European Journal of Operational Research 263 (2017) 768–781

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Managing limited retail space for basic products: Space sharing vs. space dedication

Wei Zhang[a,*], Kumar Rajaram[b]

[a] Faculty of Business and Economics, University of Hong Kong, Pokfulam Road, Hong Kong
[b] Anderson School of Management, University of California, Los Angeles, 110 Westwood Plaza, Los Angeles, CA 90095, USA

ARTICLE INFO

Article history:
Received 1 November 2016
Accepted 24 May 2017
Available online 30 May 2017

Keywords:
Retailing
Basic products
Inventory strategy
Shelf-space management
Scheduling

ABSTRACT

In this paper, we study the problem of managing limited retail shelf or storage space for basic products by considering two inventory management strategies: space dedication and space sharing. When space is dedicated to each product, there is more flexibility in planning as different products can be replenished independently. In contrast, when space is shared across different products, there is potential for saving space; however, replenishment has to be coordinated across products and this leads to additional costs due to the lack of flexibility in replenishing each product individually. We model this problem as a non-linear mixed integer program and develop an effective heuristic and an upper bound for each strategy. We introduce three different but consistent criteria to compare each strategy. Through an extensive computational study, we identify the most relevant factors that impact the relative benefit of space sharing over space dedication. In addition, we show that space sharing with an optimal replenishment scheduling program can on average reduce space consumption by 31%.

1. Introduction

Retailers usually carry a large assortment of products, and they face a even greater set of potential choices for their assortments. According to the Food Marketing Institute, the average number of items carried in a supermarket in 2015 is 39,500.[1] On the other hand, retailers are usually constrained by limited shelf or storage space. For retailers that carry basic, long life-cycle products, limited space can lead to either a restricted assortment or a large enough assortment with low inventory levels and frequent replenishment. Both these situations can limit revenues. Hence, retailers need to manage their space by carefully making two types of decisions. The first is determining the optimal assortment, given that a limited number of products can be carried. The second relates to managing inventory levels and replenishment schedules in order to utilize the space effectively. The assortment and inventory management problems are closely related because they generate inputs of decision making for each other. Assortment management determines the optimal product offerings and the demand rate for each product; with the demand information, inventory management chooses the economic ordering quantities and the optimal replenishment schedule, determining the economic cost of offering each product, which in turn is used to choose the product assortment. Therefore, it is necessary to consider these two sets of decisions simultaneously in making the best space management decisions.

The retail space management problem is faced by several urban and suburban retailers such as Walgreens, CVS, City Target, etc, by food chains such as Whole Foods, Safeway, Ralphs, etc, and by other retail chains such as Office Depot, Staples, Guitar Center, etc. Due to the product strategies adopted by these retailers, they mostly sell large assortments of basic, long life-cycle products with stable, predictable demand (20). Their locations often have high property rent and thus limited shelf and storage space. Moreover, these retailers have stores at different locations and they may have different assortments that vary over time, thus making the retail space management problem a recurring and complicated task.

Retailers adopt two different strategies in allocating limited space to different products. The first of these is to dedicate space to each product, and the second is to allow products to share the space (4). For example, retailers may allocate one or multiple fixed columns of the shelf to each shampoo and hand soap (as shown on the left side of Fig. 1).[2] We may also observe that different

* Corresponding author.
E-mail addresses: zhangw.03@gmail.com, wzhang15@hku.hk (W. Zhang), kumar.rajaram@anderson.ucla.edu (K. Rajaram).

[1] Source: https://www.fmi.org/our-research/supermarket-facts, accessed in May, 2017.

[2] Examples of space sharing abound in practice. For example, in grocery stores like Ralphs, the locations of price tags for many products are adjusted based on where products are placed and inventory level. Target stores sell basic garments

http://dx.doi.org/10.1016/j.ejor.2017.05.045
0377-2217/© 2017 Elsevier B.V. All rights reserved.

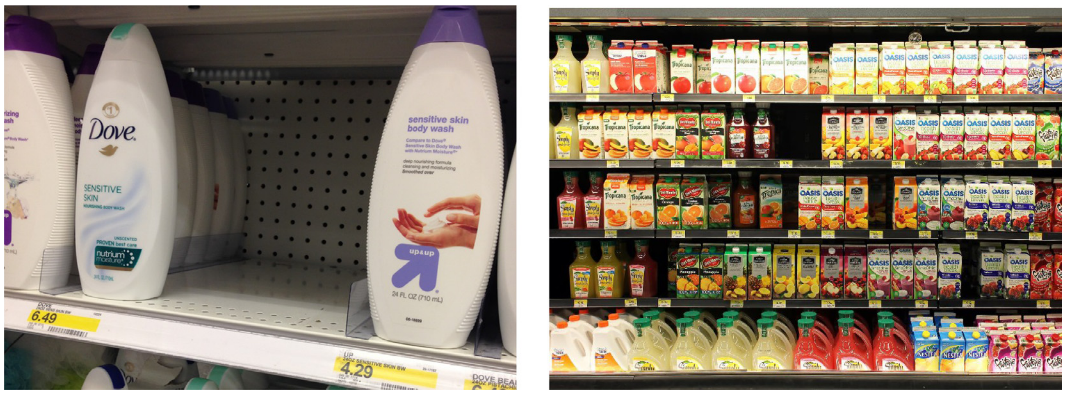

Fig. 1. Examples of space dedication (left) and sharing (right).

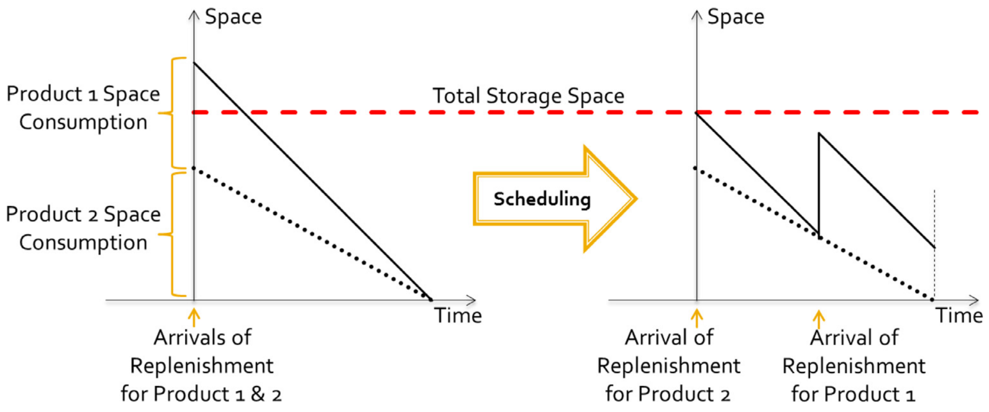
Space
Product 1 Space Consumption
Product 2 Space Consumption
Total Storage Space
Scheduling
Time
Arrivals of Replenishment for Product 1 & 2
Space
Time
Arrival of Replenishment for Product 2
Arrival of Replenishment for Product 1

Fig. 2. Space consumption and replenishment scheduling for two products.

juices share the shelf space with location-flexible price tags and thus, the space allocated to a product depends on the inventory level (as shown on the right side of Fig. 1). Although space dedication may be required in practice due to several reasons (for example, shelves are sometimes designed to hold bottles of particular shapes and suppliers may contract with retailers for dedicated shelf space), in many other situations it is not apparent which strategy should be adopted given the following trade-off exists between these two strategies. When space is dedicated to each product, inventory management is easier because different products can be replenished independently. However, a larger amount of shelf space may be required. In contrast, when space is shared across different products, space requirement can be potentially reduced. But, replenishment has to be coordinated at the cost of flexibility and optimality for individual products. It is important to note that if a schedule is not coordinated, there may not be sufficient space when the replenishment arrives. To illustrate, consider two products, X and Y, for which each unit consumes one unit of space and the demand rates are 1 unit per day for both products. For simplicity, assume that the buffer space required for both products are zero. A total space of size 10 can sustain a replenishment cycle of 6 days for both products: The first product is replenished at day 1 and the second is at day 4, as shown on the right side of Fig. 2. However, a space of size 10 cannot sustain a 6-day cycle for product X and a 5-day cycle for product Y, because the replenishment dates will coincide and space constraint of 10 units will be violated (as shown on the left side of Fig. 2). This simple example shows that space sharing can increase space utilization and allow the same space to accommodate larger assortments with properly scheduled replenishment. In addition, space sharing can also reduce space requirements and their associated retail rents (per product).

This paper compares space dedication with space sharing and identifies under what conditions, which strategy would be more preferable to retailers. To do this, we develop an optimization model to jointly determine the optimal assortment of products, their inventory levels and replenishment schedules. Here, we embed an assortment planning problem and a replenishment scheduling problem into a multi-product EOQ model.[3] The resulting joint optimization problem is a non-linear mixed integer program, and we develop effective methods to solve this problem. We use these methods to conduct an extensive computational study that compares space sharing with space dedication and draw managerial insights.

This paper is organized as follows. We provide a brief literature review in Section 2. In Section 3, we develop models for the retail space management problem under different conditions specifying the shelf space constraint and type of replenishment. We analyze and solve these formulations with solution methods that correspond to a space sharing and a space dedication strategies used by retailers in practice. Specifically, we solve the space sharing strategy in Section 4 and similarly solve the space dedication strategy in Section 5. We present results from our numerical study

such as T-shirts, shirts, socks, etc., with different brands sharing the same rack. In addition, the storage space in the retail backroom is normally shared across products.

[3] While stochastic approaches such as the Newsboy model are used for fashion products that have high demand uncertainties and relatively short life-cycles (e.g., Rajaram [21] and 2), deterministic approaches such as the EOQ model are used for basic products that have low demand uncertainties and relatively long life-cycles (14).

in Section 6 and summarize our work in Section 7. All proofs are provided in the Appendix.

2. Literature review

The problem studied in this paper is related to several streams of literature: the classic joint replenishment problem, the assortment planning problem, the space-constrained inventory management problem, and the joint problem of assortment and inventory management under space constraints.

The joint replenishment problem (JRP) studies a retailer's problem of when to place orders and how to combine orders of different products. Given that we only consider managing basic products in this paper, we focus on deterministic demand. There is a large body of literature on the JRP with deterministic demand: e.g., Jackson, Maxwell, and Muckstadt [16], Anily and Federgruen [1], Federgruen and Zheng [11], and Viswanathan and Mathur [25]. Similar to the traditional JRP literature, these studies do not consider the shelf-space cost or constraint (7). Cachon [7] studies the management of transportation, shelf space, and inventory costs for a retailer that sells multiple products with stochastic demand. But this paper does not consider the assortment problem as well as demand substitution among different products. Khouja and Goyal [17] conduct a review of the JRP literature from 1989 to 2005 and they find that recent research on the JRP has focused on finding faster algorithms to the classic JRP rather than on improving the solution quality. Our work focuses on improving the solution quality by incorporating the assortment decision because the JRP and assortment management are interrelated with space constraints and demand substitution.

The assortment planning problem has been extensively studied. Kök, Fisher, and Vaidyanathan [19] provides a comprehensive review. A large body of this literature studies the assorment planning for fashion products. Papers in this stream either employ consumer choice models (e.g., 22; 12; 15), or used exogenous demand models (e.g., 23; Rajaram [21]; 8; Bernstein et al. [2]). There has also been considerable research on assortment planning for basic products (e.g., 5 and 6; see a brief review in Rajaram [21]). However, all these papers, on both fashion and basic products, do not consider *joint* optimization of assortment planning and inventory management *under storage space constraints*. The literature for space-constrained inventory models for multi-items is based on Hadley and Whitin [14]. However, work in this area (e.g., 13; 26) does not consider replenishment scheduling and assortment planning. In contrast, this paper develops a model that jointly optimizes the selection of products, their inventory levels, and replenishment schedules to increase the effective utilization of limited storage or shelf space that is shared among products.

The joint assortment and inventory management under space constraints has also been studied. Early work is represented by Corstjens and Doyle [9], Borin, Farris, and Freeland [3], and Urban [24]. Recently, Kök and Fisher [18] study the problem with shelf-space constraints, one level substitution and a multi-nomial logit model. They develop a procedure for estimating parameters of substitution behavior and demand for products. They solve the problem using an iterative optimization heuristic. Application of their methods to a large super market chain suggests a 50% increase in profits. However, all these papers implicitly assume space dedication and thus do not consider the replenishment scheduling problem. In our work, we explicitly consider replenishment scheduling, thus enabling us to compare space dedication and space sharing, and provide insights on how to make the choice between these two inventory strategies.

To summarize, our paper makes the following contributions. First, we complement the research stream of the joint assortment and inventory management problem for basic products by considering replenishment scheduling and space sharing. Second, we propose an efficient upper bound as well as effective heuristics to solve this complicated problem. Third, we compare two commonly used inventory management strategies: space sharing and space dedication, and show which choice would be better under which circumstance.

3. Model formulation

We formulate the retail space management problem as a nonlinear mixed integer program. Let $i \in \mathcal{P} = \{1, 2, \dots, M\}$ index the set of potential products and $n \in \mathcal{G} = \{1, 2, \dots, N\}$ index the set of product groups. A product group can be defined as the set of products of the same brand or alternatively specified as the set of products offered by the same distributor. Hence, a product can only be from one group. We make the following assumptions.

First, we assume that a fixed level of safety stock can be used to cover the low levels of demand uncertainty that may occur for basic products during the replenishment cycle time. Given the cycle time and the demand variability, the safety stock for that period can be calculated for a certain service level. We consider a service level that is close to 100%. Since we want to control the replenishment cycle time T_i for each product, the required safety stock level for product i is $\phi_i = \theta_i \cdot T_i \cdot s_i$, where $\theta_i := z_{SL} \cdot CV_i$ with z_{SL} being the z-value corresponding to the service level, and CV_i is the (time-invariant) coefficient of variation for the demand of product i. Further, as explained below, s_i is the effective demand rate of product i.

Second, considering that inventory level is normally high and stock-outs are rare for basic products, we focus on assortment-based product substitution and assume that stock-out-based substitution can be ignored.

Third, we use an exogenous demand model, which is the most commonly used demand model in the literature on inventory management for substitutable products (Kök, Fisher, and Vaidyanathan [19]). In particular, each product in the set $\mathcal{P}$ has an original demand; if a product is not offered, a fraction of the original demand for this product will be transferred to other products that are included in the assortment. For the sake of tractability, we assume that the fraction of demand that would be substituted by any other particular product is fixed and independent of the assortment. Despite of this limitation, exogenous demand models have their unique strength compared with other demand models such as the Multinomial Logit Model. For example, exogenous demand models can use a substitution matrix to capture many different substitution patterns. The interested reader is referred to Kök, Fisher, and Vaidyanathan [19] for a detailed discussion.

Next, we define the following parameters and variables.

Parameters:

d_i: the original (or default) demand rate of product i.

w_{ij}: the fraction of the demand for product j that will be transferred to product i given that product $j \neq i$ is not offered; $w_{ii} = 1$.

v_i: the profit margin of product i.

h_i: the unit holding cost rate of product i.

K_n: the replenishment setup cost each order for product group n, which is independent of the number of products included in the order.

κ_i: the replenishment setup cost of product i.

θ_i: the safety stock coefficient of product i.

Decision variables:

y_i: 0–1 variable that equals 1 if product i is included in the assortment, and 0 otherwise.

Q_i: the batch order quantity of product i.

Auxiliary variables:

x_{ij}: demand diverted from product j to product i.

s_i: the total effective demand rate of product i; $s_i = y_i\left[d_i + \sum_{j\neq i} w_{ij} d_j \left(1-y_j\right)\right]$.

We assume that the profit margins are high enough so that $v_i Q_i > \kappa_i$ if $y_i = 1$; otherwise, product i will never be included in the assortment. The total effective demand rate for product i can be rewritten as $s_i = \sum_j w_{ij} d_j x_{ij}$, where $x_{ij} = y_i\left(1-y_j\right)$ for $j \neq i$ and $x_{ij} = y_i$ for $j = i$. This form will be used in the models described below. Note that we can let x_{ij} be a continuous variable to simplify the specification of the model, but it will still be integral once the constraints are imposed. Let $y := (y_1, \ldots, y_M)'$, $Q := (Q_1, \ldots, Q_M)'$, and $s := (s_1, \ldots, s_M)'$ be the vectors of assortment decisions, order quantities, and effective demand, respectively; let $\mathbf{x} := \{x_{ij}\}$ be a matrix.

We consider two types of replenishment mechanisms that occur in practice. The first is *independent* replenishments in which there are no group-specific replenishment setup costs for ordering a product from a group. This occurs when the retailers use different distributors for their products and the associated transaction costs are subsumed in the product replenishment setup costs κ_i for $\forall i$. In the second type, which we refer to as *combined* replenishments, there is an additional fixed replenishment setup cost K_n incurred by the retailer when any product in group n is ordered. This could include the fixed cost of a truck used for delivery for a distributor who supplies a particular product group. Thus, the retailer can benefit by combining multiple products of the same group in an order. However, incorporating the decision on which subset of products to combine in an order significantly complicates the analysis. Therefore, in the following, we investigate this problem in three steps. First, we discuss the uncapacitated problem, which has no storage space constraints, with independent replenishments. Next, we study the capacitated problem with storage space constraints and independent replenishments. Finally, we tackle the full version: the capacitated problem with combined replenishments.

3.1. Uncapacitated problem with independent replenishments

The objective of the uncapacitated problem is to maximize the total profit averaged over time *without* storage space constraints. With independent replenishments (i.e., $K_n = 0$ for $\forall\ n \in \mathcal{G}$), this can be formulated as:

$$(UP)\quad \max_{y,Q,s,x} \sum_{i\in\mathcal{P}} \left[v_i s_i - h_i\left(\frac{Q_i}{2} + \theta_i \cdot Q_i\right) - \frac{\kappa_i s_i}{Q_i}\right] \tag{1}$$

$$s.t.\ s_i = \sum_j w_{ij} d_j x_{ij} \quad \forall i, \tag{2}$$

$$x_{ij} \le y_i \quad \forall i, j, \tag{3}$$

$$x_{ij} \le 1 - y_j \quad \forall i \neq j, \tag{4}$$

$$y_i \in \{0, 1\} \quad \forall i, \tag{5}$$

$$x_{ij} \ge 0 \quad \forall i, j, \tag{6}$$

$$Q_i \ge 0, \quad \forall i. \tag{7}$$

Objective (1) equals the sum of the average gross profit net of the average holding cost and the average replenishment setup cost for each product. To simplify further, we write $H_i := h_i \cdot \left(\frac{1}{2} + \theta_i\right)$ so that the second term in the objective becomes $H_i Q_i$. Constraint (2) defines the effective demand rate of a product as the sum of substituted demand streams from all products. Constraints (3) and (4) enforce the condition that demand is diverted from j to i only if product i is stocked and j is not stocked. Note that (3) and (4) are derived by first representing $x_{ij} = y_i$ for $j = i$ and $x_{ij} = y_i\left(1-y_j\right)$ for $j \neq i$ by two inequality constraints for each and then dropping constraints, $x_{ij} \ge y_i$ for $j = i$ and $x_{ij} \ge y_i \cdot \left(1-y_j\right)$ for $j \neq i$. Given *UP* is a maximization problem and (by assumption) the profit margin is high enough so that $v_i > \frac{\kappa_i}{Q_i}$, it is always optimal to increase s_i if we ignore the constraints; thus, given that $w_{ij} d_j > 0$, it is always optimal to increase x_{ij} without constraints. Hence, $x_{ij} \le y_i \cdot \left(1-y_j\right)$ for $j \neq i$ and $x_{ij} \le y_i$ for $j = i$ will be binding and be equivalent to the equality constraints, so that $x_{ij} \ge y_i \cdot \left(1-y_j\right)$ for $j \neq i$ and $x_{ij} \ge y_i$ for $j = i$ can be dropped. Finally, we write $x_{ij} \le y_i$ and $x_{ij} \le 1 - y_j$ for $j \neq i$ to jointly represent $x_{ij} \le y_i \cdot \left(1-y_j\right)$ for $j \neq i$. Constraint (5)-(7) enforce the range of the variables. Observe that *UP* is a mixed-integer, non-linear optimization problem. For any assortment choice and consequently $s \in \mathbb{R}_+^M$, it is easy to solve for $Q \in \mathbb{R}_+^M$, since it reduces to the EOQ problem with $Q_i^* = \sqrt{\frac{\kappa_i s_i}{H_i}}$. The following lemma establishes the convexity of the objective function given $Q_i^* = \sqrt{\frac{\kappa_i s_i}{H_i}}$ and simplifies the optimization over s.

Lemma 1. *The objective function (1) is convex in s given $Q_i = \sqrt{\frac{\kappa_i s_i}{H_i}}$ for $\forall i$.*

We use Lemma 1 to establish proposition 1, which simplifies the computation of *UP*.

Proposition 1. *The integrality constraints in the uncapacitated problem can be dropped without affecting the optimal solution.*

In light of Proposition 1, we replace (5) with

$$y_i \in [0, 1] \quad \forall i. \tag{8}$$

Thus, the *UP* is a simpler non-linear optimization problem which is amenable to solution using standard methods. This is useful for the heuristics used to solve the capacitated problem described next. Furthermore, Lemma 1 and Proposition 1 will be used to simplify the solution methods for the space sharing and space dedication strategies in Sections 4 and 5, respectively.

3.2. Capacitated problem with independent replenishments

In this section, we add in the constraint of limited storage or shelf space for inventory. If space is not dedicated but shared by all the products in the assortment, replenishment schedules of different products should be coordinated to ensure that there is sufficient space when the replenishment arrives. We will analyze all the formulations considered in this section under the space sharing and space dedication strategies in Sections 4 and 5, respectively.

For the space sharing strategy, we use a common replenishment cycle in which all the products are replenished exactly once in a determined sequence during the cycle. A common cycle approach is often used in practice because of the ease of implementation, especially for products of the same category (e.g., sodas, cereals, soaps, ballpens, etc). Later, we will allow different replenishment cycles for the space dedication strategy. The following additional notations are introduced.

Model parameters:

C: total available space;

c_i: space consumption for a unit of product i.

Decision variables:

T: the common replenishment cycle time;

τ_i: the order arrival time of product i in a cycle;

t_{ij}: the time between the replenishment of product i and product j; define $t_{ij} = \left(\tau_j - \tau_i\right) \cdot \mathbb{I}\left\{\tau_i \le \tau_j\right\} + \left(T + \tau_j - \tau_i\right) \cdot \mathbb{I}\left\{\tau_i > \tau_j\right\}$ if $i < j$ and $t_{ij} = \left(\tau_j - \tau_i\right) \cdot \mathbb{I}\left\{\tau_i < \tau_j\right\} + \left(T + \tau_j - \tau_i\right) \cdot \mathbb{I}\left\{\tau_i \ge \tau_j\right\}$ if $i \ge j$, where $\mathbb{I}$ is the indicator function; define matrix $\mathbf{t} := \{t_{ij}\}$. This is illustrated by Fig. 3.

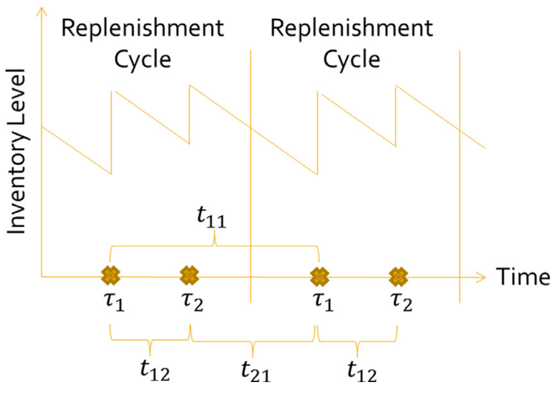

Fig. 3. Definition of t_{ij} in a two-product example.

The replenishment schedule is then determined by $\tau := \{\tau_i : i \in \mathcal{P}\}$. For the capacitated problem, the space constraint must be satisfied. Since we consider basic products that typically exhibit low demand variability, we assume the fluctuations in demand and consequently space consumption can be covered by a fixed buffer space, which can be calculated using the approach outlined previously. Hence, in the following analysis, we focus on the expected values. Let $I_i(t)$ denote the average inventory level of product i at time t. Then, we need to ensure the following space constraint is satisfied:

$$\sum_i c_i I_i(t) \le C, \quad \forall t \in [0, T]. \tag{9}$$

In principle, we can discretize time t and define a time unit as a day or an hour. However, if the required time unit is small in comparison to T, this would lead to a large number of constraints, resulting in a complicated integer program. To simplify the analysis, note that the highest inventory level occurs at the delivery time, i.e. at τ_i for some $i \in \mathcal{P}$. Hence, we only need to check the space constraint at M time spots, where replenishments occur. The following proposition uses this idea and simplifies (9).

Proposition 2. *The space constraint (9) is equivalent to*

$$\sum_j c_j s_j \left(t_{ij} + T \cdot \theta_j\right) \le C, \quad \forall i \in \mathcal{P}. \tag{10}$$

For the feasibility of $\mathbf{t}$, we know from the definition (as illustrated by Fig. 3) that

$$t_{ij} + t_{ji} = T, \quad \forall i \ne j. \tag{11}$$

However, $t_{ij} + t_{ji} = T$ determines the relative location of τ_i and τ_j in the cycle without considering the delivery time of other products. In order to guarantee the uniqueness of τ_i for $\forall i$, we introduce a "triangle" relationship: for $\forall i \ne j \ne k$, once t_{ij} and t_{ik} are fixed and τ_i is known, the relative location of τ_j and τ_k should be determined. If $\tau_j < \tau_k$, we have $t_{ij} - t_{ik} = -t_{jk}$; otherwise, we have $t_{ij} - t_{ik} = t_{kj}$. Therefore, $t_{ij} - t_{ik}$ equals *either* $-t_{jk}$ *or* t_{kj}, depending on which product is replenished first. Note that once $\mathbf{t}$ is determined in this way, the set τ can be determined given any τ_i. However, the specific value of τ is not important, as we focus on the long-term average profit. Thus, it is enough to determine $\mathbf{t}$.

To simplify these either-or constraints that maintain the "triangle" relationship, we introduce indicator binary variable $z_{ij} \in \{0, 1\}$ for $\forall i, j$ and let matrix $\mathbf{z} := \{z_{ij}\}$. Then, the either-or relashionship can be written as

$$t_{ij} - t_{ik} = z_{kj} t_{kj} - z_{jk} t_{jk}, \quad \forall i \ne j \ne k \in \mathcal{P}, \tag{12}$$

$$z_{kj} + z_{jk} = 1, \quad \forall j \ne k \in \mathcal{P}. \tag{13}$$

The capacitated problem with independent replenishments can now be formulated as

$$(CPI) \qquad \max \sum_{i \in \mathcal{P}} \left(v_i s_i - H_i Q_i - \frac{\kappa_i y_i}{T} \right)$$
$$s.t. \ (2)-(6), \ (10)-(13),$$
$$Q_i = s_i T \quad \forall i, \tag{14}$$

$$z_{ij} \in \{0, 1\} \quad \forall i, j, \tag{15}$$

$$t_{ij} \ge 0, t_{ii} = T \quad \forall i, j. \tag{16}$$

Constraint (14) ensures that demand in a cycle ($s_i T$) equals replenishment quantity (Q_i), while (15) and (16) enforce the variable range. The *CPI* is a mixed integer program with a non-linear objective and quadratic constraints.

Theorem 1. *The CPI is NP-hard.*

Theorem 1 implies that we may not be able to solve the *CPI* to optimality for large sized real problems. We verify the complexity in the computational study. In Section 4, we will analyze *CPI* first and the results will be useful to analyze the capacitated problem with combined replenishments considered next.

3.3. Capacitated problem with combined replenishments

We consider the case when a setup cost $K_n \ge 0$ is incurred for a replenishment order of group $n \in \mathcal{G}$, no matter how many products of group n are included in the same order. In this case, we should decide which products to combine in a replenishment order. Let $\mathcal{P}_n$ index the set of products in group n, and $n(i)$ index the group that contains product i; i.e., $i \in \mathcal{P}_{n(i)}$. We further introduce the following decision variables.

Decision variables:

o_j: 0–1 variable that equals 1 if the jth replenishment order is placed, and 0 otherwise. Note that o_j originally corresponds to product j. There are M products in total, so we can place at most M orders in a replenishment cycle.

r_{ij}: 0–1 variable that equals 1 if product i is included in the jth order, and 0 otherwise; Because it is useful to combine only products of the same group, we require that $r_{ij} = 0$ for any $j \notin \mathcal{P}_{n(i)}$.

The variables should satisfy the following constraints. First, any product that is included in the assortment has to be included in one and at most one replenishment order. Thus, we need

$$y_i \le \sum_{j \in \mathcal{P}_{n(i)}} r_{ij} \quad \forall i \in \mathcal{P}. \tag{17}$$

Futhermore, if any products are included in an order, the order has to be placed. Thus, we have

$$o_j \ge r_{ij} \quad \forall i, j \in \mathcal{P}. \tag{18}$$

Lastly, if any two products are combined for replenishment in the same order, their replenishments are made at the same time. Hence,

$$t_{i'i''} \le T \cdot \left(2 - r_{i'j} - r_{i''j}\right) \quad \forall i', i'' \in \mathcal{P}_{n(j)}, i' < i'', \forall j \in \mathcal{P}. \tag{19}$$

The full version of the retail space management problem can be formulated as

$$(CPC) \qquad \max \sum_{i \in \mathcal{P}} \left(v_i s_i - H_i s_i T - \frac{\kappa_i y_i}{T} - \frac{K_{n(i)} o_i}{T} \right)$$
$$s.t. \ (2)-(6), \ (10)-(13), \ (15)-(19)$$
$$o_i \in \{0, 1\} \quad \forall i, \tag{20}$$

$$r_{ij} \in \left\{0, \mathbb{I}\left\{j \in \mathcal{P}_{n(i)}\right\}\right\} \quad \forall i, j. \tag{21}$$

Given the additional binary variables and constraints, the *CPC* can be potentially more difficult to solve than the *CPI*. In Section 4.2, we decompose the problem and develop structural properties to facilitate the development of an effective solution procedure.

4. The space-sharing strategy

In this section, to analyze and solve the capacitated problem with the space-sharing strategy, we focus on independent replenishments first and then consider combined replenishments. Due to the computational difficulty of the problems, we focus on heuristics that provide good feasible solutions. An upper bound will then be developed to gauge the effectiveness of these heuristics.

4.1. Managing independent replenishments

Let $\upsilon := (\upsilon_1, \ldots, \upsilon_M)'$, $H := (H_1, \ldots, H_M)'$, $\kappa := (\kappa_1, \ldots, \kappa_M)'$, $K := (K_{n(1)}, \ldots, K_{n(M)})'$, $o := (o_1, \ldots, o_M)'$, and $\mathbf{r} := \{r_{ij}\}$. Observe that there are two subproblems embedded in the *CPI*. One is an assortment planning problem that decides $\{y, x, s\}$, and the other is a replenishment scheduling problem, which decides $\{T, \mathbf{t}, \mathbf{z}\}$. Note that Q is defined by s and T. However, due to the quadratic term in the objective and constraints, s and $\{T, \mathbf{t}\}$ cannot be decomposed by traditional methods such as Lagrangian relaxation. Considering that the two problems optimize the same objective, we *decouple CPI* into a Capacitated Assortment Planning Problem (*CAPP*) given a common replenishment cycle and a Capacitated Replenishment Scheduling Problem (*CRSP*) given an assortment.

$$(CAPP) \qquad g(T, \mathbf{t}) = \max_{s,x,y} \upsilon' s - T \cdot H' s - \frac{1}{T} \cdot \kappa' y$$
$$s.t. \; (2) - (6), \text{ and } (10)$$

$$(CRSP) \qquad f(s, y) = \max_{T,\mathbf{t},\mathbf{z}} \upsilon' s - T \cdot H' s - \frac{1}{T} \cdot \kappa' y$$
$$s.t. \; (10) - (13), \; (15), \text{ and } (16)$$

Observe that *CAPP* and *CRSP* have the same objective. Further, as shown in Lemma 2 below, we do not need to obtain all the optimal decisions at once; instead, we just need to determine (y^*, s^*) or (T^*, $\mathbf{t}^*$), and then we can get the rest by solving *CRSP* or *CAPP*, respectively. We call this property *preservation of optimality*.

Lemma 2. *If* $\{T^*, \mathbf{t}^*, x^*, y^*, \mathbf{z}^*, s^*\}$ *solves CPI and achieves objective value* V^**, then* $V^* = g(T^*, \mathbf{t}^*) = f(s^*, y^*)$.

Observe that the objective of *CAPP* given T is a linear function, and constraint (10) given (T, $\mathbf{t}$) is a set of linear constraints. We use Lagrangian multipliers to move (10) into the objective. Note that for a given value of the multipliers, the Lagrangian-relaxed *CAPP* has a linear (and thus convex) objective. Then, from Proposition 1, the binary constraints (5) can be replaced by (8) and the problem becomes an LP. In the *FI* algorithm introduced later, we use a bisection (binary search) method to find the Lagrangian multipliers so that constraint (10) holds and the space consumption is feasible in each round of iteration.

The *CRSP* is more complicated and it is not amenable to a standard solution method. However, after examining the structure of *CRSP*, we find that it can be simplified as follows. We first formulate a Maximum Space Minimization Scheduling Problem (*MSMSP*), where B is the amount of space that can accommodate a given assortment and replenishment schedule.

$$(MSMSP) \qquad \underline{B}(s, T) = \min_{\mathbf{t},\mathbf{z},B} B$$
$$s.t. \sum_j c_j s_j \left(t_{ij} + T \cdot \theta_j\right) \le B \quad \forall i, \tag{22}$$
$$(11) - (13), \; (15), \text{ and } (16).$$

MSMSP minimizes the required maximum space given assortment s and cycle time T. Note that *MSMSP* incorporates all of *CRSP*'s constraints except (10), the aggregate space constraint. However, (10) can be incorporated if we add in $B \le C$. Hence, when s is given, T is feasible for *CRSP* if and only if $\underline{B}$ $(s, T) \le C$. This is because if $\underline{B}$ $(s, T) \le C$, then there exists a set of $\{T, \mathbf{t}, \mathbf{z}\}$ that satisfies (10) – (13), (15), and (16), and vice versa. Therefore, if (s, T) is the solution of *MSMSP*, then *CRSP* is reduced to $\min_{T\ge 0} T \cdot H's + \frac{1}{T} \cdot \kappa' y$ subject to $\underline{B}$ $(s, T) \le C$. In the following, we show that *MSMSP* is equivalent to a simpler problem, which we call the Normalized Scheduling Problem (*NSP*). In preparation, let $\beta = B/T$ represent the minimum marginal space required as the cycle time increases.

Lemma 3. *MSMSP is equivalent to the NSP*

$$(NSP) \qquad \underline{B}(s, T) = \min_{\mathbf{t},\mathbf{z},\beta} T \cdot \beta$$
$$s.t. \sum_j c_j s_j \left(t_{ij} + \theta_j\right) \le \beta \quad \forall i, \tag{23}$$

$$\mathbf{t} + \mathbf{t}' = \mathbf{1}_{n\times n} + \mathbf{I}_n, \tag{24}$$

$$\mathbf{z} + \mathbf{z}' = \mathbf{1}_{n\times n} - \mathbf{I}_n, \tag{25}$$

$$t_{ij} - t_{ik} + t_{jk} = z_{kj}, \quad \forall i \ne j \ne k, \tag{26}$$

$$\mathbf{t} \ge 0, \mathbf{z} : \text{binary}. \tag{27}$$

Note that T is not contained in any constraints of *NSP*. Consequently, the optimal β for *NSP* only depends on s. Let $\beta(s)$ be the optimal solution. We thus have $\underline{B}(s, T) = \beta(s) \cdot T$, and constraint $\underline{B}$ $(s, T) \le C$ is equivalent to

$$\beta(s) \cdot T \le C. \tag{28}$$

Therefore, if we can solve *NSP* to obtain $\mathbf{t}$ and $\beta(s)$, then *CRSP* can be solved, as shown next in Proposition 3.

Proposition 3. *The optimal cycle time for the CRSP given s is* $T^*(s) = \min\left\{\sqrt{\frac{\kappa' y}{H's}}, \frac{C}{\beta(s)}\right\}$.

Since the *NSP* is a mixed integer linear program, it could still be NP-hard. However, we find that, by exploiting the property described in Theorem 2 below, we can show that it can be further reduced to an LP. To proceed, let $SC_i\left(\tau_j\right) = c_i s_i \left(t_{ji} + \theta_i\right)$ denote the normalized expected Space Consumption of product i at replenishment time τ_j, and $SC(t) = \sum_i SC_i(t)$ the total space consumption.

Theorem 2. *The optimal solution to NSP is* $\tau^*(s)$ *such that* $SC(\tau_i) = \beta(s)$ *for all* $i \in \{j\text{: } s_j > 0\}$*, and is invariant of the order of products, i.e.,* $\tau^*(s)$ *is not unique.*

Here we give a sketch of the proof. Define $A_s := \{j\text{: } s_j > 0\}$. For any $i \in A_s$, define $SC_{-i}(t) := \sum_{k\in A_s\setminus\{i\}} SC_k(t)$ as the expected total space consumption of products except i. For any $t \ne \tau_k$, where $k \in A_s\setminus\{i\}$, we have $dSC_{-i}(t)/dt = \sum_{k\in A_s\setminus\{i\}} c_k s_k$. Thus, if we increase τ_i by δ_i, i.e., $\tau_i' = \tau_i + \delta_i$, then we get $SC(\tau_i) - SC\left(\tau_i'\right) = \delta_i \cdot \sum_{k\in A_s\setminus\{i\}} c_k s_k$; in other words, the spike generated by the replenishment of i will be reduced by $\delta_i \cdot \sum_{k\in A_s\setminus\{i\}} c_k s_k$. However, at the same time, other spikes will all be increased by $\delta_i c_i s_i$. Therefore, we can solve *NSP* in the following way.

Let $\tau > 0$ be the starting set. Let $\tau_i = 0$ for $i \notin A_s$. Let $i_0 := \arg\min\{\tau_i : i \in A_s\}$ and set $\tau_{i_0} = 0$. Let $A_l = \{i_0\}$. Let $i_1 := \arg\min\{\tau_i : i \in A_s \setminus A_l\}$ and adjust τ_{i_1} so that $SC(\tau_{i_0}) = SC(\tau_{i_1})$. Then let $A_l = A_l \cup \{i_1\}$ and continue until $A_l = A_s$. At last, we will have $\tau_{i_0} < \tau_{i_1} < \cdots$ and $SC(\tau_{i_0}) = SC(\tau_{i_1}) = \cdots = SC(\tau_{i_k}) = \cdots = \beta$ for all $i_k \in A_s$. Now if we adjust any τ_i, we get $\max_t SC(t) > \beta$. It can be verified *ex post* that replenishment follows a cycle in which $\tau_{i_k} - \tau_{i_{k-1}} = c_{i_k} s_{i_k} / \sum s_i c_i := l_{i_k}$, and this is independent of the location of τ_{i_k}; if we switch the order for i_{k-1} and i_k and set $\tau_{i_k} = \tau_{i_{k-2}} + l_{i_k}$ and $\tau_{i_{k-1}} = \tau_{i_k} + l_{i_{k-1}}$, we obtain the same total space consumption β. We can apply this result to switching order between any two products and get the same total space consumption. A detailed proof is provided in the appendix.

Since the order in which products are replenished is not consequential, the binary variables used to enforce the order can be discarded. Therefore, we can use any fixed order, and *NSP* is equivalent to the following Linear Scheduling Problem (*LSP*).

$$(LSP) \qquad \underline{B}(s,T) = \min_{\tau,\beta} T \cdot \beta$$

$$s.t. \sum_i SC_i(\tau_j) \le \beta \quad \forall j, \tag{29}$$

$$SC_i(\tau_j) = c_i s_i (1 + \theta_i) \quad \forall i = j, \tag{30}$$

$$SC_i(\tau_j) = c_i s_i (\tau_i - \tau_j + \theta_i) \quad \forall i > j, \tag{31}$$

$$SC_i(\tau_j) = c_i s_i (\tau_i - \tau_j + 1 + \theta_i) \quad \forall i < j, \tag{32}$$

$$0 \le \tau_i \le \tau_j \le 1 \quad \forall i < j. \tag{33}$$

The minimum, normalized total space consumption is given by the following Corollary. Note that s contains all the information in (s, y) because $y_i = \mathbb{I}(s_i > 0)$. Thus, we use s to replace (s, y) hereafter.

Corollary 1. *Given assortment s, the minimum, normalized total space consumption is*

$$\beta(s) = \sum_{i=1}^{M} \left(s_i c_i \cdot \sum_{j=1}^{i} s_j c_j \right) / \sum_{i=1}^{M} s_i c_i. \tag{34}$$

To summarize, we have transformed the *CPI* into two interrelated problems that are more amenable to computation: The first is a space-relaxed *CAPP*, which is an LP; the second is the *CRSP*, which has a pseudo-closed-form solution that is based on a linear program (*LSP*). In addition, the output solution of one will serve as the input of the other, and they are all feasible. Hence, we can design a Feedback-Iteration (*FI*) algorithm to approach the optimal solution. Due to the optimality-preservation porperty of our decoupled problems, the iteration will converge and stop at optimality. The following decribes the steps in the *FI* algorithm.

4.1.0.1. Feedback-iteration algorithm. [0] Initialize $k = 0$, $s^k = d$, $\mathbf{t}^k = 0$, and $T^k = 0$. Set stopping criteria δ and $\mathcal{S}$. Let $LOWER = L$ be the lower bound, and $\tilde{s} = d$ be the current best assortment.

[1] Let $k = k + 1$ and solve $f(s^{k-1})$ to get an improved cycle time T^k and replenishment schedule $\mathbf{t}^k$.

[2] Solve $g(T^k, \mathbf{t}^k)$ to get an improved assortment s^k.

[3] If $LOWER < f(s^k) - \delta$ (i.e., a significant improvement exists), then let $\tilde{s} = s^k$, update the lower bound by setting $LOWER = f(s^k)$, and go to step [4]; otherwise, stop.

[4] If $k > \mathcal{S}$ (i.e., we have sufficient number of iterations), stop; otherwise, go to step [1].

4.2. Managing combined replenishments

The *CPC* consists of three subproblems: an assortment problem, a consolidating problem, and a scheduling problem. Note that the assortment problem and scheduling problem are related by the space constraint (10), the assortment problem and consolidating problem are related by constraint (17), and the scheduling problem and consolidating problem are related by constraint (19). The assortment and consolidating problems are more closely related because products that are consolidated in one order can be viewed as a single product and thus the consolidating problem essentially generates a new assortment.

To solve the *CPC*, we adopt a heuristic-based, feedback-iteration approach similar to what we use for the *CPI*. This approach consists of three steps. In the first step, we solve the scheduling problem given the full assortment without consolidations. In the second step, we solve the joint assortment and consolidating problem given the schedule and cycle time obtained from the previous step. When solving the joint assortment and consolidating problem, we relaxed the integrality constraints on y and o. The optimal values of y and o will be achieved on the boundaries, given that r's are integers. In this manner, we greatly reduce the number of integer variables in our problem. In the third step, with a new assortment and the consolidation solution, we go back and solve the scheduling problem with the combined products taken as a single product. We repeat this procedure until the optimal profit converges. This approach is formalized by the following Sequenced-Feedback-Iteration (*SFI*) algorithm.

4.2.0.2. Sequenced-feedback-iteration algorithm. [0] Set auxiliary variable $A_{ij} = 1$ for all i and j.[4] Set stopping criteria $\mathcal{S} > N$. Let $\tilde{s} = d$, $\tilde{y} = 1$, $\tilde{o} = 1$, and $\tilde{\mathbf{r}} = \mathbf{I}$. Re-index the groups according to K_n in a descending order. Let pointer $pn = 1$ and the set of product pairs $\Omega = \emptyset$.

[1] If $|\mathcal{P}_{pn}| < 2$ or $(i, j) \in \Omega$ for $\forall i, j \in \mathcal{P}_{pn}$ and $i \neq j$, then set $pn = pn + 1$ and go to step [1]. If $pn \le N$, find $i^*, j^* \in \mathcal{P}_{pn}$ such that $(i^*, j^*) = \arg\min_{(k1,k2)\notin\Omega}\{s_{k1} \cdot c_{k1} \cdot s_{k2} \cdot c_{k2}\}$ and set $A_{i^*j^*} = A_{j^*i^*} = 2$.

[2] Solve *LSP* using the effective assortment $\tilde{s}'\tilde{\mathbf{r}}$ to get $\beta(\tilde{s}'\tilde{\mathbf{r}})$. Solve $L = \max_T v's - T \cdot H'\tilde{s} - \frac{1}{T} \cdot \kappa'\tilde{y} - \frac{1}{T} \cdot K'\tilde{o}$ subject to $0 \le T \le C/\beta(\tilde{s}'\tilde{\mathbf{r}})$, where $K' = \left(K_{n(1)}, \ldots, K_{n(M)}\right)$. Obtain an improved cycle time $\widetilde{T}$.

[3] Solve $U = \max_{s,y,x,o,\mathbf{r}} v's - \widetilde{T} \cdot H's - \frac{1}{\widetilde{T}} \cdot \kappa'y - \frac{1}{\widetilde{T}} \cdot K'o$ subject to (2)–(4), (6), (8), (10), (21), $o_i \in [0, 1]$ for $\forall i \in \mathcal{P}$, $r_{im} + r_{jm} \le A_{ij}$ for $\forall m \in \mathcal{P}$ and $\forall i \neq j$, and $r_{ij} = \tilde{r}_{ij}$ for $\forall i, j \notin \mathcal{P}_{pn}$. Obtain s^*, y^*, o^*, and $\mathbf{r}^*$.

[4] If $U \le L$, set $pn = pn + 1$. Otherwise, set $\tilde{s} = s^*$, $\tilde{y} = y^*$, $\tilde{o} = o^*$, and $\tilde{\mathbf{r}} = \mathbf{r}^*$.

[5] If $pn \le N$, let $\Omega = \Omega \cup \{(i^*, j^*)\}$. If $pn > \mathcal{S}$, stop; otherwise, go to step [1].

Notice that in the second step of this algorithm, we do not solve the consolidating problem for the entire product set. Instead, we solve the consolidating problem group by group, in a descending order of group setup cost. Further, in solving the consolidating problem within a group, we gradually expand the set of products that can be consolidated until the profit is no longer improved. In particular, we each time choose the consolidation of two more products so that this causes the least impact on the required space. The following corollary describes the impact on required space if we combine the replenishment orders of product i and j of the same group. Basically, we treat the combination of product i and j

[4] Note that auxiliary variable A_{ij} can take two possible integer values: 1 and 2. Orders for products i and j can (cannot) be combined if $A_{ij} = 2$ ($A_{ij} = 1$).

as a single product that frees up the space at the rate of $s_i c_i + s_j c_j$ and we denote $\tilde{s}$ as the "new" assortment.

Corollary 2. *The change in the minimum, normalized total space consumption after combining the replenishment of product i and j is*

$$\beta(\tilde{s}) - \beta(s) = s_i \cdot c_i \cdot s_j \cdot c_j / \sum_{k=1}^{M} s_k c_k. \tag{35}$$

4.3. An upper *bound*

To assess the effectiveness of our heuristics, we develop an upper bound on the optimal solution. Here we focus on the *CPC*, because the analysis of the *CPI* is similar. An upper bound for the *CPC* can be obtained by solving the following linear program, which we call the Relaxed *CPC* (*RCPC*). Let T_L and T_U denote a lower and an upper bound on *T*, respectively. Note that $\theta_j \cdot T_L \leq t_{ij} + T \cdot \theta_j$. Thus, we relax the space constraint (10) by replacing it with

$$\sum_j c_j s_j \theta_j T_L \leq C. \tag{36}$$

By dropping t_{ij}, we drop the scheduling problem entirely. Notice that the costs of scheduling are mainly driven by the group setup costs. If the group setup costs are high, retailers should consolidate the orders and compromise on the scheduling optimality; if the group setup costs are low, dropping the scheduling problem will not significantly affect the total profit. The *RCPC* can be formulated as

$$(RCPC) \qquad \max_{s,x,y,r,o} \sum_{i \in \mathcal{P}} \left[v_i s_i - H_i s_i T_L - \frac{\kappa_i y_i}{T_U} - \frac{K_{n(i)} o_i}{T_U} \right]$$
$$s.t.\ (2)-(4),\ (6),\ (8),\ (17),\ (18),\ (36),$$
$$o_i \in [0,1],\ r_{ij} \in \left[0, \mathbb{I}\left\{j \in \mathcal{P}_{n(i)}\right\}\right] \quad \forall i, j.$$

Proposition 4. *The optimal value for RCPC is an upper bound on the CPC.*

Note that for a given T_U and T_L, the *RCPC* is an LP, and thus can be solved efficiently. To find T_U and T_L, we first set $T_U = T_L = T$ and then conduct a linear search for the optimal T^* by utilizing a small step size δ. Once we find the $\hat{T}^*$ that gives the highest profit, then set T_U and T_L plus and minus a step from $\hat{T}^*$, respectively. Note that T^* and $\hat{T}^*$ may not be the same, but we should have $T^* \in \left[\hat{T}^* - \delta, \hat{T}^* + \delta\right]$. Because the optimal value of the *RCPC* given $T_U = T_L = T^*$ is an upper bound for *CPC*, the relaxation of T_U and T_L also generates an upper bound. As shown later in Section 6, this upper bound and the profit generated by our heuristic algorithms are quite close for a wide range of parameter settings, which validates the effectiveness of our heuristics as well as the upper bound.

5. The space-dedication strategy

When space is dedicated to each product, replenishment is more flexible and easier to implement as there is no need to coordinate the replenishment schedules and adopt a common replenishment cycle for all the products. Thus, each product can have a different replenishment cycle time. This flexibility may enable the space-dedication strategy to outperform the space-sharing strategy. However, we still need to solve the consolidating problem in order to reduce setup cost to the extent possible. In this context, if two products are combined for replenishment, we need to ensure that they have the same replenishment schedule.

Here, instead of using a common cycle time *T*, each product *i* has its own cycle time T_i. Given the fixed space allocation, we can replace the space constraints in (10) with a single space constraint

$$\sum_{i \in \mathcal{P}} (1 + \theta_i) c_i s_i T_i \leq C. \tag{37}$$

In addition, we need to enforce that products that are combined for replenishment must have the same cycle time. Let $\mathbb{T}_i$ denote the cycle time of the *i*th order and replace (19) with

$$T_i = \sum_j r_{ij} \cdot \mathbb{T}_j \quad \forall i. \tag{38}$$

Note that (17) will always be binding in order to maximize profit. In other words, $\sum_j r_{ij} = 0$ if $y_i = 0$ and $\sum_j r_{ij} = 1$ if $y_i = 1$. Given such conditions, (38) requires that T_i should equal at most one of the $\mathbb{T}_j$'s. Equivalently, $1/T_i$ should equal at most one of the $1/\mathbb{T}_j$'s. Therefore, (38) can be written as $1/T_i = \sum_j r_{ij}/\mathbb{T}_j$ if $T_i \neq 0$ and $\mathbb{T}_i \neq 0$ for all *i*. This form allows the r_{ij}'s to appear in the numerator and makes it easier to solve the problem. Lastly, all the constraints related to **t** and **z** can be dropped. As a result, the problem of dedicated-space strategy can be formulated as follows.

$$(DSS) \qquad \max \sum_{i \in \mathcal{P}} v_i s_i - \sum_{i \in \mathcal{P}} H_i s_i \sum_{j \in \mathcal{P}} r_{ij} \cdot \mathbb{T}_j - \sum_{i \in \mathcal{P}} \kappa_i y_i \sum_{j \in \mathcal{P}} r_{ij}/\mathbb{T}_j$$
$$- \sum_{j \in \mathcal{P}} K_{n(j)} o_j / \mathbb{T}_j$$
$$s.t.\ (2)-(6),\ (17),\ (18),\ (20),\ (21),\ (37),\ \text{and}\ (38).$$

Because we do not have the scheduling problem, the *DSS* consists of an assortment subproblem, which determines *s* and $\mathbb{T}$, and a consolidating subproblem, which determines **r**. Note that *y* is determined by *s*, and *o* is determined by **r**. Observing that $\mathbb{T}$ is only contained in constraints (37) and (38), we first use a Lagrangian multiplier $\lambda \geq 0$ to move (37) to the objective to get

$$\mathbb{T}_j^*(s, \mathbf{r}, \lambda) = \sqrt{\frac{\sum_i \kappa_i y_i r_{ij} + K_{n(j)} o_j}{\sum_i s_i r_{ij} [(1 + \theta_i) c_i \lambda + H_i]}}. \tag{39}$$

We then use (39) in the following Dedicated-Space Iteration (*DSI*) algorithm to solve this problem.

Dedicated-Space Iteration Algorithm. [0] Let $k = 0$, set $s^k = d$, $y^k = \mathbf{1}$, $o^k = \mathbf{1}$, and $\mathbf{r}^k = \mathbf{I}$.

[1] Let $k = k + 1$. Use (39) to compute $\mathbb{T}^k\left(s^{k-1}, \mathbf{r}^{k-1}, \lambda\right)$ for a given λ.

[2] Use the bisection method to search for the smallest $\lambda^* \geq 0$ that satisfies (37).

[3] Solve *DSS* given $\mathbb{T}^k$, $\mathbf{r}^{k-1}$ and o^{k-1} to get s^k and y^k. If *s* converges, go to [4]; Else, let $\mathbf{r}^k = \mathbf{r}^{k-1}$, $o^k = o^{k-1}$, and go to [1].

[4] Solve *DSS* given $\mathbb{T}^k$, s^k and y^k to get $\mathbf{r}^k$ and o^k. If **r** converges, stop; Else, go to [1].

Note that in steps [3] and [4], respectively, the integrality constraints on *y*, **r**, and *o* can be relaxed and replaced by interval constraints. The argument is similar to the proof of Proposition 1. Further, note that this algorithm preserves optimality because the same objective function and the same set of constraints are used in each step. The objective value will only be improved in each iteration. Hence, the effectiveness of this algorithm depends on how many local maxima there are in the feasible region. Fortunately, the objective function of *DSS* does not have multiple local maxima.[5] Lastly, finding the largest λ^* that dissatisfies (37) in step [2] and relaxing (37) with λ^* in steps [3] and [4] can generate an upper bound, denoted by *DSU*.

[5] By checking the first order conditions of the objective function with respect to all the decision variables, we can find that there can be at most one solution for every variable within the feasible range. In addition, the objective function is not strictly convex for any decision variable. Therefore, the objective function can have only one maximum point in the feasible region. The detailed proof is omitted.

Table 1
An Instance of *CPI*.

i	d_i	v_i	κ_i	c_i	θ_i	h_i	C	w_{ij}	#1	#2	#3
#1	194	6	50	0.02	2	0.02	80	#1	1	0	0
#2	182	19	100	0.07	2	0.07	80	#2	0.4	1	0
#3	190	18	50	0.03	2	0.03	80	#3	0.4	0.4	1

Table 2
Summary of parameters used in computational study.

	Distribution/Formula	Parameters	Distribution/Formula
M	$5 + round(10U)$	N	$roundup(M/5)$
C	$(5 + 30U) \cdot M$	κ_i	$\bar{\kappa} + (U - 0.5)\Delta_\kappa$
CV_i	$U \cdot CV_B$	$\bar{\kappa}$	$30 + 70U$
CV_B	$1 + 9U$	Δ_κ	$50U$
θ_i	$1.65 \cdot CV_i$	c_i	$\bar{c} + (U - 0.5)\Delta_c$
d_i	$\bar{d} + (U - 0.5)\Delta_d$	$\bar{c}$	$0.05 + 0.1U$
$\bar{d}$	$50 + 100U$	Δ_c	$0.09U$
Δ_d	$80U$	H_i	$c_i \cdot \left(\frac{1}{2} + \theta_i\right)$
v_i	$\bar{v} + (U - 0.5)\Delta_v$	K_n	$\bar{K} + (U - 0.5)\Delta_K$
$\bar{v}$	$5 + 10U$	$\bar{K}$	$10 + 40U$
Δ_v	$8U$	Δ_K	$20U$

Note: U represents a uniformly random number on (0,1).

6. Computational study

In this section, we verify the computational difficulty of *CPI* using numerical examples, then evaluate how the heuristics perform against the upper bounds on the *CPC* and *DSS* under various parameter settings, and finally explore when space sharing outperforms space dedication by comparing the performance of the heuristics and upper bounds.

6.1. Computational difficulty

Given the *CPI* can be derived as a special instance of the *CPC*, computing the *CPC* will be at least as hard as the *CPI*. Thus, it suffices to focus on the computational difficulty of the *CPI*. We tried solving the *CPI* using GAMS via NEOS Server.[6] Within NEOS, we employed two powerful, commercially available solvers: the DICOPT solver,[7] which is used for mixed integer nonlinear optimization problems, and the LINDOGlobal solver,[8] which uses branch-and-cut methods to solve non-linearly constrained optimization problems.

We found that the DICOPT solver could not solve the *CPI*, while LINDOGlobal could solve this problem only up to three products. An instance of size 3 is shown in Table 1. Any problem instance with more than 3 products was not solved by the LINDOGlobal solver even after 1,000,000 seconds. Hence, this justifies the need for heuristics to address the *CPI* and the *CPC*.

6.2. Performance of heuristics

In this section, we evaluate the performance of the heuristics used to compute the solutions under the space sharing and the space dedication strategies. Recollect that the Sequenced Feedback Iteration (*SFI*) algorithm was used for the space sharing strategy, while the Dedicated Space Iteration (*DSI*) algorithm was employed for the space dedication algorithm. To evaluate the heuristics, we consider a broad range of parameters (summarized in Table 2) and generated 500 random problems instances (27), each comprising between 5 to 15 candidate product.[9] We then computed the solutions provided by the appropriate algorithms and compared these solutions with the appropriate upper bound solution for each strategy. To solve the optimization problems associated with the upper bounds and heuristics, we use the CVX in Matlab with the Mosek solver (version 7.1.0.12) on a computer with an Intel Core i5-3210M 2.50 gigahertz processor and 4 gigabytes of RAM memory. The performance of the heuristics (V_i, $i \in \{SFI, DSI\}$) is evaluated based on the % gap from their respective upper bounds (V_j, $j \in \{RCPC, DSU\}$). The % gaps are defined as: $\mu_{SFI} := \frac{V_{RCPC} - V_{SFI}}{V_{SFI}} \times 100\%$ and $\mu_{DSI} := \frac{V_{DSU} - V_{DSI}}{V_{DSI}} \times 100\%$.

Table 3
Performances of algorithms.

	Mean	Median	Std. Dev.	Range
V_{RCPC}	9177	8318	5016	28,234
V_{SFI}	8904	7936	4895	27,395
V_{DSU}	9002	8081	4989	31,333
V_{DSI}	8999	8071	4946	28,057
μ_{SFI}	3.30%	2.03%	3.69%	24.97%
μ_{DSI}	0.40%	0.24%	0.95%	11.47%

Table 3 summarizes the mean, median, standard deviation, and range of the values of the upper bounds, heuristics, and the appropriate gaps. The following observations can be made. First, for the space-sharing strategy, the performance of the *SFI* algorithm is reasonably good, with a mean gap of 3.3% from the upper bound. Second, for the space-dedication strategy, the performance of the *DSI* algorithm is extremely good, with a mean gap of 0.4% from the upper bound. This suggests that both heuristics can achieve profits that are very close to the optimal.

6.3. Space sharing vs. space dedication

In this section, we use the performance of the heuristics and the upper bound to better understand the performance of the space sharing and the space dedication strategies. We then use this analysis to develop insight on which strategy is better under which circumstance and why. Although the heuristics are near-optimal, we still cannot compare the performance of space sharing versus space dedication solely based on the gaps between the upper bounds and the heuristics. This is because the gaps could be caused by the inefficiency of the heuristic algorithms or the inefficiency of the upper bound algorithms. Therefore, we need to supplement gaps with other criteria for making these comparisons.

First, we know that space sharing is certainly better (worse) than space dedication if $V_{SFI} > V_{DSU}$ ($V_{RCPC} < V_{DSI}$). Otherwise, we consider the two strategies to have similar performance. We call this the *Absolute Criterion*. However, this criterion is not always satisfied in the numerical experiments. Hence, we need a weaker criterion. The second criterion we use is called the *Bounds Criterion*. We decide that space sharing outperforms space dedication if both $V_{RCPC} > V_{DSU}$ and $V_{SFI} > V_{DSI}$. Similarly, space dedication outperforms space sharing if both $V_{RCPC} < V_{DSU}$ and $V_{SFI} < V_{DSI}$. Else, as before, we define the performance of the two strategies to be similar. We also use an even weaker third criterion, which we call the *Middle Criterion*. Under this criterion, space sharing outperforms space dedication if $\frac{V_{RCPC}+V_{SFI}}{2} > \frac{V_{DSU}+V_{DSI}}{2}$, underperforms space dedication if $\frac{V_{RCPC}+V_{SFI}}{2} < \frac{V_{DSU}+V_{DSI}}{2}$, and similar if $\frac{V_{RCPC}+V_{SFI}}{2} = \frac{V_{DSU}+V_{DSI}}{2}$. As we can see from the following analysis, the results with different criteria are very consistent, which suggests that these criteria can be reliably used to compare space sharing with space dedication.

Given these criteria, we next compare the performance of the two strategies. An important aspect in this paper is the space availability per product represented by C/M. We divide the value of C/M

[6] http://www.neos-server.org/neos/
[7] https://www.gams.com/help/topic/gams.doc/solvers/dicopt/index.html
[8] https://www.gams.com/help/topic/gams.doc/solvers/lindo/index.html
[9] 5˜15 is a reasonable size for a product category such as orange juice, tooth paste, basic undergarments, socks, etc. at a retail store.

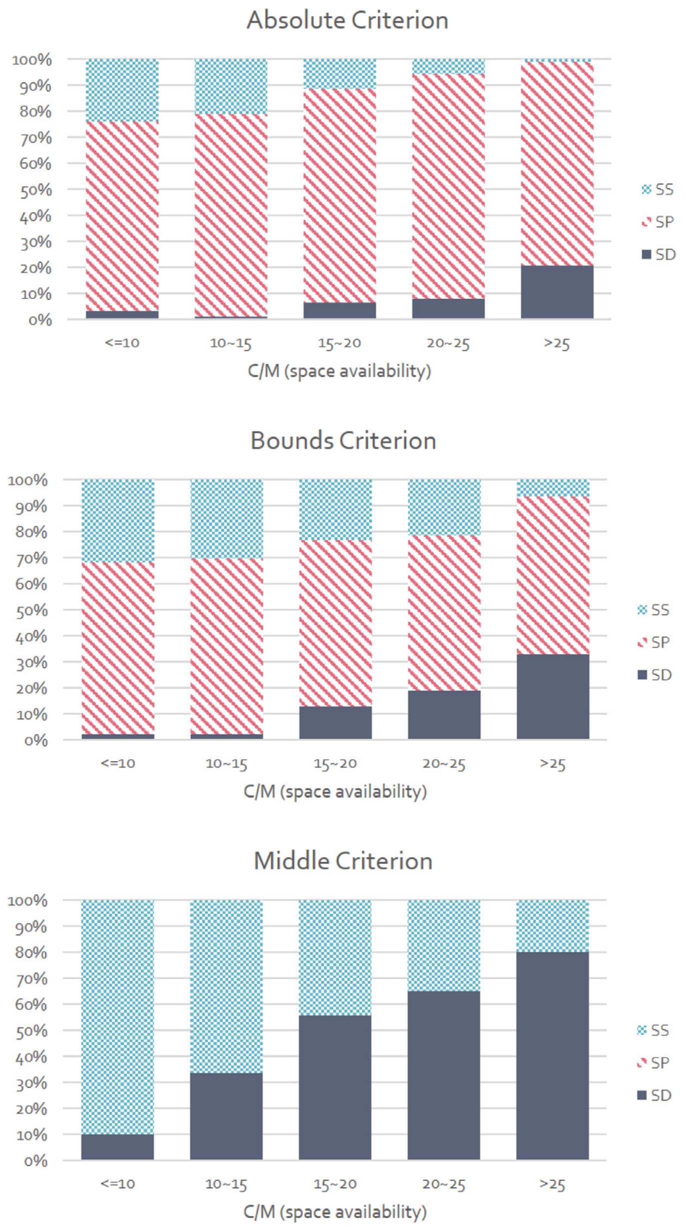

Fig. 4. The Impact ofspace availability on the result of comparison *Note*: SS = space sharing is better; SP = similar performances; SD = space dedication is better.

into several segments (levels). Given a level of space availability per product, we compute the percentage of numerical instances in which space sharing outperforms space dedication, underperforms space dedication, and they have similar performance. The results for the three different criteria are shown in Fig. 4. We can see that space sharing is likely to outperform (underperform) space dedication across all criteria when the level of space availability is low (high). This is intuitive because as the space availability increases, the benefit of space sharing decreases while the benefit of space dedication increases.

To better understand how the performance of the two strategies change with model parameters, we use regression analysis and define the dependent variable as follows. We set the dependent variable to 1 if space sharing is better than dedication, -1 if space dedication is better than sharing, and 0 if the performance is similar. The following variables are included in the regression model: M, CV_B, C, C/M, $\bar{d}$, Δ_d, $\bar{v}$, Δ_v, $\bar{\kappa}$, Δ_κ, $\bar{c}$, Δ_c, $\bar{K}$, and Δ_K. Based on this analysis, we find that factors that significantly influence the relative performance of the two strategies include: the number of products (M), the demand variability (CV_B), the availablility of storage space (C/M), the mean demand level ($\bar{d}$), the mean order setup cost for each product ($\bar{\kappa}$), the mean space consumption rate ($\bar{c}$), and the mean order setup cost for each group ($\bar{K}$). Details on these significant factors are summarized in Table 4.

Table 4
Regressions results for selected factors.

	Absolute criterion	Bounds criterion	Middle criterion
M	−0.051	−0.0700	−0.0538
	(0.0155)	(0.0208)	(0.0320)
CV_B	−0.0256***	−0.0157*	−0.0255*
	(0.0069)	(0.0093)	(0.0143)
C/M	−0.0296***	−0.0439***	−0.0537***
	(0.0073)	(0.0098)	(0.0151)
$\bar{d}$	0.0025***	0.0023***	0.0035***
	(0.0006)	(0.0008)	(0.0012)
$\bar{\kappa}$	0.0038***	0.0067***	0.0091***
	(0.0009)	(0.0012)	(0.0019)
$\bar{c}$	2.8526***	2.4862***	3.7486***
	(0.6253)	(0.8401)	(1.2901)
$\bar{K}$	−0.0026*	−0.0055***	−0.0099***
	(0.0016)	(0.0021)	(0.0032)

Note: Standard errors are in brackets. *$p < 0.1$; ** $p < 0.05$; ***$p < 0.01$.

Our analysis shows that space sharing or space dedication can be optimal, depending on the parameter setting. In this regard, Table 4 can be used to draw the following conclusions. First, the benefits of space sharing increase as the number of products, demand variability across products and space availability per product decrease. Among these observations, it is interesting that space sharing is more attractive when we have fewer products. This is because fewer products require less replenishment coordination and less associated costs. As demand variability decreases, there is more stability in space consumption across products making space sharing more effective and space dedication less necessary. As space availability per product decreases, the gains from space sharing increase since the costs under space dedication increase due to more frequent replenishments now necessary due to lower inventory. Second, the benefits from space sharing increase when average product demand, product replenishment setup cost and average space consumption rate increases. As demand or product space consumption increases, there are larger inventory requirements making space dedication more costly to implement. As product replenishment setup costs increase, there are more benefits for space sharing as it allows less frequent replenishments by increasing space utilization. However, if the group replenishment setup costs increase, the benefits of space sharing are diminished due to the increased cost of coordinating replenishments within a group.

We next consider the potential reduction in space under space sharing when compared to space dedication. To conduct this analysis, define the percentage space consumption gap as 100%× (1 - [space consumed with space sharing]/[space consumed with space dedication]). In Fig. 5, we plot the space consumption by space sharing and space dedication, respectively, for 200 different assortments. The average space consumption gap is 31% with a standard deviation of 5.53%. Therefore, by sharing space with an optimal replenishment scheduling program, we can on average reduce space consumption by 31% across a range of parameter values, which is quite significant. The practical implication is that by adopting an inventory strategy that uses space sharing rather than space dedication, we can potentially carry more products with the same amount of space. Conversely, for the same assortment, space sharing requires a smaller storage or display room, and this could potentially lead to lowered property rent and administration costs if

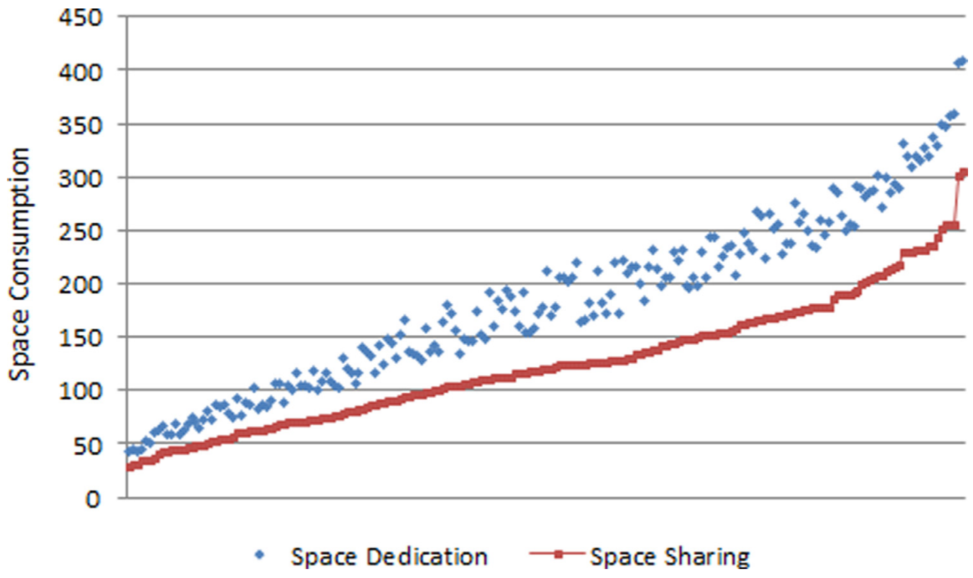

Fig. 5. Marginal impact of space sharing on space consumption *Note:* The dots represent space consumed by dedication and sharing strategies, respectively, for 200 different assortments. The assortments are ordered and indexed according to the consumption by space sharing strategy. The horizontal axis is the index of the assortment. The graph shows how the gap evolves as the space consumption increases.

this strategy is chosen before we make the decision on how much retail space to procure.

To summarize, the following managerial insights can be drawn from the computational study. First, the optimal choice of inventory strategy is not clear *a priori*, since this depends on many different factors that are related to space availability, the total number of products, their demand characteristics, and the cost of inventory replenishment. Second, space sharing is likely to be a better choice for basic product categories that have small product choice sets, lower demand variability, less available space, greater demand rates, higher individual setup costs, higher space consumption rates, and lower group setup costs. Third, the space-sharing strategy could be used to carry more products or possibly save space and retail rents.

7. Conclusions

In this paper, we formulated the retail space management problem for basic products. The main components are assortment selection, replenishment scheduling, and the consolidation of product replenishment. These components are linked by the storage or shelf space constraints. We showed that this problem is NP-hard. We therefore decoupled the joint optimization problem into an assortment planning problem, a replenishment scheduling problem, and a consolidation problem. We proved that the sequencing problem in the replenishment scheduling can be disregarded and this allows for an efficient solution to this problem. We also developed heuristic algorithms and upper bounds to solve this problem under space sharing and space dedication strategies and defined three criteria for the comparison. The algorithms are efficient and the performance gaps are small relative to an upper bound on the optimal solution.

Using an extensive computational study, we find that by sharing space with the optimal replenishment scheduling program, space consumption can be reduced by 31% on average. While space dedication is easier to implement as it does not require coordination of replenishment schedules across products, our results show that it requires more space than space sharing does. In addition, the relative benefit of space sharing over space dedication depends on a number of different factors (e.g., space availability, demand rate, space consumption rate, and order set up costs), which are described and analyzed. Interestingly, we find that space sharing is more attractive when we have fewer products. This is because fewer products require less replenishment coordination and less associated costs. Furthermore, we find that individual product and group order setup costs—which are not directly related to the cost of shelf space—impact the relative performance of space sharing in opposite ways. Particularly, space sharing is more attractive if suppliers charge less for each order or charge more for each product being ordered. This is because space sharing allows longer replenishment cycles (and makes replenishment less frequent for each product), but it requires that fewer products be consolidated in an order.

The findings of this paper can be potentially used by retailers that sell basic products and are not required to dedicate their retail space. To do so, it is important that the appropriate data is collected and used to estimate the demand substitution matrix and all the other relevant parameters. Our heuristic algorithms are then easy to implement as they require minimal coding. Although our computational study only shows the performance of our heuristics for problem instances that have 5 ˜ 15 candidate products (i.e., the number of products for a typical product category), our heuristics can be jointly used to solve problems at the store level. For instance, we can first divide the entire retail shelf or storage space into a number of independent segments dedicated to different product categories. Then in each scenario, we can use the space-sharing and space-dedication heuristics to determine the optimal assortment and replenishment schedule for each product category. By repeating this procedure many times, we can find out the optimal space allocation plan for the entire store.

This model has the following limitations. First, we do not consider complementary effects among products. This is because in this paper we focus on the assortment and inventory management for products in the same category, and it is more reasonable to consider substitution effects. When we need to jointly manage multiple categories, complementary effects should be considered, and coordinating replenishments will become even more challenging. Second, our methods and results could be tested with other types of demand models. Third, various practical situations may entail additional and different constraints. Extending our model to incorporate these constraints in a real application could be a fruitful area for future research.

Appendix

The Proof of Lemma 1. Given $Q_i = \sqrt{\frac{\kappa_i s_i}{H_i}}$ for $\forall i$, objective (1) now becomes $\sum_{i\in N}\left(\nu_i s_i - 2\sqrt{H_i\kappa_i s_i}\right)$. Since $\sqrt{s_i}$ is concave, $-\sqrt{s_i}$ is convex in s_i. $\nu_i s_i$ is linear in s_i, so the objective is convex in s_i.

The Proof of Lemma 2. Suppose $W^* := \{x^*, y^*, \mathbf{z}^*, s^*, T^*, \mathbf{t}^*\}$ maximizes V, *CPI*'s objective. If $\{x^*, y^*, s^*\} \notin \arg\max V(x, y, s|\mathbf{z}^*, T^*, \mathbf{t}^*)$, then $\exists\{x^0, y^0, s^0\}$ such that $V(x^0, y^0, s^0, \mathbf{z}^*, T^*, \mathbf{t}^*) > V(W^*)$, which contradicts the fact that W^* maximizes V. A similar argument applies to $\{\mathbf{z}^*, T^*, \mathbf{t}^*\} = \arg\max V(\mathbf{z}, T, \mathbf{t}|x^*, y^*, s^*)$. Then $g(T^*) = V(W^*) = f(s^*, y^*)$.

The Proof of Lemma 3. The constraint (26) can be obtained by substituting (13) into (12). Then we can scale up both sides of all the constraints by T and define $t'_{ij} = t_{ij}\cdot T$ as a decision variable, to get the *MSMSP*.

The Proof of Proposition 1. Note that by definition $s_i = \sum_j w_{ij}d_j x_{ij}$, so s is linear in x. Because $w_{ij}d_j \ge 0$ for $\forall i, j$, so by Lemma 1, we know that the objective is convex in x_{ij}. Hence, the optimal x_{ij} is either 0 or the maximal. Combining (3) and (4), we have for a specific i that $x_{ij} \le y_i \le 1 - x_{ki}$ for $\forall j$ and $\forall k \ne i$.

Now suppose the binary constraints in (5) is replaced by $y_i \in [0, 1]$ and there exists i such that the optimal $y_i^* \in (0, 1)$. Let $\mathcal{J}$ and $\mathcal{K}$ be two sets such that for $\forall j \in \mathcal{J}$ and $\forall k \in \mathcal{K}$ we have $x_{ij}^* > 0$ and $x_{ki}^* > 0$. (I) If $\mathcal{J} = \mathcal{K} = \varnothing$, then the value of y_i is irrelevant and the LP relaxation would not affect the optimal value. (II) If $\mathcal{J} = \varnothing$ but $\mathcal{K} \ne \varnothing$, then we should have $y_i^* = 0$, which is a contradiction. (III) If $\mathcal{K} = \varnothing$ but $\mathcal{J} \ne \varnothing$, then we should have $y_i^* = 1$, which is a contradiction. (IV) Suppose $0 < x_{ij}^* = y_i^* = 1 - x_{ki}^* < 1$ for $\forall j \in \mathcal{J}$ and $\forall k \in \mathcal{K}$. Thus, it is profitable to increase both x_{ij} and x_{ki}, and the total marginal value of increasing all the x_{ij}'s and that of increasing all x_{ki}'s are equal. However, since the objective is convex in x, we can either increase y_i or decrease y_i to increase the objective value. If we increase y_i, then the total marginal gain from x_{ij}'s will overweigh the total marginal loss from x_{ki}'s. The same argument applies to decreasing y_i. As a result, we should have $y_i^* = 0$ or 1, which is a contradiction. Therefore, an LP relaxation on y will not affect the optimal solution.

The Proof of Proposition 2. At any time t such that $\tau_i - T \le t < \tau_i$, the average inventory level $I_i(t)$ equals the average demand during $[t, \tau_i]$ plus the amount of safety stock $T\cdot s_i\cdot\theta_i$; Similarly, At any time t such that $\tau_i \le t < \tau_i + T$, the average inventory level $I_i(t)$ equals the average demand during $[t, \tau_i + T]$ plus the amount of safety stock. Note that $\tau_i - T$ and $\tau_i + T$ are the order delivery time in the previous and next cycle, respectively. Accordingly, we have

$$I_i(t) = \begin{cases} s_i(\tau_i - t) + T\cdot s_i\cdot\theta_i, & t < \tau_i \\ s_i(T + \tau_i - t) + T\cdot s_i\cdot\theta_i, & t \ge \tau_i \end{cases}$$
$$= s_i[T\cdot\mathbb{I}\{t \ge \tau_i\} - t + \tau_i + T\cdot\theta_i]. \tag{A1}$$

For $t = \tau_j$, we can write (A1) as $I_i(\tau_j) = s_i\left(t_{ji} + T\cdot\theta_i\right)$ except when $j < i$ and $\tau_i = \tau_j$. (Note that $t_{ji} = 0$ when $j < i$ and $\tau_i = \tau_j$, but we should have $I_i(\tau_j) = s_i(T + T\cdot\theta_i)$.) However, this issue will not affect the optimal solution, because we have two space constraints at time $\tau_i = \tau_j$: $I_i(\tau_i) + I_j(\tau_i) \le C$ and $I_i(\tau_j) + I_j(\tau_j) \le C$. One of the constraints must be effective in the sense that it always represents the true situation and is always tighter. This logic applies to the case wherein more than two products are replenished at the same time and the space constraint corresponding to the product with the largest index is always effective. The rest of the constraints are redundant.

The Proof of Proposition 3. Note that *CRSP* given s can be written as $\min_{0\le T\le C/\beta(s)} T\cdot H's + \frac{1}{T}\cdot\kappa' y$, which is a simple constrained convex optimization problem. Using the KKT conditions, we can obtain the following. If $\sqrt{\frac{\kappa' y}{H's}} < \frac{C}{\beta(s)}$, then $T^*(s) = \sqrt{\frac{\kappa' y}{H's}}$; If $\sqrt{\frac{\kappa' y}{H's}} \ge \frac{C}{\beta(s)}$, then we have $\lambda(s) = \frac{\beta(s)^2}{C^2}\kappa' y - H's$ and thus $T^*(s) = \sqrt{\frac{\kappa' y}{H's + \lambda(s)}} = \frac{C}{\beta(s)}$.

The Proof of Proposition 4. First, since we no longer need $\mathbf{t}$ in the space constraint, we drop all constraints related to $\mathbf{t}$ and $\mathbf{z}$. Second, we substitute (14) into the objective function to eliminate Q_i and its associated domain in (7). Third, we relax (5), (20), and (21) by replacing them with their linear relaxations. These relaxations of (5), (20), (21) and (10) and the eliminations of (11) through (16) relax the feasible set. This leads to a non-linear optimization problem with objective $\sum_{i\in\mathcal{P}}\left[\nu_i s_i - H_i s_i T - \frac{\kappa_i y_i}{T} - \frac{K_{n(i)} o_i}{T}\right]$ and linear constraints. Next, it is clear that $\sum_{i\in\mathcal{P}}\left[\nu_i s_i - H_i s_i T_L - \frac{\kappa_i y_i}{T_U} - \frac{K_{n(i)} o_i}{T_U}\right]$ is an upper bound on the objective of the *CPC*. Hence, the result follows.

The Proof of Theorem 1. To simplify this problem, we eliminate constraint (14) by replacing Q_i with $s_i T$ in the objective function. This removes a set of quadratic equality constraints. By introducing an auxiliary variable λ, and by adding quadratic inequality constraint $\lambda T \ge 1$, we can transform the objective function into

$$\max \sum_{i\in\mathcal{P}} \nu_i s_i - \sum_{i\in\mathcal{P}} H_i s_i T - \sum_{i\in\mathcal{P}} \kappa_i y_i \lambda, \tag{A2}$$

Thus, we transform the *CPI* into a Mixed Integer Quadratically Constrained Quadratic Program (MIQCQP), in which the objective is not convex. In addition, a special instance of the *CPI* can be obtained by dropping the integrality constraints so that we get a non-convex QCQP, which is NP-hard (10). Since a special instance of the *CPI* is NP-hard, this reduction establishes that the *CPI* is also NP-hard.

The Proof of Theorem 2. The main idea of the proof is to show that any change of the product replenishment order does not affect the maximum space consumption. To prove this, it suffices to show that if we switch the order for any two adjacent products, this does not affect the maximum space consumption. To be consistent with *NSP*, we normalize T to 1.

Claim 1. *For any given product order* $\Omega = \left\{\tau_{i_1} \le \cdots \le \tau_{i_n}\right\}$, *we have* $SC\left(\tau_{i_k}^*\right) = \beta_\Omega$ *for all* $i_k \in \{j\colon s_j > 0\}$ *if* $\tau^* = \arg\min_{\tau|\Omega}\max_{0\le t\le 1} SC(t)$.

Without loss of generality, suppose $\max_{0\le t\le 1} SC(t) = SC\left(\tau_{i_k}\right) > SC\left(\tau_{i_m}\right)$ for all $m > k$, then we can shift all τ_{i_m} slightly earlier for all $m > k$, then $\max_{0\le t\le 1} SC(t)$ is reduced.

Claim 2. *For* $\tau^* = \arg\min_{\tau|\Omega}\max_{0\le t\le 1} SC(t)$, *if we switch the order of any two adjacent products and get* $\Omega' = \left\{\cdots \le \tau_{i_{k+1}} \le \tau_{i_k} \le \cdots\right\}$, *then we have* $\beta_{\Omega'} = \beta_\Omega$.

For notational simplicity, let $i = i_k$ and $j = i_{k+1}$, so $j = i + 1$. The graphical representation of inventory level with the optimal schedule given Ω is shown in Fig. A1. In Fig. A1, we use the solid line to represent the aggregate inventory level for products except i, j, and use the dotted line to represent the aggregate inventory level for all products. Hence, the slope of the dotted line is $\Sigma_k c_k s_k$ and the slope of the solid line is $\Sigma_{k\ne i,j} c_k s_k$. In addition, denote $x := \tau_i^* - \tau_{i-1}^*$ and $y := \tau_j^* - \tau_{i-1}^*$. Therefore, we have

$$a = x\cdot c_i s_i + y\cdot c_j s_j. \tag{A3}$$

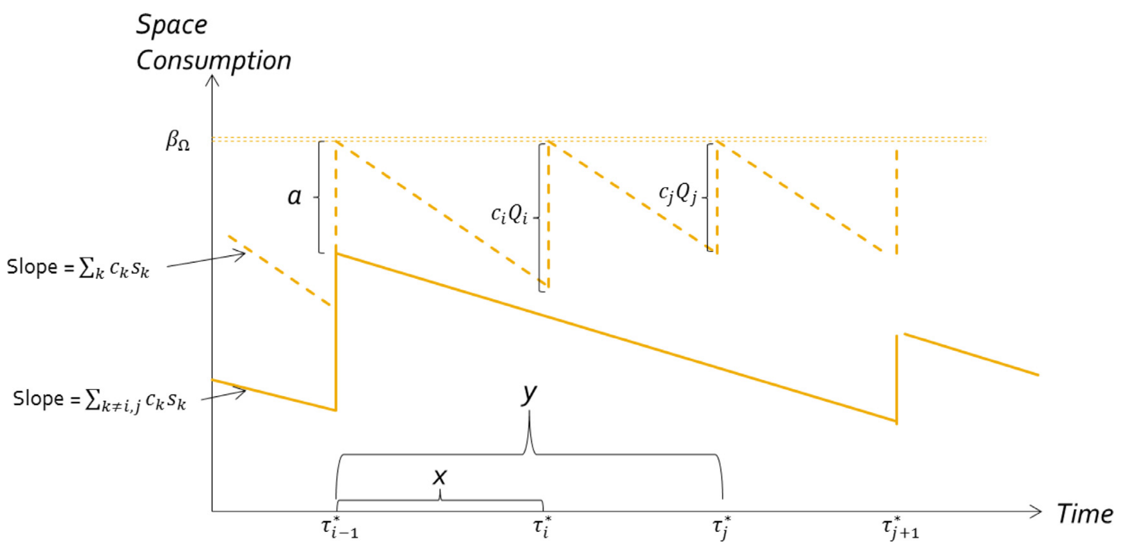

Fig. A1. Inventory levels with optimal schedules.

According to Claim 1, we have $\beta_\Omega = SC(\tau^*_{i-1}) = SC(\tau^*_i) = SC(\tau^*_j)$. First, by using $SC(\tau^*_{i-1}) = SC(\tau^*_i)$, we get

$$a + x\cdot\sum_{k\neq i,j} c_k s_k = c_i Q_i + (y-x)\cdot c_j s_j. \qquad (A4)$$

Second, from $SC(\tau^*_{i-1}) = SC(\tau^*_j)$, we obtain

$$a + y\cdot\sum_{k\neq i,j} c_k s_k = c_j Q_j + (c_i Q_i - (y-x)\cdot c_i s_i). \qquad (A5)$$

Moreover, we have that $Q_i = s_i T = s_i$ for all i. By combining (A3), (A4) and (A6), we get $x = c_i s_i / \sum_k c_k s_k$ and $y = (c_i s_i + c_j s_j)/\sum_k c_k s_k$. Therefore,

$$a = \frac{(c_i s_i)^2 + (c_j s_j)^2 + c_i s_i c_j s_j}{\sum_k c_k s_k}. \qquad (A6)$$

Now if we switch the order of i, j and go through the same analysis, then we have $y = c_j s_j / \sum_k c_k s_k$ and $x = (c_i s_i + c_j s_j)/\sum_k c_k s_k$, which lead to $a' = x\cdot c_i s_i + y\cdot c_j s_j = a$. Note that the schedule of other products are not changed, so the solid line is unaffected. Hence, $\beta_{\Omega'} = \beta_\Omega$.

The Proof of Corollary 1. Without loss of any generality, suppose the products are sequenced by their index. We know that the time interval between product $i-1$ and i is $l_i = c_i s_i / \sum s_j c_j$. Suppose product 1 is replenished at time l_1 and product M is replenished at time $t = 1$ or 0. Hence, the total space consumption at time 0 is $\beta(s) = \sum_{i=1}^M s_i c_i \sum_{j=1}^i l_j$, which can be rewritten as (37).

The Proof of Corollary 2. Denote $u_i = s_i \cdot c_i$ and $U = \sum_{i=1}^M s_i c_i$. It is easy to verify that

$$\beta(s) = \left(u_i^2 + u_i\cdot\sum_{k\neq i,j} u_k + u_j^2 + u_j\cdot\sum_{k\neq i,j} u_k + u_i\cdot u_j + L\right)/U, \qquad (A7)$$

where L does not contain u_i and u_j. Similarly, we have

$$\beta(\tilde{s}) = \left[(u_i + u_j)^2 + (u_i + u_j)\cdot\sum_{k\neq i,j} u_k + L\right]/U. \qquad (A8)$$

Hence, $\beta(\tilde{s}) - \beta(s) = u_i\cdot u_j / U$.

References

Anily, S., & Federgruen, A. (1991). Capacitated two-stage multi-item production/inventory model with joint setup costs. *Operations Research, 39*(3), 443–455.

Bernstein, F., Kök, A. G., & Xie, L. (2015). Dynamic assortment customization with limited inventories. *Manufacturing & Service Operations Management, 17*(4), 538–553.

Borin, N., Farris, P. W., & Freeland, J. R. (1994). A model for determining retail product category assortment and shelf space allocation. *Decision Sciences, 25*(3), 359–384.

Brewster, C. (2016). Dedicated Warehousing vs. Shared Warehousing. BusinessBib (October 27). http://www.businessbib.net/dedicated-warehousing-vs-shared-warehousing/, accessed in May, 2017.

Bultez, A., & Naert, P. (1988). SHARP: Shelf allocation for retailer's profit. *Marketing Science, 7*(3), 211–231.

Bultez, A., Naert, P., Gijsbrechts, E., & Abelle, P. V. (1989). Asymmetric cannibalism in retail assortment. *Journal of Retailing, 65*(2), 153–192.

Cachon, G. (2001). Managing a retailer's shelf space, inventory, and transportation. *Manufacturing & Service Operations Management, 3*(3), 211–229.

Caro, F., & Gallien, J. (2007). Dynamic assortment with demand learning for seasonal consumer goods. *Management Science, 53*(2), 276–292.

Corstjens, M., & Doyle, P. (1981). A model for optimizing retail space allocations. *Management Science, 27*(7), 822–833.

d'Aspremont, A., & Boyd, S. (2003). *Relaxations and randomized methods for nonconvex QCQPs*. Stanford University.

Federgruen, A., & Zheng, Y. S. (1992). The joint replenishment problem with general joint cost structures. *Operations Research, 40*(2), 384–403.

Gaur, V., & Honhon, D. (2006). Assortment planning and inventory decisions under a locational choice model. *Management Science, 52*(10), 1528–1543.

Geng, P. C., & Vickson, R. G. (1988). WRLSP: A single-machine, warehouse restricted lot scheduling problem. *IIE Transactions, 20*(4), 354–359.

Hadley, G., & Whitin, T. M. (1963). *Analysis of inventory systems*. Prentice Hall.

Honhon, D., Gaur, V., & Seshadri, S. (2010). Assortment planning and inventory decisions under stockout-based substitution. *Operations Research, 58*(5), 1364–1379.

Jackson, P., Maxwell, W., & Muckstadt, J. (1985). The joint replenishment problem with power-of-two intervals. *IIE Transactions, 17*, 25–32.

Khouja, M., & Goyal, S. (2008). A review of the joint replenishment problem literature:1989-2005. *European Journal of Operational Research, 186*(1), 1–16.

Kök, A. G., & Fisher, M. L. (2007). Demand estimation and assortment optimization under substitution: Methodology and application. *Operations Research, 55*(6), 1001–1021.

Kök, A. G., Fisher, M. L., & Vaidyanathan, R. (2009). Assortment planning: Review of literature and industry practice. In N. Agrawal, & S. A. Smith (Eds.), *Retail supply chain management*.

Levy, M., & Weitz, B. A. (2008). *Retail management*. McGraw-Hill Education. 7 edition.

Rajaram, K. (2001). Assortment planning in fashion retailing: Methodology, application and analysis. *European Jornal of Operational Research, 129*(1), 186–208.

van Ryzin, G., & Mahajan, S. (1999). On the relationship between inventory costs and variety benefits in retail assortments. *Management Science, 45*(11), 1496–1509.

Smith, S. A., & Agrawal, N. (2000). Management of multi-item retail inventory systems with demand substitution. *Operations Research, 48*(1), 50–64.

Urban, T. L. (1998). An inventory-theoretic approach to product assortment and shelf-space allocation. *Journal of Retailing, 74*(1), 15–35.

Viswanathan, S., & Mathur, K. (1997). Integrating routing and inventory decisions in one-warehouse multi-retailer multiproduct distribution systems. *Management Science, 43*(3), 294–312.

Yao, M.-J., & Chu, W.-M. (2008). A genetic algorithm for determining optimal replenishment cycles to minimize maximum warehouse space requirements. *Omega, 36*, 619–631.

Zhang, W., & Rajaram, K. (2017). Managing limited retail space for basic products. Mendeley Data, v2. http://dx.doi.org/10.17632/s8ppj2hpdd.2.