

For each policy π, the loss of π, denoted as L_π, is defined as the average of $r^{\text{HO}}(\omega) - r^{\pi}(\omega)$ over all sample paths ω.

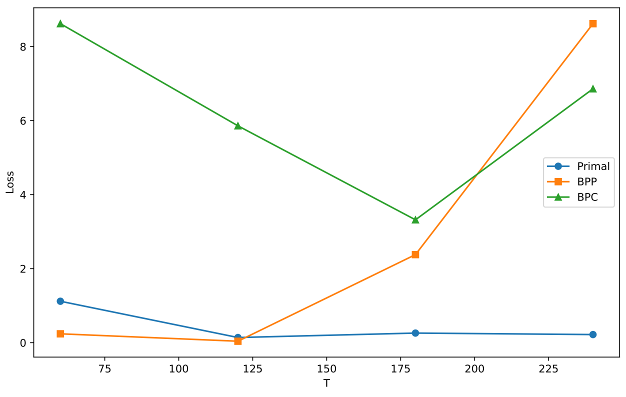

Figure 1: Loss over T

7 Conclusion

We study the dynamic stochastic multi-type multiple knapsack problem. We consider the bid-price control policy. Then we introduce the dynamic primal policy.

We conduct several numerical experiments to investigate various aspects of our approach. First, we compare different policies. Our proposed policy demonstrates superior and more consistent performance relative to these benchmarks.

References

Adelman, D., 2007. Dynamic bid prices in revenue management. Operations Research 55, 647–661.

Aydin, N., Birbil, S.I., 2018. Decomposition methods for dynamic room allocation in hotel revenue management. European Journal of Operational Research 271, 179–192.

Balseiro, S., Feldman, J., Mirrokni, V., Muthukrishnan, S., 2011. Yield optimization of display advertising with ad exchange, in: Proceedings of the 12th ACM conference on Electronic commerce, pp. 27–28.

Bertsimas, D., Popescu, I., 2003. Revenue management in a dynamic network environment. Transportation Science 37, 257–277.

Bienkowski, M., Pacut, M., Piecuch, K., 2020. An optimal algorithm for online multiple knapsack. arXiv preprint arXiv:2002.04543 .

Bitran, G.R., Mondschein, S.V., 1995. An application of yield management to the hotel industry considering multiple day stays. Operations Research 43, 427–443.

Chekuri, C., Khanna, S., 2005. A polynomial time approximation scheme for the multiple knapsack problem. SIAM Journal on Computing 35, 713–728.

Dantzig, G.B., 1957. Discrete-variable extremum problems. Operations Research 5, 266–288.

Ferreira, C.E., Martin, A., Weismantel, R., 1996. Solving multiple knapsack problems by cutting planes. SIAM Journal on Optimization 6, 858–877.

Gallego, G., van Ryzin, G., 1997. A multiproduct dynamic pricing problem and its applications to network yield management. Operations Research 45, 24–41.

Goldman, P., Freling, R., Pak, K., Piersma, N., 2002. Models and techniques for hotel revenue management using a rolling horizon. Journal of Revenue and Pricing Management 1, 207–219.

Khuri, S., Bäck, T., Heitkötter, J., 1994. The zero/one multiple knapsack problem and genetic algorithms, in: Proceedings of the 1994 ACM symposium on Applied computing, pp. 188–193.

Kleywegt, A.J., Papastavrou, J.D., 1998. The dynamic and stochastic knapsack problem. Operations Research 46, 17–35.

Kleywegt, A.J., Papastavrou, J.D., 2001. The dynamic and stochastic knapsack problem with random sized items. Operations Research 49, 26–41.

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

Mangasarian, O.L., Shiau, T.H., 1987. Lipschitz continuity of solutions of linear inequalities, programs and complementarity problems. SIAM Journal on Control and Optimization 25, 583–595.

Martello, S., Toth, P., 1990. Knapsack problems: algorithms and computer implementations. John Wiley & Sons, Inc.

Papastavrou, J.D., Rajagopalan, S., Kleywegt, A.J., 1996. The dynamic and stochastic knapsack problem with deadlines. Management Science 42, 1706–1718.

Pisinger, D., 1999. An exact algorithm for large multiple knapsack problems. European Journal of Operational Research 114, 528–541.

van Ryzin, G.J., Talluri, K.T., 2005. An introduction to revenue management, in: Emerging Theory, Methods, and Applications. INFORMS, pp. 142–194.

Sun, B., Zeynali, A., Li, T., Hajiesmaili, M., Wierman, A., Tsang, D.H., 2020. Competitive algorithms for the online multiple knapsack problem with application to electric vehicle charging. Proceedings of the ACM on Measurement and Analysis of Computing Systems 4, 1–32.

Talluri, K., van Ryzin, G., 1998. An analysis of bid-price controls for network revenue management. Management Science 44, 1577–1593.

Talluri, K.T., van Ryzin, G.J., 2006. The Theory and Practice of Revenue Management. Springer Science & Business Media.

Tönissen, D.D., Van den Akker, J., Hoogeveen, J., 2017. Column generation strategies and decomposition approaches for the two-stage stochastic multiple knapsack problem. Computers & Operations Research 83, 125–139.

Williamson, E.L., 1992. Airline network seat inventory control: Methodologies and revenue impacts. Ph.D. thesis. Massachusetts Institute of Technology.

Zhu, F., Liu, S., Wang, R., Wang, Z., 2023. Assign-to-seat: Dynamic capacity control for selling high-speed train tickets. Manufacturing & Service Operations Management 25, 921–938.

8 Proofs

Proof of Proposition 1

We model the problem as a special case of the multiple knapsack problem, then we consider the LP relaxation of this problem. In the model, groups are categorized into M distinct types. Each type i is characterized by a fixed size w_i, which serves as the weight, and an associated profit equal to r_i. For every type i, there are d_i items. Altogether, the total number of groups is given by $K = \sum_{i=1}^{M} d_i$. Each individual item k inherits its profit and weight from its type; specifically, if item k belongs to type i, then its profit p_k is r_i, and its weight W_k is w_i. To apply the greedy approach for the LP relaxation of (3), sort these items in non-increasing order of their profit-to-weight ratios: $\frac{p_1}{W_1} \geq \frac{p_2}{W_2} \geq \ldots \geq \frac{p_K}{W_K}$. The break item b is the smallest index such that the cumulative weight of item 1 to item b meets or exceeds the total capacity $\tilde{c}$: $b = \min\{j : \sum_{k=1}^{j} W_k \geq \tilde{c}\}$, where $\tilde{c} = \sum_{j=1}^{N} c_j$ is the total size of all knapsacks. For the LP relaxation of (3), the Dantzig upper bound (Dantzig, 1957) is given by $u_{\text{MKP}} = \sum_{j=1}^{b-1} p_j + \left(\tilde{c} - \sum_{j=1}^{b-1} W_j\right) \frac{p_b}{W_b}$. The corresponding optimal solution is to accept the whole groups from 1 to $b-1$ and fractional $(\tilde{c} - \sum_{j=1}^{b-1} W_j)$ item b. Suppose the item b belong to type $\tilde{i}$, then for $i < \tilde{i}$, $x_{ij}^* = 0$; for $i > \tilde{i}$, $x_{ij}^* = d_i$; for $i = \tilde{i}$, $\sum_{j=1}^{N} x_{ij}^* = (\tilde{c} - \sum_{i=\tilde{i}+1}^{M} d_i w_i)/w_{\tilde{i}}$. ■

Proof of Lemma 1

According to the Proposition 1, the aggregate optimal solution to LP relaxation of problem (3) takes the form $xe_{\tilde{i}} + \sum_{i=\tilde{i}+1}^{M} d_i e_i$, then according to the complementary slackness property, we know that $z_1 = \ldots = z_{\tilde{i}} = 0$. This implies that $\beta_j \geq \frac{r_i}{w_i}$ for $i = 1, \ldots, \tilde{i}$. Since $\frac{r_i}{w_i}$ increases with i, we have $\beta_j \geq \frac{r_{\tilde{i}}}{w_{\tilde{i}}}$. Consequently, we obtain $z_i \geq r_i - w_i \frac{r_{\tilde{i}}}{w_{\tilde{i}}} = \frac{r_i w_{\tilde{i}} - r_{\tilde{i}} w_i}{w_{\tilde{i}}}$.

Given that $\mathbf{d}$ and $\mathbf{L}$ are both no less than zero, the minimum value will be attained when $\beta_j = \frac{r_{\tilde{i}}}{w_{\tilde{i}}}$ for all j, and $z_i = \frac{r_i w_{\tilde{i}} - r_{\tilde{i}} w_i}{w_{\tilde{i}}}$ for $i = \tilde{i} + 1, \ldots, M$. ■

Proof of Lemma 2

We consider two cases based on whether the set $\mathcal{J}_i$ is empty or not.

Case 1: $\mathcal{J}_i = \varnothing$ If $\mathcal{J}_i = \varnothing$, then for all $j \in \mathcal{N}$, we have $r_i - \beta_{ij}^{\dagger *} \leq 0$. This implies that the constraint: $\alpha_i \geq r_i - \beta_{ij}^{\dagger *}$ is automatically satisfied for all $j \in \mathcal{N}$ when $\alpha_i \geq 0$. Therefore, these constraints are redundant and can be removed without affecting the solution.

Case 2: $\mathcal{J}_i \neq \varnothing$ We prove by contradiction that there exists $h_i^* \geq 1$ for some $j' \in \mathcal{J}_i$.

Assumption for contradiction: Suppose that in the optimal solution, for all $j' \in \mathcal{J}_i$, we have $h_i^* = 0$.

Since $\mathcal{J}_i \neq \varnothing$, there exists at least one $j' \in \mathcal{J}_i$ such that $r_i > \beta_{ij'}^{\dagger *}$. From the constraint $\alpha_i \geq r_i - \beta_{ij'}^{\dagger *}$ and the optimality conditions, we must have $\alpha_i = r_i - \beta_{ij'}^{\dagger *} > 0$. Now consider the value:

$$\gamma_{j'} = \max_{\boldsymbol{h} \in S(c_{j'})} \sum_i \beta_{ij'}^{\dagger *} h_i$$

Under the contradiction assumption ($h_i^* = 0$ for all $j' \in \mathcal{J}_i$), we have: $\gamma_{j'} = \sum_i \beta_{ij'}^{\dagger *} h_i^*$.

Now examine the objective function:

$$\sum_i \alpha_i d_i + \sum_j \gamma_j.$$

Consider perturbing $\beta^{\dagger *}_{ij'}$ by increasing it slightly to $\beta^{\dagger *}_{ij'} + \delta$ (for some small $\delta > 0$ such that $r_i > \beta^{\dagger *}_{ij'} + \delta$ still holds). Then:

- Since α_i is exactly at the boundary ($\alpha_i = r_i - \beta^{\dagger *}_{ij'}$), we can now set $\alpha_i^{\text{new}} = r_i - (\beta^{\dagger *}_{ij'} + \delta) < \alpha_i$.
- The term $\gamma_{j'}$ remains unchanged because $h_i^* = 0$ for $j' \in \mathcal{J}_i$ (by our contradiction assumption), and the perturbation does not affect other terms.
- Since $d_i > 0$ (positive arrival rate for type i), the objective decreases by $\delta \cdot d_i > 0$.

This contradicts the optimality of the current solution. Therefore, our initial assumption must be false, and there must exist some $j' \in \mathcal{J} i$ such that $hi^* \geq 1$. ■

Proof of Proposition 2

■

Proof of Proposition 3

If $x^*_{ij} > 0$, then $\sum_{\boldsymbol{h} \in S(c_j)} h_i y_{j\boldsymbol{h}} > 0$, then there exists $\boldsymbol{h}$ such that $y_{j\boldsymbol{h}} > 0$ and $h_i \geq 1$. Then $\boldsymbol{h} \in \arg\max(\sum_i (r_i - \alpha_i) h_i - \gamma_j)$.

According to the complementary slackness property, $\alpha_i + \beta_{ij} = r_i$, then $\max \sum_i (r_i - \alpha_i) h_i - \gamma_j$ equals $\max \sum_i \beta_{ij} h_i$. Thus, there exists j such that $\boldsymbol{h}^* \in \arg\max_{\boldsymbol{h} \in S(c_j)} \sum_i \beta^{\dagger}_{ij} h_i, h_i^* \geq 1$.

■

Lemma 5. *$z(DLP) \geq V^{HO}$ results from the concave property.*

Proof of Lemma 5

Consider the standard linear program: $\phi(\boldsymbol{d}) = \{\max c^T \boldsymbol{x} : A\boldsymbol{x} \leq \boldsymbol{d}, \boldsymbol{x} \geq \boldsymbol{0}\}$. Suppose that $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are two demand vectors, the optimal solution is $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. For any $\lambda \in [0, 1]$, $\boldsymbol{d}_\lambda = \lambda \boldsymbol{d}_1 + (1 - \lambda)\boldsymbol{d}_2$. Let $\boldsymbol{x}_\lambda = \lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2$, then $A\boldsymbol{x}_\lambda = A(\lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2) \leq \lambda \boldsymbol{d}_1 + (1 - \lambda)\boldsymbol{d}_2 = \boldsymbol{d}_\lambda$. Thus, $\boldsymbol{x}_\lambda$ is a feasible solution for $\boldsymbol{d}_\lambda$. Then, $\phi(\boldsymbol{d}_\lambda) \geq \boldsymbol{c}^T \boldsymbol{x}_\lambda = \lambda \boldsymbol{c}^T \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{c}^T \boldsymbol{x}_2 = \lambda \phi(\boldsymbol{d}_1) + (1 - \lambda)\phi(\boldsymbol{d}_2)$, which indicates $\phi(\boldsymbol{d})$ is concave. Let $\phi(\boldsymbol{d})$ indicate the optimal value of the linear relaxation of the SPDR problem. Substitute $\boldsymbol{x}$ with $y_{j\boldsymbol{h}}$ and view $y_{j\boldsymbol{h}}$ as the decision variables, then the concave property still holds for (11). $V^{\text{HO}} = E[\phi(\boldsymbol{d})] \leq \phi(E[\boldsymbol{d}]) = z(\text{DLP})$. ■