10-2019

How to optimize storage classes in a unit-load warehouse

Marcus ANG
Singapore Management University, marcusang@smu.edu.sg

Yun Fong LIM
Singapore Management University, yflim@smu.edu.sg
DOI: https://doi.org/10.1016/j.ejor.2019.03.046

How to Optimize Storage Classes in A Unit-Load Warehouse

Marcus Ang*, Yun Fong Lim[1,*]

Abstract

We study a problem of optimizing storage classes in a unit-load warehouse such that the total travel cost is minimized. This is crucial to the operational efficiency of unit-load warehouses, which constitute a critical part of a supply chain. We propose a novel approach called the FB method to solve the problem. The FB method is suitable for general receiving-dock and shipping-dock locations that may not coincide. The FB method first ranks the locations according to the frequencies that they are visited, which are estimated by a linear program based on the warehouse's layout as well as the products' arrivals and demands. The method then sequentially groups the locations into a number of classes that is implementable in practice. After forming the classes, we use a policy based on robust optimization to determine the storage and retrieval decisions. We compare the robust policy with the traditional storage-retrieval policies on their respective optimized classes. Our results suggest that if the warehouse utilization is low, different class-formation methods may lead to very different total travel costs, with our approach being the most efficient. We observe the robustness of this result across various parameter settings. A case study based on data from a third-party logistics provider suggests that the robust policy under the FB method outperforms the other storage-retrieval policies by at least 8% on average, which indicates the potential savings by our approach in practice. One of our findings is that the importance of optimizing classes depends on the warehouse utilization.

Keywords: logistics, unit-load warehouse, storage-retrieval policy, class-based storage

1. Introduction

Warehouses play an important role in a supply chain as they support business by storing, consolidating, and distributing products. The operational efficiency of a warehouse is often crucial to the supply chain's performance because up to thousands of different products are stored and retrieved daily (de Koster et al. 2007 and Gu et al. 2007). We study a *unit-load* warehouse where all products are

*Lee Kong Chian School of Business, Singapore Management University, Stamford Road, Singapore 178899
Email addresses: marcusang@smu.edu.sg (Marcus Ang), yflim@smu.edu.sg (Yun Fong Lim)
[1]Corresponding author

stored and retrieved in unit-load (pallet) quantities. Only items of the same product are stored on each pallet, which is usually handled singly at a time. Unit-load warehouses can be found in the reserve areas of large distribution centers. These reserve areas store products to replenish fast-pick areas in separate locations within the centers (Bartholdi and Hackman 2016). For illustration, we assume that each pallet is moved by a forklift. However, as we will see later, our model can be applied to warehouses using very-narrow-aisle trucks or automated storage and retrieval systems (Roodbergen and Vis 2009).

Pallets arriving in every period (say, every day) at a unit-load warehouse generally follow some schedule predetermined by the suppliers' production plans. The arriving pallets are stored to locations within the warehouse. These pallets are then retrieved from the storage locations when random demands occur. We assume *single-command* travel. To store an arriving pallet under this assumption, a forklift first carries the pallet from a receiving dock to a location. After inserting the pallet into the location, the forklift returns to the receiving dock. Similarly, to retrieve a departing pallet, a forklift first moves from a shipping dock to the pallet's location, extracts the pallet, and then carries it to the shipping dock. We ignore the time durations to insert and to extract each pallet at its storage location.

Many unit-load warehouses are managed by third-party logistics companies. Minimizing the total travel cost (distance or time) is crucial for such a unit-load warehouse to remain competent given its tight profit margins in today's business environment. Furthermore, the responsiveness of the warehouse is getting more important as its clients typically require their goods to be delivered promptly. To maximize its efficiency and responsiveness, the warehouse solves a *storage and retrieval problem* to determine the locations to store the arriving pallets and the locations to retrieve the departing pallets, with an objective of minimizing the total travel cost over a planning horizon.

A well-known approach to determine the storage and retrieval decisions is the *class-based turnover (TOS) policy* (Hausman et al. 1976, Graves et al. 1977, and Schwarz et al. 1978). Storage locations are first grouped into a few classes, and then this policy assigns products with the highest turnover rate to the class with the smallest average travel cost. Within a class, a pallet is assigned randomly to any empty location. Goetschalckx and Ratliff (1990) introduce the *class-based duration-of-stay (DOS) policy*, which sorts the pallets in ascending duration of stay and sequentially assigns the pallets to a predetermined number of classes. The pallets with the shortest duration of stay are assigned to the class with the smallest average travel cost. More recently, Ang et al. (2012) study the storage and retrieval problem with time-varying, deterministic arrivals and non-stationary, stochastic demands. They introduce a storage-retrieval policy called the *restricted linear decision rule* (RLR) that is based on a robust optimization formulation. Their extensive numerical experiments suggest that the RLR

significantly outperforms the TOS and the DOS policies.

Ang et al. (2012) assume that the storage locations are grouped into classes based on a grid. Given the classes, they compare the RLR with the TOS and the DOS policies in terms of the total travel cost. However, the performance of a storage-retrieval policy (the RLR, TOS, or DOS policy) can be quite different if the classes are formed in different ways. Hausman et al. (1976) and Goetschalckx and Ratliff (1990) have described methods to optimize the classes for the TOS and the DOS policies respectively. For the RLR, what is an optimal way to group storage locations into classes such that the total travel cost is minimized? How many classes are needed and how big should each class be? To ensure operational efficiency, a warehouse manager needs to address these important and challenging questions. The problem is non-trivial especially if the warehouse has general receiving-dock and shipping-dock locations, and the products have time-varying arrivals and stochastic demands in a multi-period setting.

Classes are used in practice because it is too tedious to manage (say, thousands of) individual locations for a large warehouse. Furthermore, the storage and retrieval problem may become computationally intractable if individual locations are considered in the model. Figure 1 shows two different ways to group storage locations into a class. In Figure 1(a), locations 1 and 4 with an identical travel distance from the receiving and shipping docks (denoted as R/S) are grouped into one class. In Figure 1(b), locations 1 and 2 are grouped into one class because of their proximity to each other (they have similar coordinates). Different ways of forming the classes may lead to very different solutions to the storage and retrieval problem, and thus result in substantial differences in the total travel cost.

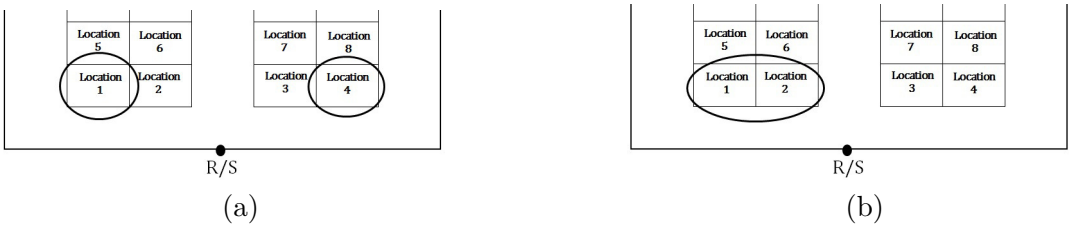


Figure 1: **Different ways of grouping the locations.** (a) Locations with the same total travel distance from the receiving and shipping docks (R/S) are grouped into one class. (b) Locations near each other are grouped into one class.

Many papers in the literature study different methods to form classes such that the total travel cost is minimized. Almost all of these papers assume a specific layout in which the receiving and shipping docks coincide at a corner of the warehouse (Hausman et al. 1976, Rosenblatt and Eynan 1989, Eynan and Rosenblatt 1994, Kouvelis and Papanicolaou 1995, Yu et al. 2015). In this paper, we introduce a class-formation method, called the FB method, for general receiving-dock and shipping-dock locations that may not coincide. Although we present the FB method using a warehouse with horizontal and vertical aisles, the method can be applied to more general aisle orientations (see Öztürkoğlu et al. 2014).

Furthermore, all of the above papers rank each storage location based on the travel cost between the location and the receiving and shipping docks. This approach considers the warehouse's layout, which is determined by the receiving and shipping docks' locations and the aisles' orientations. However, this approach ignores other system characteristics such as the arrivals and the demands of products. In contrast, the FB method ranks each storage location based on its *visit frequency*: the number of times the location is visited over the planning horizon. The visit frequency of each location can be estimated by a linear programming (LP) model, which considers not only the warehouse's layout, but also the products' arrivals and demands. Our paper shows the additional value to warehouse managers of considering more-detailed information: (i) the difference between the store cost and the retrieve cost of each location, and (ii) the imbalance between the inflow and the outflow of each product.

Our numerical experiments suggest that if the warehouse's space utilization is low then the savings from optimizing the classes can be very substantial. We compare the RLR on the classes formed by the FB method with the TOS policy and the DOS policy on their respective optimized classes. We observe that the RLR under the FB method results in the lowest total travel cost across various parameter settings. Furthermore, a case study based on data from a third-party logistics provider also reveals that the RLR under the FB method outperforms the two traditional policies on their respective optimized classes. This shows the potential of our approach for practical use. We also observe and prove that if the warehouse is highly utilized, then the FB method only forms a single class. This provides a justification on why the random policy (with one class) is the most commonly used storage-retrieval policy in a unit-load warehouse in the upstream of a supply chain, where maximizing warehouse utilization is usually more important than reducing response time.

Our contributions fill the gap in the literature in the following ways:

1. Most papers in the literature consider a specific layout in which the receiving and the shipping docks coincide at a corner of the warehouse. In contrast, we optimize the classes for a layout with general receiving-dock and shipping-dock locations that may not coincide. Our class-formation method (the FB method) can also be applied to more general aisle orientations. Our paper generates insights into forming effective storage classes under more general warehouse settings.
2. By introducing the FB method, we show the additional value of considering (i) the difference between the store cost and the retrieve cost of each location, and (ii) the imbalance between the inflow and the outflow of each product.
3. Through a series of carefully designed numerical experiments and a study using real data from a

third-party logistics provider, we demonstrate the impact of optimizing classes on the performance of storage-retrieval policies.

This paper is organized as follows. After reviewing the related literature, we describe a model of the storage-retrieval problem in Section 3. Section 4 introduces the FB method to form classes. Section 5 illustrates the FB method through a small example. Section 6 compares numerically the performance of the RLR, the TOS, and the DOS policies on their respective optimized classes across various parameter settings. Section 7 demonstrates through a case study that the RLR under the FB method outperforms the two traditional storage-retrieval policies. Section 8 derives analytical results and managerial insights for some special cases. The paper concludes with remarks in Section 9.

2. Related literature

Heskett (1963, 1964) introduces a *dedicated storage policy* that determines the storage locations of pallets. Each storage location is reserved only for a specific product. The author associates each product with a cube-per-order index (COI), which is defined as the ratio of the product's allocated storage space to its demand rate. The author ranks products in increasing COI and assigns them sequentially to available locations with the smallest travel time. This policy is also known as a *full turnover policy* (de Koster et al. 2007, Roodbergen and Vis 2009) because the inverse of COI of a product is called the turnover rate of the product. Mallette and Francis (1972) consider multiple receiving and shipping docks. The authors show that the full turnover policy (or the COI policy) is optimal if all products have the same probability mass function for selecting a dock. Malmborg and Bhaskaran (1990) study the optimality of the full turnover policy for more complicated settings.

The dedicated storage policy does not efficiently utilize the space because empty locations cannot be used for other products when the inventory of a product is depleted. One way to overcome this problem is to use a *shared storage policy* that assigns an empty location to any product. For example, a *random policy* assigns a pallet randomly to any available location with equal probability. Hausman, Graves, and Schwarz (1976, 1977, 1978) propose a class-based turnover policy (abbreviated as TOS policy), which groups locations into several classes. The authors assign the product with the highest turnover rate to the class with the smallest average travel time. Within a class, a pallet is assigned randomly to any empty location. Rosenblatt and Eynan (1989) assume the receiving and shipping docks coincide at a corner of a square-in-time warehouse (that is, from the corner, the time to reach the most distant row equals the time to reach the most distant column). They develop a search procedure to numerically

find the optimal boundaries of n classes for the TOS policy. Eynan and Rosenblatt (1994) generalize the search procedure to determine the optimal class boundaries for a rectangular warehouse.

Kouvelis and Papanicolaou (1995) derive formulas for the optimal boundary of two classes in a rectangular warehouse. These formulas allow us to see the effects of various factors (such as the access frequencies of the two classes and the warehouse's dimensions) on the optimal class boundary. Yu et al. (2015) study the TOS policy for a finite number of products. They develop an algorithm to determine the optimal number and the boundaries of classes in a warehouse. They find that a small number of classes is usually optimal, and for a wide range of numbers of classes around this optimum, the total travel cost is similar. This finding is consistent with Guo et al. (2016) who consider various storage policies. Rao and Adil (2017) develop a travel distance model, and consider the full and class-based turnover policies with a hybrid product placement scheme. They study the policies' performance on a unit-load warehouse in a real-world setting. Zaerpour et al. (2017a) optimize the dimensions and the zone boundary of a two-class live-cube storage system to minimize its response time. Zaerpour et al. (2017b) derive the expected retrieval time of an arbitrary unit load in a live-cube system. They also optimize the system dimensions to minimize the retrieval time. It is worth noting that all the papers studying class formation above consider a specific warehouse layout, in which the receiving and shipping docks coincide at a corner of the warehouse. In contrast, we optimize the classes for a layout with general receiving-dock and shipping-dock locations that may not coincide in this paper. Furthermore, our method can handle layouts with multiple receiving and shipping docks such as the one studied by Gharehgozil et al. (2017) and Weidinger and Boysen (2018). Gharehgozil et al. (2017) develop an algorithm to sequence the storage and retrieval decisions that minimize the travel time in a layout with two shipping docks. Weidinger and Boysen (2018) propose a binary search-based heuristic to scatter products all around a mixed-shelves warehouse with multiple receiving entries.

Pohl et al. (2011) study the full turnover policy in non-traditional warehouse designs. They find that the best warehouse design parameters under the random policy also perform well under the full turnover policy. Thonemann and Brandeau (1998) consider stochastic demand with a stationary distribution, and derive the expected travel time for the random, full turnover, and class-based turnover policies.

A warehouse is perfectly balanced if, for every time period, the number of arriving pallets equals the number of departing pallets with an identical duration of stay. Goetschalckx and Ratliff (1990) study a *full duration-of-stay policy*, which assigns the pallet with the shortest duration of stay to the location with the smallest travel time. They prove that if a warehouse is perfectly balanced, the full duration-of-stay policy is an optimal shared storage policy. Goetschalckx and Ratliff (1990) also introduce the

class-based duration-of-stay policy (abbreviated as DOS policy), which sorts the pallets in increasing duration of stay and sequentially assigns the pallets with the shortest duration of stay to the class with the smallest average travel time. For a warehouse that is not perfectly balanced, they use simulations to compare the performance of different policies based on a deterministic model. Their results suggest that the full duration-of-stay policy outperforms the TOS policy with two classes, which in turn outperforms the DOS policy with two classes. Kulturel et al. (1999) perform simulations on a warehouse with three classes facing stochastic demands, and find that the TOS policy generally outperforms the DOS policy. This is consistent with the findings of Goetschalckx and Ratliff (1990) for a warehouse with two classes.

Ang et al. (2012) consider products with time-varying, deterministic arrivals and non-stationary, stochastic demands over multiple periods. Based on a robust optimization formulation, the authors determine the RLR to solve the storage and retrieval problem. They assume the classes are predetermined by a grid, and compare the RLR with other storage-retrieval policies based on the same class formation. Through extensive numerical experiments, they find that the RLR is significantly more efficient than the TOS and the DOS policies. Although the RLR is an efficient storage-retrieval policy, Ang et al. (2012) do not optimize the class formation. In contrast, we propose a method to optimize the storage classes for the RLR policy.

To improve a warehouse's performance, Gue and Meller (2009), Thomas and Meller (2014), Çelk and Süral (2014), Öztürkoğlu et al. (2014), and Öztürkoğlu et al. (2018) study different ways to configure aisles in the warehouse. In contrast, we propose a method to optimize the classes in a layout with general aisle orientations and with general receiving-dock and shipping-dock locations. We also study the impact of class-formation methods on the performance of the storage-retrieval policies. For excellent reviews of warehouse designs and operations, see Gu et al. (2007), de Koster et al. (2007), and Gu et al. (2010).

3. A deterministic model of the storage-retrieval problem

In this section, we present a model of the storage-retrieval problem with products having prescheduled arrivals and deterministic demands. We consider a unit-load warehouse with single command travel. We assume a single receiving dock and a single shipping dock located anywhere in the warehouse, and they may not coincide with each other. We focus on this simplest case, but our model can be generalized to warehouses with multiple receiving and shipping docks. For each storage location, we define its *store cost* (*retrieve cost*) as the travel distance to move from the receiving dock R (shipping

dock S) to the location and then return to R (S). We define the *location travel cost* of each location as the sum of its store and retrieve costs.

Assume the storage locations of the warehouse are grouped into N classes. Let s_j and r_j denote the average store cost and the average retrieve cost, respectively, of all locations in class j, for $j = 1, \ldots, N$. For the purpose of modelling, we ignore the differences of the locations in the same class. If a pallet is assigned to a class, it is stored at an arbitrary location in the class. Each class j has capacity c_j, which represents the number of locations in the class. We also consider a class $N+1$ that represents external emergency storage with infinite capacity ($c_{N+1} = \infty$) but incurs high average store and retrieve costs.

Suppose there are M products indexed by $i = 1, \ldots, M$. We divide the planning horizon into T periods indexed by $t = 1, \ldots, T$. For each period, we assume all pallets from suppliers arrive at the start of the period, and all pallets ordered by customers during the period are retrieved at the end of the period. The warehouse's goal is to minimize the total cost over the planning horizon. For convenience, let $\mathcal{M} = \{1, \ldots, M\}$, $\mathcal{N} = \{1, \ldots, N\}$, $\mathcal{N}^+ = \{1, \ldots, N+1\}$, $\mathcal{T} = \{1, \ldots, T\}$, and $\mathcal{T}^+ = \{1, \ldots, T+1\}$.

In this deterministic model, we assume that all information throughout the entire planning horizon is available at the start of the first period. Let a_i^t denote the number of pallets of product i arriving at the start of period t. Let v_{ij}^t be a decision variable determining the number of arriving pallets of product i that are assigned to class j in period t. Since all arriving pallets must be assigned to some classes, we have $\sum_{j \in \mathcal{N}^+} v_{ij}^t = a_i^t, \text{for } i \in \mathcal{M}, t \in \mathcal{T}$. Similarly, let d_i^t denote the number of pallets of product i that are ordered in period t. Let w_{ij}^t be a decision variable determining the number of pallets of product i that are retrieved from class j in period t. We have $\sum_{j \in \mathcal{N}^+} w_{ij}^t = d_i^t, \text{for } i \in \mathcal{M}, t \in \mathcal{T}$.

Let x_{ij}^t denote the number of pallets of product i in class j at the start of period t. No backorders are allowed. Thus, $x_{ij}^t \geq 0, \text{for } i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}^+$. The inventory of product i in class j at the start of period $t+1$ is $x_{ij}^{t+1} = x_{ij}^t + v_{ij}^t - w_{ij}^t, \text{for } i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$. Since the inventory in each class j must not exceed its capacity, we have the capacity constraints $\sum_{i \in \mathcal{M}} \left(x_{ij}^t + v_{ij}^t \right) \leq c_j, \text{for } j \in \mathcal{N}, t \in \mathcal{T}$.

Rightfully, the decision variables should be restricted to integers. However, in order to yield a tractable formulation, we relax the integrality constraints and formulate a linear optimization problem to minimize the total travel cost of the warehouse as follows:

$$Z_D = \min \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{N}^+} \left(s_j v_{ij}^t + r_j w_{ij}^t \right) \tag{1}$$

$$
\begin{aligned}
\text{s.t.} \quad & \sum_{j \in \mathcal{N}^+} v_{ij}^t = a_i^t, && i \in \mathcal{M}, t \in \mathcal{T}; \\
& \sum_{j \in \mathcal{N}^+} w_{ij}^t = d_i^t, && i \in \mathcal{M}, t \in \mathcal{T}; \\
& x_{ij}^{t+1} = x_{ij}^t + v_{ij}^t - w_{ij}^t, && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}; \\
& \sum_{i \in \mathcal{M}} \left(x_{ij}^t + v_{ij}^t \right) \leq c_j, && j \in \mathcal{N}, t \in \mathcal{T}; \\
& x_{ij}^t \geq 0, && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}^+; \\
& v_{ij}^t, w_{ij}^t \geq 0, && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}.
\end{aligned}
$$

We assume any shortage will be handled by the suppliers and this will not incur any cost to the warehouse. Thus, there is always sufficient inventory to meet demand for every period. Equivalently,

$$
\sum_{\tau=1}^{t} d_i^\tau \leq \sum_{j=1}^{N+1} x_{ij}^1 + \sum_{\tau=1}^{t} a_i^\tau, \qquad i \in \mathcal{M}, t \in \mathcal{T}. \tag{2}
$$

Equation (2) ensures that we can always find a feasible solution to Problem (1).

Solving Problem (1) gives us a storage-retrieval policy, where v_{ij}^t and w_{ij}^t represent the storage decisions and the retrieval decisions respectively. However, this model does not accommodate the randomness of the demands. Ang et al. (2012) extend Problem (1) to incorporate random demands, and derive a storage-retrieval policy, called the RLR, using a robust optimization model. Due to space limitation, we describe the robust optimization model and the RLR in Appendix A.

Ang et al. (2012) demonstrate through an extensive numerical study that the RLR outperforms the TOS and the DOS policies, under the assumption that the warehouse's storage locations are grouped into different classes based on a grid. We call this the *grid-based (GB) method*. It is unclear whether there exists a better way to form classes for the RLR. Furthermore, it is also unclear how the RLR performs compared to the TOS and the DOS policies when the classes are optimized for each policy. In the next section, we introduce a *novel* class-formation method for the RLR. We show numerically in Section 6 that the new class-formation method outperforms the GB method for the RLR. Our numerical study also suggests that the RLR under the new class-formation method continues to outperform the two traditional storage-retrieval policies even if their respective classes are optimized.

4. A class-formation method based on visit frequency for the RLR

In this section, we introduce a novel method to form storage classes based on the visit frequency of each location, which can be determined by solving a special case of Problem (1). We then use the

resultant visit frequencies to rank the locations before we group them into classes. It is worth noting that our method can handle a layout where the receiving and the shipping docks may not coincide and are arbitrarily located in the warehouse. It also uses the product flow information to form classes. In contrast, the traditional class-formation methods (Eynan and Rosenblatt 1994, Goetschalckx and Ratliff 1990) rank each storage location based on its location travel cost (the sum of its store and retrieve costs).

We assume the demands are random. Specifically, we consider a special case of the factor-based demand model used by Ang et al. (2012) in which the random demand $\tilde{d}_i^t$ for product i in period t depends linearly on a sequence of uncertain factors $\tilde{z}_i^k$, $k = 1, \cdots, t$. The uncertain factor $\tilde{z}_i^k$ is related to product i and is realized in period k. Let z_i^t denote the realized value of the uncertain factor $\tilde{z}_i^t$, for $i \in \mathcal{M}, t \in \mathcal{T}$. We assume $\tilde{z}_i^k$ has mean equal to 0 and a support set $W = \{z | -q \leq z \leq q\}$. We assume $\tilde{d}_i^t$ is independent of other products but depends on the past demands for product i up to period t. Specifically, the demand for product i in period t is defined as $\tilde{d}_i^t\left(\tilde{\boldsymbol{z}}^t\right) = d_i^t + \sum_{k=1}^t d_i^{t,k} \tilde{z}_i^k$, where d_i^t represents the *mean* demand for product i in period t, and $d_i^{t,k}$ are non-negative known coefficients with $\sum_{k=1}^t d_i^{t,k} = 1$. We set the parameter q in the range $\left[0, min_{i\in\mathcal{M}, t\in\mathcal{T}}\ d_i^t\right]$.

For each $t \in \mathcal{T}$, define $\tilde{\boldsymbol{\zeta}}^t = \left(\tilde{z}_1^t, \tilde{z}_2^t, \ldots, \tilde{z}_M^t\right)$ and $\tilde{\boldsymbol{z}}^t = \left(\tilde{\boldsymbol{\zeta}}^1, \tilde{\boldsymbol{\zeta}}^2, \ldots, \tilde{\boldsymbol{\zeta}}^t\right)$. For notational convenience, define $\tilde{\boldsymbol{z}}^0 = \mathbf{0}$, where $\mathbf{0}$ is an $M \times T$ dimensional vector with all entries equal to 0, and $\tilde{\boldsymbol{z}} = \tilde{\boldsymbol{z}}^T$. The uncertain factors $\tilde{\boldsymbol{z}}$ are random variables with unknown distributions. They lie in a full dimensional polytope support set $\boldsymbol{W}$. Similarly, define $\boldsymbol{\zeta}^t = \left(z_1^t, z_2^t, \ldots, z_M^t\right)$, $\boldsymbol{z}^t = \left(\boldsymbol{\zeta}^1, \boldsymbol{\zeta}^2, \ldots, \boldsymbol{\zeta}^t\right)$, for $t \in \mathcal{T}$, and define $\boldsymbol{z} = \boldsymbol{z}^T$. We assume there is sufficient initial inventory to meet the demand for product i in period t for all $\tilde{z}_i^t \in W, i \in \mathcal{M}, t \in \mathcal{T}$. That is,

$$\sum_{\tau=1}^{t} \tilde{d}_i^\tau\left(\tilde{\boldsymbol{z}}^\tau\right) \leq \sum_{j=1}^{N+1} x_{ij}^1 + \sum_{\tau=1}^{t} a_i^\tau, \qquad i \in \mathcal{M}, t \in \mathcal{T}. \tag{3}$$

Suppose each class j contains only a single location. Let L denote the number of storage locations in the warehouse (excluding the emergency storage). We have $N = L$ and $c_j = 1$, for $j = 1, \ldots, N$. For any realization $\boldsymbol{z}$ of the uncertain factors, Problem (1) can be parametrized as follows:

$$Z_D(\boldsymbol{z}) = \min \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} \sum_{j\in\mathcal{N}^+} \left(s_j v_{ij}^t + r_j w_{ij}^t\right) \tag{4}$$

$$
\begin{aligned}
\text{s.t.} \quad & \sum_{j\in\mathcal{N}^+} v_{ij}^t = a_i^t, && i\in\mathcal{M}, t\in\mathcal{T};\\
& \sum_{j\in\mathcal{N}^+} w_{ij}^t = d_i^t + \sum_{k=1}^{t} d_i^{t,k} z_i^k, && i\in\mathcal{M}, t\in\mathcal{T};\\
& x_{ij}^{t+1} = x_{ij}^t + v_{ij}^t - w_{ij}^t, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T};\\
& \sum_{i\in\mathcal{M}} \left(x_{ij}^t + v_{ij}^t\right) \le c_j, && j\in\mathcal{N}, t\in\mathcal{T};\\
& x_{ij}^t \ge 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}^+;\\
& v_{ij}^t, w_{ij}^t \ge 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}.
\end{aligned}
$$

After solving the parameterized, deterministic Problem (4) for a given $\boldsymbol{z}$, define the visit frequency of location $j = 1, \ldots, L$ as $f_j(\boldsymbol{z}) = f_j^s + f_j^r$, where $f_j^s = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} v_{ij}^t$ and $f_j^r = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} w_{ij}^t$. Note that in Problem (4), class $N+1$ $(= L+1)$ represents the emergency storage. Define the visit frequency to the emergency storage class as $f_{L+1}(\boldsymbol{z}) = f_{L+1}^s + f_{L+1}^r$, where $f_{L+1}^s = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} v_{i,L+1}^t$ and $f_{L+1}^r = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} w_{i,L+1}^t$. If we know the distributions of the uncertain factors $\tilde{\boldsymbol{z}}$, we can compute the expected visit frequency of each $j = 1, \ldots, L+1$ as

$$
E[f_j(\tilde{\boldsymbol{z}})] = \int_{\boldsymbol{z}\in\boldsymbol{W}} f_j(\boldsymbol{z}) g(\boldsymbol{z}) d\boldsymbol{z}, \tag{5}
$$

where $g(\boldsymbol{z})$ is the probability density function of $\boldsymbol{z}$. However, it is computationally expensive (if possible) to determine $E[f_j(\tilde{\boldsymbol{z}})]$ in Equation (5). We therefore approximate $E[f_j(\tilde{\boldsymbol{z}})]$ by considering the following three scenarios: (i) $\boldsymbol{z} = -\boldsymbol{q}$, (ii) $\boldsymbol{z} = \boldsymbol{0}$, and (iii) $\boldsymbol{z} = \boldsymbol{q}$, where $\boldsymbol{q}$ is an $M \times T$ dimensional vector with all its entries equal to q. Scenario (ii) corresponds to the case where the demand for each product in each period equals its mean value. Since $\sum_{j\in\mathcal{N}^+} f_j(\boldsymbol{z}) = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} \sum_{j\in\mathcal{N}^+} (v_{ij}^t + w_{ij}^t) = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} \left(a_i^t + d_i^t + \sum_{k=1}^t d_i^{t,k} z_i^k\right)$, scenarios (i) and (iii) correspond to the smallest value and the largest value, respectively, of the total visit frequency $\sum_{j\in\mathcal{N}^+} f_j(\boldsymbol{z})$. For $j = 1, \ldots, L+1$, we approximate Equation (5) as

$$
E[f_j(\tilde{\boldsymbol{z}})] \approx f_j := \frac{1}{3}\left(f_j(-\boldsymbol{q}) + f_j(\boldsymbol{0}) + f_j(\boldsymbol{q})\right). \tag{6}
$$

We call f_j the visit frequency of location j (except for f_{L+1}, which represents the visit frequency of the emergency storage class). Note that Equation (6) requires information of the lower bounds, the means, and the upper bounds of the demands. Equation (6) may result in a fractional value of f_j. For simplicity, we round f_j to the nearest integer. For all the locations $j = 1, \ldots, L$ in the warehouse, we group the locations with identical f_j into one class.

Thus, the resultant number of classes in the warehouse equals the number of distinct visit frequencies. Class 1 contains the locations with the highest visit frequency, class 2 contains the locations with the second-highest visit frequency, and so on. We summarize the procedure as follows:

1. Set $N = L$ in Problem (4) ($c_j = 1$, $j = 1, \ldots, L$).
2. Solve Problem (4) with $\boldsymbol{z} = -\boldsymbol{q}$, $\boldsymbol{0}$, and $\boldsymbol{q}$ to find $f_j(-\boldsymbol{q})$, $f_j(\boldsymbol{0})$, and $f_j(\boldsymbol{q})$, respectively, $j = 1, \ldots, L+1$. Compute the visit frequency f_j for $j = 1, \ldots, L+1$ using Equation (6).
3. For $j = 1, \ldots, L$, combine the locations with an identical visit frequency into one class. Let N_o denote the resultant number of classes (including the emergency storage).
4. Index the classes in the warehouse as $1, 2, \ldots, N_o - 1$ from the highest visit frequency to the lowest visit frequency. Let class N_o represent the emergency storage.

According to the above procedure, storage locations are grouped based on how frequently they are visited. The visit frequencies are determined by solving Problem (4), which captures the warehouse's layout and product flow information. The resultant number of classes N_o can be large, and may not be practical. Thus, we discuss a way to reduce the number of classes as follows.

Intuitively, a convenient class should be visited more frequently. This means that the sum of visit frequencies of all the locations within a convenient class should be large. We propose an algorithm to combine the classes based on the sum of visit frequencies of all locations in each class. Let Γ_j and γ_j denote the set of locations and the sum of their visit frequencies, respectively, of class $j = 1, \ldots, N_o$. To differentiate the combined classes derived by the algorithm from the original classes, we call each combined class a *cluster*. The algorithm combines the original N_o classes into $N'(\leq N_o)$ clusters. Let Ψ_k and ψ_k denote the set of locations and the sum of their visit frequencies, respectively, of cluster k.

Algorithm 1. (Reducing the number of classes)

Given Γ_j *and* γ_j, *for* $j = 1, \ldots, N_o$, *initialize* $\Psi_1 \leftarrow \Gamma_{N_o}$, $j' \leftarrow N_o - 1$, $k' \leftarrow 2$, *and* $\Psi_{k'} \leftarrow \emptyset$.

(1) Set $\Psi_{k'} \leftarrow \Psi_{k'} \cup \Gamma_{j'}$.

If $\psi_{k'} > \psi_{k'-1}$, *then set* $k' \leftarrow k' + 1$ *and* $\Psi_{k'} \leftarrow \emptyset$.

(2) Set $j' \leftarrow j' - 1$.

If $j' = 1$, *then set* $\Psi_{k'} \leftarrow \Psi_{k'} \cup \Gamma_1$ *and go to step 3.*

Otherwise, go to step 1.

(3) If $k' = 2$ *or* $\psi_{k'} > \psi_{k'-1}$, *then set* $N' \leftarrow k'$.

Otherwise, set $\Psi_{k'-1} \leftarrow \Psi_{k'-1} \cup \Psi_{k'}$ *and* $N' \leftarrow k' - 1$.

Reverse the indices of the clusters such that $\psi_1 > \psi_2 > \cdots > \psi_{N'}$. *Return* $\Psi_1, \Psi_2, \ldots, \Psi_{N'}$.

Step 1 of Algorithm 1 adds class j' to the current cluster k'. If cluster k' has a larger sum of visit frequencies than cluster $k'-1$, then a new cluster $k'+1$ is initiated. Step 2 proceeds to class $j'-1$ and sees if it is the only remaining class in the warehouse. Step 3 terminates the algorithm. Note that if $k' > 2$ and $\psi_{k'} \leq \psi_{k'-1}$, the second line of step 3 merges cluster k' with cluster $k'-1$. The third line of step 3 ensures that the clusters are indexed as 1 to N' from the largest sum to the smallest sum of visit frequencies. We show an example of how to reduce the number of classes below. Recall that f_j represents the visit frequency *per location* for class $j = 1, 2, \ldots, N_o - 1$, but represents the total visit frequency for the emergency class N_o.

Table 1: **The original classes and the combined classes by Algorithm 1**

Original classes					**Combined classes**			
Class	s_j	r_j	c_j	f_j	**Class**	s_j	r_j	c_j
1	264	264	16	8	1	264	264	16
2	391	391	11	6	2	419	419	20
3	434	434	2	4	3	664	664	60
4	459	459	7	2	4	5,000	5,000	∞
5	664	664	60	1				
6	5,000	5,000	∞	0				

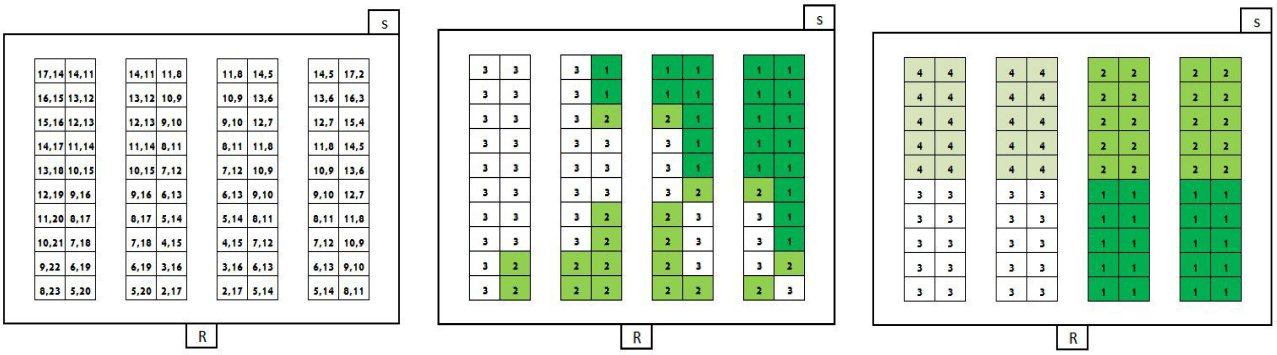
Example 1. Consider a warehouse with its classes shown on the left of Table 1. Class 6 represents the emergency storage. To initialize Algorithm 1, class 6 is assigned to cluster 1, that is, $\Psi_1 = \{\Gamma_6\}$ and $\psi_1 = f_6 = 0$. To form cluster 2, we assign class 5 to it. That is, $\Psi_2 = \{\Gamma_5\}$ and $\psi_2 = c_5 \times f_5 = 60$. We stop expanding cluster 2 because $\psi_2 > \psi_1$. To form cluster 3, we set $\Psi_3 = \{\Gamma_4\}$ and $\psi_3 = c_4 \times f_4 = 14$. Since $\psi_3 \leq \psi_2$, we add class 3 to cluster 3: $\Psi_3 = \{\Gamma_4, \Gamma_3\}$ and $\psi_3 = 14 + c_3 \times f_3 = 22$. We continue to expand cluster 3 until $\Psi_3 = \{\Gamma_4, \Gamma_3, \Gamma_2\}$ and $\psi_3 = 22 + c_2 \times f_2 = 88$. We stop expanding cluster 3 because $\psi_3 > \psi_2$. To form cluster 4, we set $\Psi_4 = \{\Gamma_1\}$ and $\psi_4 = c_1 \times f_1 = 128$. Since all the classes are assigned to some clusters, the algorithm terminates. Reversing the indices, we have $\Psi_1 = \{\Gamma_1\}$, $\Psi_2 = \{\Gamma_2, \Gamma_3, \Gamma_4\}$, $\Psi_3 = \{\Gamma_5\}$, and $\Psi_4 = \{\Gamma_6\}$. The resultant clusters are shown on the right of Table 1. Note that the average store and retrieve costs are recalculated for each combined class. □

To summarize, we estimate the visit frequency of each location by solving Problem (4). According to the visit frequencies, we group the locations into classes by Algorithm 1. We call the class-formation method described in this section the *frequency-based (FB) method*. Based on the classes formed, we solve Problem (A.5) in Appendix A to determine the RLR. We illustrate this procedure using an example in the next section.

5. A small example

In this section, we use a simple example to show the performance of the RLR under the FB and the GB class-formation methods. Through this example, we can see the effectiveness of the FB method compared to the more straightforward GB method. This shows the importance of optimizing the classes for a given storage-retrieval policy. We also compare the RLR under the FB method with the TOS and the DOS policies under their respective optimized class-formation methods.

We consider a warehouse with $L = 80$ locations in the layout shown in Figure 2(a). Note that the receiving and the shipping docks are arbitrarily located on opposite sides of the warehouse. The figure also shows the store cost (first entry) and the retrieve cost (second entry) of each location in the warehouse. We focus on a problem instance with only one product over one period. Assume $a_1^1 = 40$ and the demand for the product in the period has mean $d_1^1 = 25$ and support on $[21, 29]$ (that is, $q = 4$). Thus, the demand for the product equals $\tilde{d}_1^1\left(\tilde{\boldsymbol{z}}^1\right) = 25 + \tilde{z}_1^1$.



(a) (b) (c)

Figure 2: **A small example.** (a) Store and retrieve costs of each location. (b) Classes formed by the FB method. (c) Classes formed by the GB method.

To form the classes for the RLR, we consider the FB method and the GB method (see Ang et al. (2012) for the details). Figures 2(b) and 2(c) show the classes formed by the FB and the GB methods respectively. The FB method takes both the layout information and the product flow information into account when it forms classes. Given that there are 40 arriving pallets and 25 of them are expected to depart from the warehouse, the FB method forms class 1 containing 22 locations with a small sum of the average store and average retrieve costs (19.0). Furthermore, there are 15 pallets expected to remain in the warehouse. The FB method forms class 2 containing 19 locations with a low average store cost (5.5) but a high average retrieve cost (14.7) to accommodate these pallets. In contrast, the GB method ignores both the layout and the product flow information, and only groups the locations based on their proximity into four even classes shown in Figure 2(c).

The first two rows of Table 2 compare the performance of the RLR under the FB and the GB methods in detail. Columns 2 and 3 show the classes and their capacities, respectively, under each class-formation method. Based on the store and the retrieve costs, the number of arriving pallets (40), the mean demand (25), and the range of the uncertain factor ($q = 4$), the FB method forms classes 1, 2, and 3 with capacities 22, 19, and 39 respectively.

Based on the classes formed by the FB method (Figure 2(b)) and the GB method (Figure 2(c)), we compute the corresponding RLR by solving Problem (A.5). Assuming a realized demand of 29, we determine the storage and retrieval decisions in columns 6 and 7 respectively. The last column shows that the RLR under the FB method (with a total cost of 619.9) outperforms the RLR under the GB method (with a total cost of 652.0). This indicates that even with the same storage-retrieval policy, optimizing class formation can significantly reduce the travel cost.

Table 2: **Performance of different storage-retrieval policies under different class formations**

	Class j	**Capacity** c_j	**Costs** (s_j, r_j)	**Sum** $s_j + r_j$	**Storage** v_{1j}	**Retrieval** w_{1j}	**Storage Cost** $s_j v_{1j}$	**Retrieval Cost** $r_j w_{1j}$	**Total cost**
RLR under the FB method	1	22	(12.3, 6.7)	19.0	22	22	270.6	147.4	
	2	19	(5.5, 14.7)	20.2	18	7	99.0	102.9	
	3	39	(9.9, 14.7)	24.6	0	0	0	0	
	4	∞	(5,000, 5,000)	10,000	0	0	0	0	
							369.6	250.3	**619.9**
RLR under the GB method	1	20	(7.0, 12.0)	19.0	5	5	35.0	60.0	
	2	20	(12.0, 7.0)	19.0	20	20	240.0	140.0	
	3	20	(7.0, 18)	25.0	15	4	105.0	72.0	
	4	20	(12.0, 13.0)	25.0	0	0	0	0	
	5	∞	(5,000, 5,000)	10,000	0	0	0	0	
							380.0	272.0	**652.0**
TOS under the ER method	1	80	(9.5, 12.5)	22.0	40	29	380.0	362.5	
	2	∞	(5,000, 5,000)	10,000	0	0	0	0	
							380.0	362.5	**742.5**
DOS under the GR method	1	16	(13.0, 5.9)	18.9	16	16	208.0	94.4	
	2	64	(8.6, 14.1)	22.7	24	13	206.4	183.3	
	3	∞	(5,000, 5,000)	10,000	0	0	0	0	
							414.4	277.7	**692.1**

Table 2 also compares the RLR under the FB method with the other two storage-retrieval policies: the TOS and the DOS policies. For the TOS policy, we adapt the dynamic programming method in Eynan and Rosenblatt (1994) to form classes. We call this class-formation method the *ER method*. For the DOS policy, we use the adaptive algorithm on pages 1126–1127 in Goetschalckx and Ratliff (1990) to form classes. We call this the *GR method*. The last column of Table 2 shows that the RLR under the FB method significantly outperforms the TOS and the DOS policies with their respective optimized classes. Both the ER and GR class-formation methods *rank the locations based on their location travel costs*, before they group the locations into classes. Consequently, they result in larger total costs compared to the RLR under the FB method. We see similar results for other demand realizations.

By ranking the storage locations based on their visit frequencies, the FB method takes the difference

of the store cost and the retrieve cost of each location into account. Considering this delicacy allows the FB method to leverage the imbalance between the inflow and the outflow of each product. This is the main reason behind the superior performance of the RLR policy applied on the classes formed by the FB method. This demonstrates the additional value to the warehouse managers of considering more-detailed information: (i) the difference between the store and the retrieve costs of each location, and (ii) the imbalance between the inflow and the outflow of each product.

6. Numerical study

In this section, we conduct extensive numerical experiments to study the impact of different class-formation methods on the performance of storage-retrieval policies for various parameter settings. For each policy, we first form the classes and then determine the storage and retrieval decisions (see Section 5 for an example). Based on these decisions, we compute the total cost of each policy through simulations. Note that the FB method requires us to solve Problem (4), which is an LP model, to determine the visit frequency of each location. The LP model can be solved within minutes on a desktop computer using any commercial or open-source solver.

We consider three different layouts of the warehouse. Layout 1 is a U-flow layout (Bartholdi and Hackman 2016), in which the receiving and shipping docks coincide on one side of the warehouse. Layout 2 is a flow-through layout (Bartholdi and Hackman 2016), in which the receiving and shipping docks are located in the middle of opposite sides of the warehouse. In layout 3, the receiving dock and the shipping dock are located at arbitrary points of opposite sides of the warehouse.

In the factor-based demand model used in the simulations, we assume that for $i \in \mathcal{M}, t \in \mathcal{T}$, $d_i^{t,k} = 0.1/(t-1)$ for $k = 1, \ldots, t-1$ and $d_i^{t,k} = 0.9$ for $k = t$. We consider three different distributions for each $\tilde{z}_i^t, i \in \mathcal{M}, t \in \mathcal{T}$, in the simulations:

1. a truncated normal distribution $N\left(0, \sigma^2\right)$;
2. a scaled beta distribution, that is,

$$z = \begin{cases} (x - E[x])\, q/E[x], & \text{if } x \leq E[x]; \\ (x - E[x])\, q/\left(1 - E[x]\right), & \text{otherwise;} \end{cases}$$

   where x follows a beta distribution $Beta(a, b)$; and
3. a uniform distribution.

Since the demand is random, the storage and retrieval decisions of any policy may not be executed exactly. If the storage decisions do not match the actual number of arriving pallets, then we store any additional arriving pallets to the class with the next smallest index available. (For the FB method, a class with the next smallest index has a lower sum of visit frequencies. For the other class-formation methods, a class with the next smallest index has a larger sum of the average store and retrieve costs.) For example, if there are 15 pallets of a product arriving but the storage decision is to store 10 pallets to class 1, then we store 10 pallets to class 1 and 5 pallets to class 2 (if class 2 has available space). Similarly, if the retrieval decisions do not match the actual number of departing pallets, then we retrieve any additional departing pallets from the class with the smallest index available. For example, if the demand for a product is 15 pallets but the retrieval decision is to retrieve 10 pallets from class 3, then we retrieve the additional 5 pallets from class 1 (if the product is available there). This is to ensure the actual implementation to be as close as possible to the policy when facing the random demand.

Although we use the average store cost s_j and the average retrieve cost r_j of each class j to determine a storage-retrieval policy, we use the *actual location-to-location cost* to compute the total cost of the policy in each simulation. For example, if we store a pallet in period t to a location with store cost 76, then we record the cost as 76, even though the location may belong to a class with an average store cost of 100. This precisely determines the actual total cost of the policy. For each layout and each demand distribution, we perform 1,000 simulation runs to evaluate each policy to ensure that the standard error is within 1%. Each simulation run corresponds to one demand sample for all the products and periods.

In Section 6.1, we study the impact of class-formation methods on each storage-retrieval policy. In Section 6.2, we compare the RLR under the FB method, the TOS policy under the ER method, and the DOS policy under the GR method by varying (i) the warehouse utilization, (ii) the demand variability, (iii) the number of products, and (iv) the length of the planning horizon.

6.1. Impact of the class-formation methods under various utilization levels

We focus on the impact of the class-formation methods on each storage-retrieval policy under various levels of the warehouse's space utilization. To control the utilization, we scale the number of arriving pallets of each product in each period by a scaling factor α. Specifically, we set $a_i^t = \lfloor \alpha \underline{a}_i^t \rfloor$, where $\underline{a}_i^t$ represents the number of arriving pallets of product i in period t of a base problem instance. We create different problem instances by changing α. As α increases, the number of arriving pallets in each period increases and the utilization increases.

Define *average utilization* u based on the mean demands d_i^t as follows:

$$u = \frac{1}{T \times L} \sum_{t=1}^{T} \left[x^1 + \sum_{i=1}^{M} \left(\sum_{\tau=1}^{t-1} (a_i^\tau - d_i^\tau) + a_i^t \right) \right], \tag{7}$$

where x^1 represents the initial inventory of all the products in the warehouse. We set $x^1 = 0$ for all the experiments. The average utilization u indicates how populated the warehouse is based on the mean demands d_i^t. We set $M = 10$, $T = 6$, and $q = 2$. Table 3 shows different values of α and the corresponding average utilization used in our numerical experiments.

Table 3: **Different values of α and average utilization used in the numerical experiments**

α	1	2	3	4	5	6	7	8	9	10
u	0.33	0.47	0.63	0.85	1.00	1.52	1.96	2.55	3.08	3.55

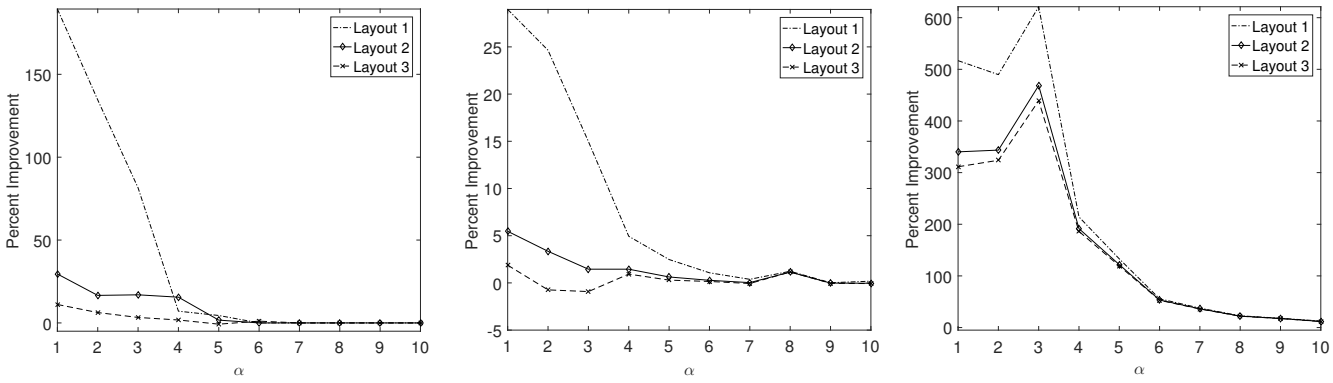
To see the effect of optimizing classes for each storage-retrieval policy, we benchmark the total cost of a policy under its optimized class-formation method against its total cost under the GB method. Let $\mathcal{P}_m$ denote the average total cost using the storage-retrieval policy $\mathcal{P}$ under the class-formation method m over the simulation runs. For example, RLR_{FB} and DOS_{GB} denote the average total costs of the RLR under the FB method and the DOS policy under the GB method respectively. For the RLR, TOS, and DOS policies, define the *percent improvement* of using an optimized class-formation method over the GB method, respectively, as follows:

$$\frac{RLR_{GB} - RLR_{FB}}{RLR_{FB}} \times 100\%, \quad \frac{TOS_{GB} - TOS_{ER}}{TOS_{ER}} \times 100\%, \quad \textit{and} \quad \frac{DOS_{GB} - DOS_{GR}}{DOS_{GR}} \times 100\%.$$

To obtain the number of classes (excluding emergency storage) for the GB method for each policy, we set it equal to the number of classes in the warehouse under the corresponding optimized class-formation method if it is even. Otherwise, we round it up to the nearest even number to ensure a fair comparison.

Figure 3 shows that the percent improvement is generally positive when $\alpha < 5$ ($u < 1.00$) under a truncated normal distribution for all the three layouts. This suggests that optimizing classes can improve a storage-retrieval policy's performance when the warehouse's utilization is low. The percent improvement is especially high for the RLR and DOS policies. However, as α increases (the utilization increases) the percent improvement generally decreases. This suggests that the impact of optimizing classes reduces as the warehouse gets more utilized. We observe similar results for the other two demand distributions. To save space, we only present the results under the truncated normal distribution.

If the warehouse's utilization is low the emergency storage is hardly used, and there is a clear advantage of optimizing classes for each policy as shown in Figure 3. However, as the utilization



(a) RLR_{FB} vs RLR_{GB} (b) TOS_{ER} vs TOS_{GB} (c) DOS_{GR} vs DOS_{GB}

Figure 3: **Percent improvement of using an optimized class-formation method over the GB method for various values of α under each policy**

increases, the emergency storage is used more frequently. This tremendously raises the total cost, which causes the savings by any optimized class formation to be relatively negligible. Thus, a proper estimate of the warehouse's utilization is necessary to determine the importance of optimizing classes. *If the warehouse's utilization is low, then it is important to optimize the storage classes.*

Note that in Figure 3 although the percent improvement of DOS_{GR} over DOS_{GB} can be larger than the percent improvement of RLR_{FB} over RLR_{GB}, RLR_{FB} actually outperforms DOS_{GR} (Section 6.2).

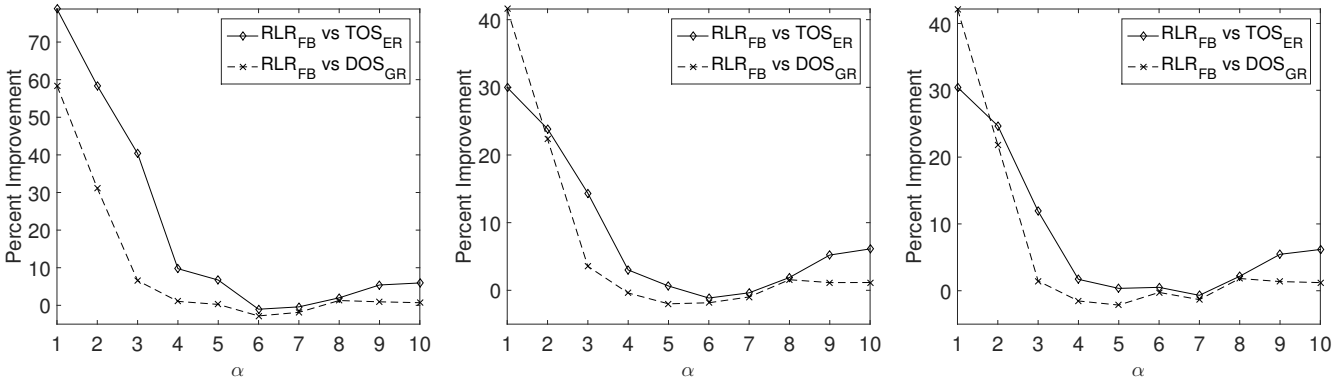
6.2. Comparing the different storage-retrieval policies on their respective optimized classes

We also compare the RLR with the TOS and DOS policies under their respective optimized class-formation methods by varying (i) the warehouse utilization, (ii) the demand variability, (iii) the number of products, and (iv) the length of the planning horizon. We define the percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR}, respectively, as follows:

$$\frac{TOS_{ER} - RLR_{FB}}{RLR_{FB}} \times 100\% \quad and \quad \frac{DOS_{GR} - RLR_{FB}}{RLR_{FB}} \times 100\%. \tag{8}$$

6.2.1. Performance under different utilization levels

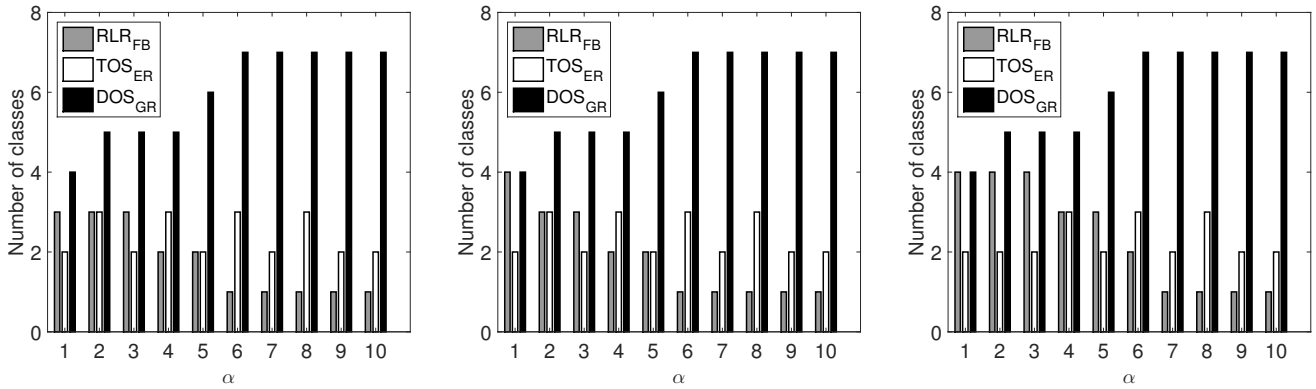
Figure 4 shows the percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR} defined in Equations (8). We set $M = 10$, $T = 6$, and $q = 2$. The RLR under the FB method significantly outperforms the other two policies for $\alpha < 5$, suggesting that the former leads to the lowest cost when the average utilization u of the warehouse is below 1.00. On the other hand, if $5 \leq \alpha \leq 8$ ($1.00 \leq u \leq 2.55$) then all the policies with their optimized classes perform similarly. When $\alpha > 8$ ($u > 2.55$), the RLR under the FB method outperforms the TOS and DOS policies again by about 5.5% and 1.5% respectively.



(a) Layout 1 (b) Layout 2 (c) Layout 3

Figure 4: **Percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR} for various values of α**

Figure 5 shows the number of classes (excluding the emergency class) for each storage-retrieval policy under different values of α. It is worth noting that the number of classes for the RLR is no larger than that for the DOS policy for all the values of α. Compared with the TOS policy, the RLR has more classes when the warehouse utilization is low, but has fewer classes when the utilization is high. The RLR requires no more than 4 classes and the number of classes drops to 1 as the utilization increases. This is promising as the low number of classes for the RLR eases the implementation in practice.



(a) Layout 1 (b) Layout 2 (c) Layout 3

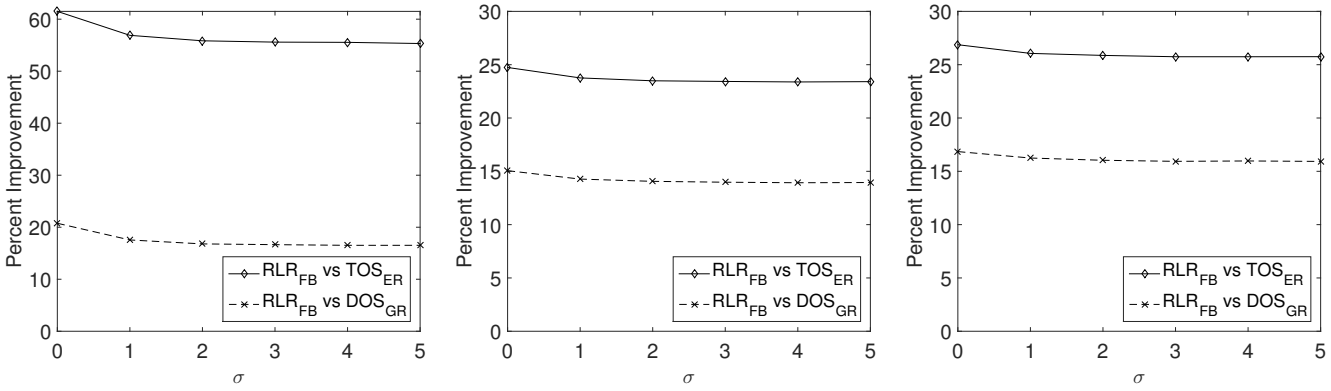
Figure 5: **The number of classes for each policy for various values of α**

It is interesting to note that for $\alpha > 6$, the FB method uses only one class for all the layouts. When there is only one class, each pallet will be stored at any arbitrary location in the warehouse. This makes the implementation of the RLR in practice especially simple. Surprisingly, this does not significantly deteriorate the effectiveness of the RLR in this range of α as it still outperforms the TOS and DOS policies (see Figure 4), which require more classes (see Figure 5). As discussed at the end of Section 6.1, under high utilization optimizing classes does not have a big impact on a storage-retrieval policy's

performance. The FB method is aligned with this insight because it only forms one class for the RLR when the utilization is high ($\alpha > 6$). In fact, we prove in Section 8.1 that when the utilization is extremely high, the FB method always forms a single class in the warehouse.

6.2.2. Performance under various levels of demand variability

We study the impact of demand variability on the performance of the three storage-retrieval policies. We set $M = 10$, $T = 6$, and $\alpha = 2$. Figure 6 shows the performance of the policies when $\tilde{z}_i^t$ follow a truncated normal distribution with $q = 2$ for various σ, for all $i \in \mathcal{M}, t \in \mathcal{T}$. Our results indicate that RLR_{FB} is lower than TOS_{ER} by at least 24% and is lower than DOS_{GR} by at least 14% for a wide range of σ. This suggests that the RLR under the FB method is robust against demand variability.



(a) Layout 1 (b) Layout 2 (c) Layout 3

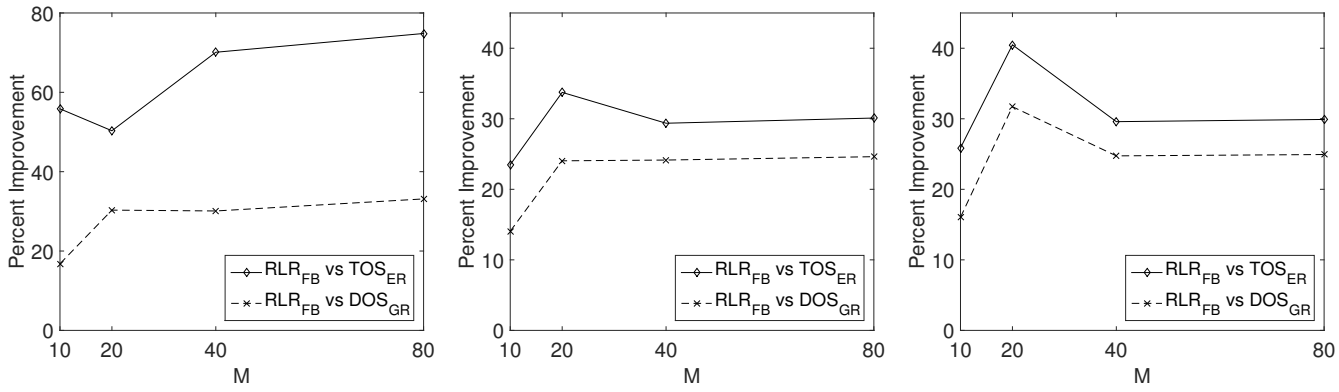
Figure 6: **Percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR} for various values of σ**

6.2.3. Performance under various numbers of products

We set $T = 6$, $q = 2$, $\alpha = 2$, and compare the three policies by varying the number of products M from 10 to 80. To maintain the same utilization, we increase the warehouse's capacity as the number of products increases. We expand the storage space in the vertical direction (for example, using pallet racks) such that we have more storage locations per unit floor area. In this manner, we retain the warehouse's layout (the relative positions of the receiving and the shipping docks) while keeping the utilization constant. Figure 7 shows that RLR_{FB} is lower than TOS_{ER} and DOS_{GR} by at least 22% and 15% respectively.

6.2.4. Performance under various lengths of the planning horizon

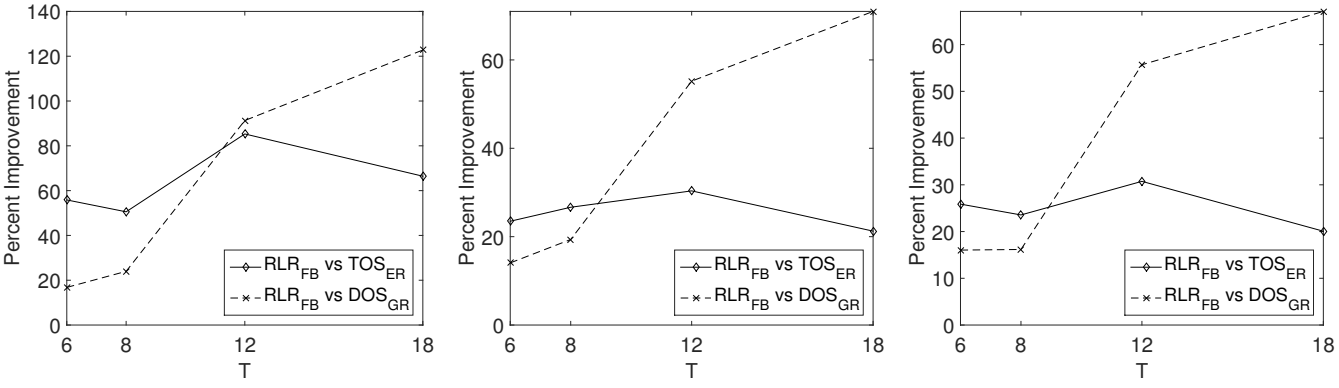
To see the impact of the length of the planning horizon, we set $M = 10$, $q = 2$, $\alpha = 2$, and vary T from 6 to 18 to compare the performance of the three policies. Figure 8 shows the percent improvement



(a) Layout 1 (b) Layout 2 (c) Layout 3

Figure 7: **Percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR} for various values of M**

of RLR_{FB} over TOS_{ER} and DOS_{GR}. We see that the RLR under the FB method outperforms the other two policies by at least 15%. It is worth noting that the performance of the DOS policy drops significantly as T increases. This is because the DOS policy relies on the information of the durations of stay of pallets, which becomes less reliable as the planning horizon gets longer.



(a) Layout 1 (b) Layout 2 (c) Layout 3

Figure 8: **Percent improvement of RLR_{FB} over TOS_{ER} and DOS_{GR} for various values of T**

7. A case study

In this section, we evaluate the policies in a case study with a third-party logistics provider. The company owns a unit-load warehouse (see Figure A.10 in Appendix A) that provides storage services for its client. When the warehouse runs out of storage locations, the company rents an emergency facility with large storage capacity that holds sufficient inventory for all products in all periods, but incurs high store and retrieve costs. The client makes all the planning and production decisions such

as demand forecasting and production scheduling. The storage and delivery of products are managed by the logistics provider, which operates in two shifts per day. Products arrive and are stored during the day shift (between 8:00AM and 5:30PM). All customer orders arriving during the day shift are retrieved in the following night shift (between 8:00 PM and 6:00 AM). Thus, the single-command travel assumption is valid under this setting.

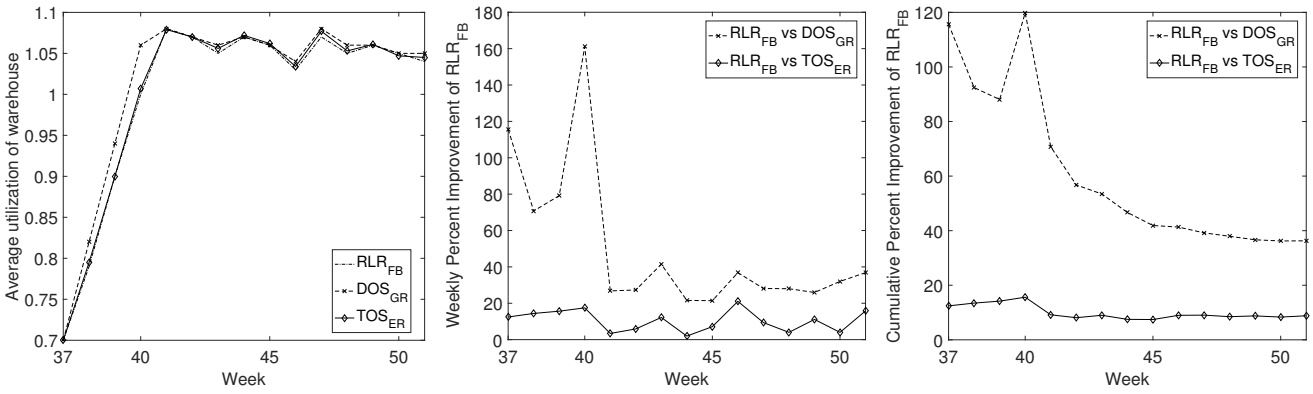
Each period in our model corresponds to a day in this case study. We have collected the actual numbers of arriving and departing pallets of each product in each day for 51 weeks. Each week consists of 6 working days. There are 410 products, where the top 10 products contribute about 28 percent of the annual total demand, and the top 122 products contribute about 80 percent of the annual total demand. There is no significant correlation between different products. The demand for product i in period t is represented as $\tilde{d}_i^t\left(\tilde{\boldsymbol{z}}_i^t\right) = d_i^t + \sum_{k=1}^{t} d_i^{t,k}\tilde{z}_i^k$, where d_i^t is the sample mean, $d_i^{t,k} = 0.1/(t-1)$ for $k = 1, \ldots, t-1$, and $d_i^{t,k} = 0.9$ for $k = t$. We assume that $\tilde{z}_i^t$ falls in the range $[\max\{-d_i^t, -3\sigma_i^t\}, 3\sigma_i^t]$, and σ_i^t is the sample standard deviation of the demand for product i in period t.

To evaluate the various storage-retrieval policies, we need to determine the means and the standard deviations of the products. We use the first 36 weeks of data to estimate these parameters, which will be used to evaluate the policies for week 37. Using a rolling-horizon principle, the demand parameters for week 38 is estimated based on the data from week 2 to week 37, and so on.

In practice, we do not change the classes often (say, every planning horizon). In this case study, we use the first 36 weeks of data to form classes for each policy. This results in two classes, two classes, and five classes for the RLR, the TOS, and the DOS policies respectively (excluding the emergency class). For week 37, we evaluate the three policies under their respective class formations. Subsequently, we reevaluate each policy for week 38 to week 51 with the same class formation.

Under their respective class formations, different policies result in different initial inventory levels for the subsequent week. This yields different average utilization u of the warehouse due to Equation (7), as shown in Figure 9(a) for week 38 to week 51. Figures 9(b) and 9(c) show the weekly percent improvement and the cumulative percent improvement, respectively, of the RLR with the FB method over the other two policies. The RLR with the FB method has the best weekly performance across the weeks and yields the lowest cumulative cost for the rest of the year.

When the average utilization u is less than 1 for all the policies (from week 37 to week 39), the RLR outperforms the TOS policy and the DOS policy by at least 12.5% and 87.9% respectively. When the average utilization u exceeds 1 for all the policies (from week 40 onwards), the RLR outperforms the TOS policy and the DOS policy by at least 7.4% and 36.3% respectively.



(a) (b) (c)

Figure 9: **Results of a case study.** (a) Average utilization of the warehouse. (b) Weekly percent improvement of RLR_{FB}. (c) Cumulative percent improvement of RLR_{FB}.

One explanation on the shortcoming of the TOS policy is that it does not factor the variable arrivals and demands into its formation of classes and storage-retrieval decisions. In contrast, the RLR with the FB method considers the variable arrivals, demands, and demand support sets in its computation. The performance of the DOS policy is the worst among the three policies (see Figures 9(b) and 9(c)), suggesting that the locations with low costs are not properly utilized and the emergency class is used excessively. One explanation of this behavior is that the DOS policy depends on the durations of stay of individual pallets (of different products). This information is difficult to estimate accurately in practice (for example, we use the first 36 weeks of data to predict the duration of stay of each individual pallet for week 37). This inaccuracy of the parameters causes the DOS policy to perform badly. In contrast, the RLR policy with the FB method can better absorb the variability of the arrivals and demands. The results in this case study suggest that the RLR with the FB method also outperforms the TOS and the DOS policies in a practical setting.

8. Special cases

In this section, we assume each class contains only one location so that $N = L$ and $c_j = 1$, for $j = 1, \ldots, L$. In Section 8.1, we show that the FB method (specifically, Algorithm 1) will group all the locations in the warehouse into only one class (excluding the emergency storage) if the warehouse is over utilized. In Section 8.2, we discuss a special case where the store cost is equal to the retrieve cost for each storage location in the warehouse. In Section 8.3, we consider deterministic demands and assume the warehouse faces symmetric arrivals and demands. In the latter two special cases, we show that ranking locations by visit frequency is equivalent to ranking by location travel cost found in the

literature. This implies that the traditional cost-based ranking is a special case of our frequency-based ranking.

8.1. An over-utilized warehouse

We consider a special case where the warehouse is over utilized: The number of arriving pallets is much larger than the number of departing pallets. In this situation, we show that the FB method produces only one class in the warehouse besides the emergency class (that is, Algorithm 1 results in $N' = 2$). This suggests that treating the entire warehouse as a single class is sufficient to guarantee efficiency. Recall that we can determine the visit frequency $f_j(\boldsymbol{z})$, $j = 1, \ldots, L$, for any given realization $\boldsymbol{z}$ by solving Problem (4). The following lemma characterizes $f_j(\boldsymbol{z})$ for an over-utilized warehouse. All proofs can be found in the online supplement.

Lemma 1. *Suppose Equation (3) holds. For any realization $\boldsymbol{z} \in \boldsymbol{W}$, if $a_i^t \to \infty$ for some $i \in \mathcal{M}$, $t \in \mathcal{T}$, then $f_j(\boldsymbol{z}) < \infty$, $j = 1, \ldots, N$, and $f_{N+1}(\boldsymbol{z}) \to \infty$.*

Corollary 1. *Suppose Equation (3) and the condition in Lemma 1 hold, then Algorithm 1 results in $N' = 2$ classes.*

Corollary 1 leads to an interesting insight: To ensure efficiency, it is sufficient to group all the locations of the entire warehouse into a single class if the warehouse is over utilized. This echoes the observations made in Sections 6.2.1, where the RLR under the FB method with only one class (excluding the emergency storage) outperforms the other policies. Corollary 1 also justifies why the random policy (with one class) is the most commonly used storage-retrieval policy in a unit-load warehouse in the upstream of a supply chain, where maximizing warehouse utilization is usually more important than reducing response time.

8.2. Symmetric store and retrieve costs

We consider a case where the receiving dock and the shipping dock are located at the same position in the warehouse such that the average store cost s_j is equal to the average retrieve cost r_j for all $j \in \mathcal{N}^+$. An example in which the receiving and shipping docks coincide at a corner of the warehouse is widely studied in the literature (Hausman et al. 1976, Rosenblatt and Eynan 1989, Eynan and Rosenblatt 1994, Kouvelis and Papanicolaou 1995, Yu et al. 2015).

Under this setting, we can simplify the objective function of Problem (4) as

$$\sum_{j \in \mathcal{N}^+} s_j \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} \left(v_{ij}^t + w_{ij}^t \right) \quad = \quad \sum_{j \in \mathcal{N}^+} s_j f_j(\boldsymbol{z}). \tag{9}$$

The following lemma shows that if $s_j = r_j$, for all $j \in \mathcal{N}^+$, and the warehouse is initially empty, then the location with the highest visit frequency also has the smallest location travel cost. This implies that ranking locations by visit frequency is equivalent to ranking them by location travel cost.

Lemma 2. *Suppose $N = L$, $s_j = r_j$, and $x_{ij}^1 = 0$, for all $i \in \mathcal{M}, j \in \mathcal{N}^+$. Given any realization $\boldsymbol{z}$ of the uncertain factors, if $s_{j'} \leq s_{j''}$ then $f_{j'}(\boldsymbol{z}) \geq f_{j''}(\boldsymbol{z})$, for $j', j'' \in \mathcal{N}$, where $f_j(\boldsymbol{z}) = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} \left(v_{ij}^t + w_{ij}^t\right)$ and v_{ij}^t, w_{ij}^t, $i \in \mathcal{M}, j \in \mathcal{N}, t \in \mathcal{T}$, are determined by solving Problem (4).*

Corollary 2. *If $N = L$ and the store cost is equal to the retrieve cost for every location in a warehouse with no initial inventory, then ranking the locations by visit frequency and ranking them by location travel cost are equivalent for the warehouse.*

8.3. Symmetric arrivals and demands

In this section, we consider a case where the demands are deterministic (this corresponds to a special case with $q = 0$ in Section 4). Suppose all the pallets of all the products have the same duration of stay $p \in [1, T]$. For the pallets that are retrieved at the end of period T (the end of the planning horizon), they all arrive at the start of period $T_p = T - p + 1$. Pallets that arrive after period T_p stay in the warehouse after period T. For convenience, let $\mathcal{T}_p = \{1, 2, , \ldots, T_p\}$. We have the following results.

Lemma 3. *Suppose all pallets have the same duration of stay $p \in [1, T]$. If $N = L$, $q = 0$, and $x_{ij}^1 = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+$, then we have $v_{ij}^t = w_{ij}^{t+p-1}$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}_p$.*

Lemma 4. *Suppose all pallets have the same duration of stay $p \in [1, T]$. If $N = L$, $q = 0$, and $x_{ij}^1 = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+$, then for each $j \in \mathcal{N}^+$, we have $f_j^s \geq f_j^r$.*

We now consider a special case where the warehouse faces symmetric arrivals and demands. We say a warehouse is *balanced* if $q = 0$ and $a_i^t = d_i^t$, for all $i \in \mathcal{M}, t \in \mathcal{T}$. This is equivalent to a case where all the pallets have a duration of stay $p = 1$. Lemma 5 shows that if a warehouse is balanced and is initially empty, then the number of pallets of product i stored to class j is equal to the number of pallets of product i retrieved from class j at every $t \in \mathcal{T}$. This leads to Lemma 6, which in turn implies that ranking the locations by visit frequency is equivalent to ranking them by location travel cost.

Lemma 5. *If $N = L$, $q = 0$, $a_i^t = d_i^t$, and $x_{ij}^1 = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$, we have the following results:*

(i) *$v_{ij}^t = w_{ij}^t$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$.*

(ii) $x_{ij}^t = 0$, *for* $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}^+$.

Lemma 6. *If* $N = L$, $q = 0$, $a_i^t = d_i^t$, *and* $x_{ij}^1 = 0$, *for* $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$, *then for each* $j \in \mathcal{N}^+$, *we have* $f_j^s = f_j^r$, *where* f_j^s *can be determined by the following optimization problem:*

$$\begin{aligned} \min \quad & \sum_{j \in \mathcal{N}^+} (s_j + r_j) f_j^s \\ \text{s.t.} \quad & \sum_{j \in \mathcal{N}^+} f_j^s = \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} a_i^t; \\ & f_j^s \le T, \qquad j \in \mathcal{N}; \\ & f_j^s \ge 0, \qquad j \in \mathcal{N}^+. \end{aligned} \tag{10}$$

Note that the inequality $f_j^s \ge f_j^r, j \in \mathcal{N}^+$, in Lemma 4 is tight for $p = 1$. Lemma 6 shows that for a balanced warehouse with no initial inventory, the visit frequency of each location can be determined by solving Problem (10). This implies the following corollary.

Corollary 3. *For a balanced warehouse with* $N = L$ *and no initial inventory, ranking the locations by visit frequency and ranking them by location travel cost are equivalent.*

It is worth noting that by taking both the layout information and the product flow information into account, the method of ranking the locations by visit frequency considers the *difference* between the store cost and the retrieve cost of each location, as well as the *imbalance* between the arrivals and the departures of each product. However, Corollaries 2 and 3 imply that if the store cost and the retrieve cost are symmetric for each location or if the arrivals and the departures are balanced for each product, then ranking the locations by visit frequency is equivalent to ranking them by location travel cost. Thus, the traditional cost-based ranking turns out to be a special case of our frequency-based ranking. The frequency-based ranking is suitable for more general settings with (i) asymmetric store and retrieve costs for each location and (ii) imbalanced inflow and outflow for each product.

9. Conclusion

Most papers in the literature study the class-formation problem in a unit-load warehouse for a specific layout where the receiving and the shipping docks coincide at a corner of the warehouse. A common approach is to rank storage locations according to their location travel costs before they are grouped into classes. In this paper, we introduce a new approach to optimize classes for general receiving-dock and shipping-dock locations that may not coincide. We propose the FB method that first ranks the storage

locations based on their visit frequencies and then groups them into classes using Algorithm 1. This method considers not only the warehouse's layout information, but also the product flow information. It bears similarity with the traditional way of ranking locations based on their location travel costs if the receiving and shipping docks coincide (see Corollary 2), or if the arrivals and the demands of each product are balanced (see Corollary 3). We use the FB method to form classes for the RLR, which is a storage-retrieval policy based on robust optimization. We benchmark the RLR under the FB method against the traditional TOS and DOS policies with their respective optimized classes.

Our numerical experiments suggest that if the warehouse's space is highly utilized, the class-formation method does not significantly affect the performance of all the three storage-retrieval policies. This is because when the utilization is high, the warehouse gets filled up and the emergency storage is frequently used. This tremendously raises the total travel cost, and the savings by any class-formation method becomes negligible. We also observe that the FB method yields only one class in the warehouse if the utilization is sufficiently high, and we analytically prove this observation in Corollary 1. This finding explains why the random policy (with one class) is the most commonly used storage-retrieval policy in a unit-load warehouse in the upstream of a supply chain, which are usually heavily utilized.

On the other hand, if the warehouse utilization is low ($\alpha < 5$ and $u < 1$), the emergency storage is hardly used and different class-formation methods may result in very different total travel costs. *Thus, the importance of optimizing classes depends on the utilization of the warehouse. Warehouse managers should first assess their warehouses' utilization. If the utilization is low then the managers should not neglect the cost effect of optimizing the storage classes.*

For low warehouse utilization (such as $\alpha = 2$), we evaluate the three policies for various levels of demand variability, numbers of products, and lengths of the planning horizon under different demand distributions and different layouts. The RLR outperforms the TOS policy (by up to 80%) and the DOS policy (by up to 125%). This manifests the value of considering both the warehouse's layout and the products' arrivals and demands in the FB method for forming classes, which improves the overall performance of the RLR. We also evaluate the policies using data from a third-party logistics provider. Our results suggest that the weekly performance of the RLR under the FB method is better than the TOS policy and the DOS policy, by an average of 8% and 51% respectively. By ranking the storage locations based on their visit frequencies, the FB method takes the difference of the store cost and the retrieve cost of each location into account (see the objective function of Problem (4)). Considering this delicacy allows the FB method to leverage the imbalance between the inflow and the outflow of each product. This is the main reason behind the superior performance of the FB method. *Our paper*

shows the additional value to the warehouse managers of considering more-detailed information: (i) the difference between the store cost and the retrieve cost of each location, and (ii) the imbalance between the inflow and the outflow of each product. One possible future research direction is to incorporate these two pieces of information into the traditional class-formation methods, such as the ER method for the TOS policy and the GR method for the DOS policy.

Our model is restricted to single-command travel for unit-load warehouses. It would be interesting to develop class-formation methods for warehouses with dual-command travel or with more complex order-picking operations (such as case picking or piece picking).

Compared to the dynamic program in the ER method (Eynan and Rosenblatt 1994), the FB method is more computationally efficient as it requires only solving a linear program (Problem (4)). We believe this requirement is quite reasonable because managers do not need to optimize classes frequently, and the savings from effective class formation can be very substantial.

References

Ang, M., Y.F. Lim, M. Sim. 2012. Robust storage assignment in unit-load warehouses. *Management Sci.* **58**(11) 2114–2130.

Bartholdi, J.J. III, S.T. Hackman. 2016. *Warehouse and Distribution Science.* `http://www.warehouse-science.com/`.

Çelk, M., H. Süral. 2014. Order picking under random and turnover-based storage policies in fishbone aisle warehouses. *IIE Trans.* **46**(3) 283–300.

de Koster, R., T. Le-Duc, K.J. Roodbergen. 2007. Design and control of warehouse order-picking: A literature review. *Eur. J. Oper. Res.* **182** 481–501.

Eynan, A., M.J. Rosenblatt. 1994. Establishing zones in single-command class-based rectangular AS/RS. *IIE Trans.* **26**(1) 38–46.

Gharehgozil, A.H., Y. Yu, X. Zhang, R. de Koster. 2017. Polynomial time algorithms to minimize total travel time in a two-depot automated storage/retrieval system. *Transportation Sci.* **51**(1) 19–33.

Goetschalckx, M., H.D. Ratliff. 1990. Shared storage policies based on the duration stay of unit loads. *Management Sci.* **36**(9) 1120–1132.

Graves, S.C., W.H. Hausman, L.B. Schwarz. 1977. Storage-retrieval interleaving in automatic warehousing systems. *Management Sci.* **23**(9) 935–945.

Gu, J., M. Goetschalckx, L.F. McGinnis. 2007. Research on warehouse operation: A comprehensive review. *Eur. J. Oper. Res.* **177** 1–21.

Gu, J., M. Goetschalckx, L.F. McGinnis. 2010. Research on warehouse design and performance evaluation: A comprehensive review. *Eur. J. Oper. Res.* **203** 539–549.

Gue. K.R., R.D. Meller. 2009. Aisle configurations for unit-load warehouses. *IIE Trans.* **41**(3) 171–182.

Guo. X., Y. Yu, R. de Koster. 2016. Impact of required storage space on storage policy performance in a unit-load warehouse. *Int. J. Prod. Res.* **54**(8) 2405–2418.

Hausman, W.H., L.B. Schwarz, S.C. Graves. 1976. Optimal storage assignment in automatic warehousing systems. *Management Sci.* **22**(6) 629–638.

Heskett, J.L. 1963. Cube-per-order index: A key to warehouse stock location. *Transportation and Distribution Management* **3** 27–31.

Heskett, J.L. 1964. Putting the cube-per-order index to work in warehouse layout. *Transportation and Distribution Management* **4** 23–30.

Kouvelis, P., V. Papanicolaou. 1995. Expected travel time and optimal boundary formulas for a two-class-based automated storage/retrieval system. *Int. J. Prod. Res.* **33**(10) 2889–2905.

Kulturel, S., N.E. Ozdemirel, C. Sepil, Z. Bozkurt. 1999. Experimental investigation of shared storage assignment policies in automated storage/retrieval systems. *IIE Trans.* **31** 739–749.

Mallette, A.J., R.L. Francis. 1972. A generalized assignment approach to optimal facility layout. *AIIE Trans.* **4**(2) 144–147.

Malmborg, C.J., K. Bhaskaran. 1990. A revised proof of optimality for the cube-per-order index rule for stored item location. *Appl. Math. Modelling* **14** 87–95.

Öztürkoğlu, Ö., D. Hoser. 2018 A discrete cross aisle design model for order-picking warehouses. *Eur. J. Oper. Res.*, forthcoming.

Öztürkoğlu, Ö., K.R. Gue, R.D. Meller. 2014. A constructive aisle design model for unit-load warehouses with multiple pickup and deposit points. *Eur. J. of Oper. Res.* **236** 382–394.

Pohl, L.M., R.D. Meller, K.R. Gue. 2011. Turnover-based storage in non-traditional unit-load warehouse designs. *IIE Trans.* **43**(10) 703–721.

Rao, S.S., G.K. Adil. 2017. Analytical models for a new turnover-based hybrid storage policy in unit-load warehouses. *Int. J. Prod. Res.* **55**(2) 327–346.

Roodbergen, K.J., I.F.A. Vis. 2009. A survey of literature on automated storage and retrieval systems. *Eur. J. Oper. Res.* **194** 343–362.

Rosenblatt, M.J., A. Eynan. 1989. Deriving the optimal boundaries for class-based automatic storage/retrieval systems. *Management Sci.* **35**(12) 1519–1524.

Sahni, S. 1975. Approximate algorithms for the 0/1 knapsack problem. *Journal of The Association for Computing Machinery* **22** 115–124.

Schwarz, L.B., S.C. Graves, W.H. Hausman. 1978. Scheduling policies for automatic warehousing systems: Simulation results. *AIIE Trans.* **10**(9) 260–270.

Thonemann, U.W., M.L. Brandeau. 1998. Note. Optimal storage assignment policies for automated storage and retrieval systems with stochastic demands. *Management Sci.* **44**(1) 142–148.

Thomas, L.M., R.D. Meller. 2014. Analytical models for warehouse configuration. *IIE Trans.* **46**(10) 928–947.

Weidinger, F., N. Boysen. 2018. Scattered Storage: How to distribute stock keeping units all around a mixed-shelves warehouse. *Transportation Sci.*, forthcoming.

Yu, Y., R. de Koster, X. Guo. 2015. Class-based storage with a finite number of items: Using more classes is not always better. *Production Oper. Management*, **24**(8), 1235–1247.

Zaerpour, N., Y. Yu, R. de Koster. 2017a. Optimal two-class-based storage in a live-cube compact storage system. *IIE Trans.* **49**(7) 653–668.

Zaerpour, N., Y. Yu, R. de Koster. 2017b. Small is beautiful: A framework for evaluating and optimizing live-cube compact storage systems. *Transportation Sci.* **51**(1) 34–51.

Appendix A.

Appendix A.1. The restricted linear decision rule (RLR)

In this section, we describe a robust optimization model proposed by Ang, Lim, and Sim (2012), and derive a storage-retrieval policy for products with prescheduled arrivals and random demands. The arrivals of products in each period are generally determined by the suppliers' production plans and follow some given schedule. In comparison, it is more difficult to determine the demands for the products in each period as they are more uncertain. As a result, we adopt the factor-based demand model described in Section 4.

A *storage-retrieval policy* decides where to store the pallets when they arrive and which pallets to retrieve when the demands are realized (note that a product may have multiple pallets at different locations in the warehouse). At the start of period t, pallets arrive at the warehouse and we decide which classes to store them based on $\tilde{\boldsymbol{z}}^{t-1}$. At the end of period t, the demands in the period (equivalently, the uncertain factors $\tilde{\boldsymbol{z}}^{t}$) are realized and we decide which classes to retrieve the pallets. We define the following decision variables: (i) $v_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t-1}\right)$ represents the number of arriving pallets of product i stored to class j at the start of period t after $\tilde{\boldsymbol{z}}^{t-1}$ is realized. We make this decision at the start of period t after the arrival of the pallets. (ii) $w_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t}\right)$ represents the number of pallets of product i retrieved from class j at the end of period t after $\tilde{\boldsymbol{z}}^{t}$ is realized. We make this decision at the end of period t after the demands in the period are realized. (iii) $x_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t-1}\right)$ represents the number of pallets of product i in class j at the start of period t. Note that this is a dependent variable determined by $v_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t-1}\right)$ and $w_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t}\right)$.

We define $\mathcal{U}$ as a family of distributions of the uncertain factors. For each distribution $\mathcal{P} \in \mathcal{U}$, $E_{\mathcal{P}}(\cdot)$ represents the expected value of $\cdot$ under $\mathcal{P}$. We adopt the approach by Gilboa and Schmeidler (1989) that minimizes the worst-case expected total cost over the family of distributions $\mathcal{U}$ as follows:

$$Z_R = \min \quad \max_{\mathcal{P}\in\mathcal{U}} E_{\mathcal{P}}\left[\sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{N}^+}\left(s_j v_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t-1}\right) + r_j w_{ij}^{t}\left(\tilde{\boldsymbol{z}}^{t}\right)\right)\right] \tag{A.1}$$

$$
\begin{aligned}
\text{s.t.} \quad & \sum_{j\in\mathcal{N}^+} v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) = a_i^t, && i\in\mathcal{M}, t\in\mathcal{T};\\
& \sum_{j\in\mathcal{N}^+} w_{ij}^t\left(\tilde{\boldsymbol{z}}^{t}\right) = \tilde{d}_i^t\left(\tilde{\boldsymbol{z}}^{t}\right), && i\in\mathcal{M}, t\in\mathcal{T};\\
& x_{ij}^{t+1}\left(\tilde{\boldsymbol{z}}^{t}\right) = x_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) + v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) - w_{ij}^t\left(\tilde{\boldsymbol{z}}^{t}\right), && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T};\\
& \sum_{i\in\mathcal{M}} \left(x_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) + v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right)\right) \le c_j, && j\in\mathcal{N}, t\in\mathcal{T};\\
& x_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) \ge 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}^+;\\
& v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right), w_{ij}^t\left(\tilde{\boldsymbol{z}}^{t}\right) \ge 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T};\\
& v_{ij}^t, x_{ij}^t \in \mathcal{F}_{M(t-1)}, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T};\\
& w_{ij}^t \in \mathcal{F}_{Mt}, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T};
\end{aligned}
$$

where $\mathcal{F}_p$ denotes a family of measurable functions that map $\Re^p$ to $\Re$. In an optimal solution to Problem (A.1), v_{ij}^t and w_{ij}^t are functions representing the optimal storage and retrieval decisions respectively.

Problem (A.1) is intractable in general. For tractability, we adopt the restricted linear decision rule (RLR) proposed by Ang et al. (2012). The RLR can be described by the following decision variables:

$$
v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) = v_{ij}^{t,0} + \sum_{k=1}^{t-1} v_{ij}^{t,k}\tilde{z}_i^k, \qquad i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}; \tag{A.2}
$$

$$
w_{ij}^t\left(\tilde{\boldsymbol{z}}^{t}\right) = w_{ij}^{t,0} + \sum_{k=1}^{t} w_{ij}^{t,k}\tilde{z}_i^k, \qquad i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}; \tag{A.3}
$$

where $v_{ij}^{t,0}, v_{ij}^{t,k}, w_{ij}^{t,0}$, and $w_{ij}^{t,k}$ are coefficients, for $i\in\mathcal{M}, j\in\mathcal{N}^+$, and $k\le t\in\mathcal{T}$. Given these coefficients, once the uncertain factors are realized we can evaluate the functions $v_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right)$ and $w_{ij}^t\left(\tilde{\boldsymbol{z}}^{t}\right)$, which determine the storage and the retrieval decisions respectively. Similarly, the functional form of the inventory level $x_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right)$ is defined as follows:

$$
x_{ij}^t\left(\tilde{\boldsymbol{z}}^{t-1}\right) = x_{ij}^{t,0} + \sum_{k=1}^{t-1} x_{ij}^{t,k}\tilde{z}_i^k, \qquad i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}^+; \tag{A.4}
$$

where $x_{ij}^{t,0}$ and $x_{ij}^{t,k}$ are coefficients, for $i\in\mathcal{M}, j\in\mathcal{N}^+$, and $k<t\in\mathcal{T}^+$.

To obtain the coefficients $v_{ij}^{t,0}, v_{ij}^{t,k}, w_{ij}^{t,0}$, and $w_{ij}^{t,k}$, for $i\in\mathcal{M}, j\in\mathcal{N}^+$, and $k\le t\in\mathcal{T}$, we first substitute Equations (A.2) and (A.3) into Problem (A.1). Using Theorem 1 and Proposition 2 of Ang et al. (2012), we convert Problem (A.1) into the following optimization problem:

$$
Z_{RLR} = \min \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{N}^+} \left(s_j v_{ij}^{t,0} + r_j w_{ij}^{t,0}\right) \tag{A.5}
$$

$$
\begin{aligned}
\text{s.t.} \qquad \sum_{j\in\mathcal{N}^+} v_{ij}^{t,k} &= \begin{cases} a_i^t, & \text{if } k=0, \\ 0, & \text{if } k\neq 0, \end{cases} && i\in\mathcal{M}, t\in\mathcal{T}, k\in\mathcal{T}; \\
\sum_{j\in\mathcal{N}^+} w_{ij}^{t,k} &= \begin{cases} d_i^t, & \text{if } k=0, \\ d_i^{t,k}, & \text{if } k\neq 0, \end{cases} && i\in\mathcal{M}, t\in\mathcal{T}, k\in\mathcal{T}; \\
x_{ij}^{t+1,k} &= \begin{cases} x_{ij}^{t,k}+v_{ij}^{t,k}-w_{ij}^{t,k}, & \text{if } k<t, \\ -w_{ij}^{t,k}, & \text{if } k=t, \end{cases} && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}; \\
\sum_{i\in\mathcal{M}}\left(\left(x_{ij}^{t,0}+v_{ij}^{t,0}\right)+q\sum_{k=1}^{t-1}\left|x_{ij}^{t,k}+v_{ij}^{t,k}\right|\right) &\leq c_j, && j\in\mathcal{N}, t\in\mathcal{T}; \\
v_{ij}^{t,0}-q\sum_{k=1}^{t-1}\left|v_{ij}^{t,k}\right| &\geq 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}; \\
w_{ij}^{t,0}-q\sum_{k=1}^{t}\left|w_{ij}^{t,k}\right| &\geq 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}; \\
x_{ij}^{t,0}-q\sum_{k=1}^{t-1}\left|x_{ij}^{t,k}\right| &\geq 0, && i\in\mathcal{M}, j\in\mathcal{N}^+, t\in\mathcal{T}^+.
\end{aligned}
$$

We solve Problem (A.5) once to obtain the optimal values of the coefficients $v_{ij}^{t,0}, v_{ij}^{t,k}, w_{ij}^{t,0}$, and $w_{ij}^{t,k}$, for $i\in\mathcal{M}, j\in\mathcal{N}^+$, and $k\leq t\in\mathcal{T}$, which determine the RLR in Equations (A.2) and (A.3).

Appendix A.2. Warehouse layout in the case study in Section 7

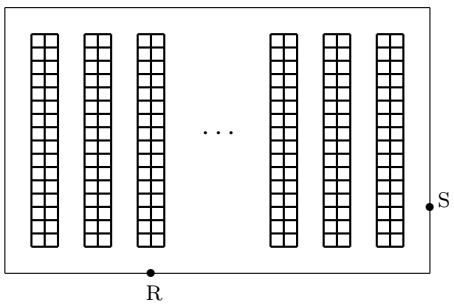


Figure A.10: Layout of the logistics provider's warehouse

Appendix A.3. Proof of Lemma 1

Equation (3) ensures that there is always a feasible solution to Problem (4). Due to the condition $c_j = 1$, $j = 1,\ldots,N$, and the nonnegativity constraints $x_{ij}^t \geq 0$, $v_{ij}^t \geq 0$, and $w_{ij}^t \geq 0$ in Problem (4),

the fourth constraint of Problem (4),

$$\sum_{i\in\mathcal{M}} \left(x_{ij}^t + v_{ij}^t\right) \leq c_j = 1, j \in \mathcal{N}, t \in \mathcal{T},$$

implies that $0 \leq v_{ij}^t \leq 1$, for $i \in \mathcal{M}, j \in \mathcal{N}, t \in \mathcal{T}$. Therefore, we have $\sum_{j\in\mathcal{N}} v_{ij}^t \leq N = L < \infty$, for $i \in \mathcal{M}, t \in \mathcal{T}$. Define $i' = \min\{i \in \mathcal{M} |\ a_i^t \to \infty \text{ for some } t \in \mathcal{T}\}$ and define $t' = \min\{t \in \mathcal{T} |\ a_{i'}^t \to \infty\}$. The first constraint of Problem (4) implies

$$\begin{aligned} a_{i'}^{t'} &= \sum_{j\in\mathcal{N}} v_{i'j}^{t'} + v_{i',N+1}^{t'}; \\ \Rightarrow a_{i'}^{t'} &\leq L + v_{i',N+1}^{t'}. \end{aligned}$$

Since $a_{i'}^{t'} \to \infty$, we have $v_{i',N+1}^{t'} \to \infty$. Thus,

$$\begin{aligned} f_{N+1}(\boldsymbol{z}) &= \sum_{t\in\mathcal{T}} v_{i',N+1}^t + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}\setminus\{i'\}} v_{i,N+1}^t + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} w_{i,N+1}^t \\ &\geq v_{i',N+1}^{t'} \to \infty. \end{aligned}$$

The nonnegativity constraints and the fourth constraint of Problem (4) also imply that $x_{ij}^t + v_{ij}^t \leq 1$, for $i \in \mathcal{M}, j \in \mathcal{N}, t \in \mathcal{T}$. The third constraint of Problem (4) implies

$$w_{ij}^t = \left(x_{ij}^t + v_{ij}^t\right) - x_{ij}^{t+1} \leq 1 - x_{ij}^{t+1} \leq 1, \quad i \in \mathcal{M}, j \in \mathcal{N}, t \in \mathcal{T}.$$

Therefore, for $j = 1, \ldots, N$, we have

$$\begin{aligned} f_j(\boldsymbol{z}) &= \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} v_{ij}^t + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} w_{ij}^t \\ &\leq 2MT < \infty. \end{aligned}$$

This completes the proof. □

Appendix A.4. Proof of Corollary 1

If Equation (3) and the condition in Lemma 1 hold, then $f_j(\mathbf{0})$, $f_j(\boldsymbol{q})$, and $f_j(-\boldsymbol{q})$ are finite, for $j \leq N$, and $f_{N+1}(\mathbf{0})$, $f_{N+1}(\boldsymbol{q})$, and $f_{N+1}(-\boldsymbol{q})$ tend to ∞. As a result, Equation (6) implies that $f_j < \infty$, for $j \leq N$, and $f_{N+1} \to \infty$. Thus, in Algorithm 1 we have $\Psi_1 = \Gamma_{L+1}$ and $\psi_{L+1} = \psi_{N+1} \to \infty$. Since $\psi_j < \infty$ for all $j \leq N = L$, we cannot find a k' such that $\psi_{k'} > \psi_{N+1}$. According to Algorithm 1, all classes $j = 1, \ldots, N$ are grouped into 1 cluster. This results in $N' = 2$. □

Appendix A.5. Proof of Lemma 2

We prove by contradiction. Given an optimal solution to Problem (4), suppose there exist $j', j'' \in \mathcal{N}$ such that $s_{j'} \leq s_{j''}$ and $f_{j''}(\boldsymbol{z}) = f_{j'}(\boldsymbol{z}) + \epsilon$ for some $\epsilon > 0$. From the third constraint of Problem (4) and the assumption that $x_{ij}^1 = 0$ for $i \in \mathcal{M}, j \in \mathcal{N}^+$, we have

$$
\begin{aligned}
& x_{ij}^{t+1} = x_{ij}^t + v_{ij}^t - w_{ij}^t,\ i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}; \\
\Rightarrow \quad & x_{ij}^{t+1} = x_{ij}^1 + \sum_{\tau=1}^{t} \left(v_{ij}^\tau - w_{ij}^\tau\right),\ i \in \mathcal{M}, , j \in \mathcal{N}^+, t \in \mathcal{T}; \\
\Rightarrow \quad & x_{ij}^{t+1} = \sum_{\tau=1}^{t} \left(v_{ij}^\tau - w_{ij}^\tau\right),\ i \in \mathcal{M}, , j \in \mathcal{N}^+, t \in \mathcal{T}.
\end{aligned}
$$

We can rewrite Problem (4) as

$$
Z_D(\boldsymbol{z}) = \min \quad \sum_{j \in \mathcal{N}^+} s_j \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} \left(v_{ij}^t + w_{ij}^t\right) \tag{A.6}
$$

$$
\begin{aligned}
\text{s.t.} \quad & \sum_{j \in \mathcal{N}^+} v_{ij}^t = a_i^t, && i \in \mathcal{M}, t \in \mathcal{T}; \\
& \sum_{j \in \mathcal{N}^+} w_{ij}^t = d_i^t + \sum_{k=1}^{t} d_i^{t,k} z_i^k, && i \in \mathcal{M}, t \in \mathcal{T}; \\
& x_{ij}^{t+1} = \sum_{\tau=1}^{t} \left(v_{ij}^\tau - w_{ij}^\tau\right), && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}; \\
& \sum_{i \in \mathcal{M}} \left(x_{ij}^t + v_{ij}^t\right) \leq 1, && j \in \mathcal{N}, t \in \mathcal{T}; \\
& x_{ij}^t \geq 0, && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}^+; \\
& v_{ij}^t, w_{ij}^t \geq 0, && i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}.
\end{aligned}
$$

In the above formulation, the indices j' and j'' are interchangeable in the constraints. Thus, we can exchange these indices in the solution and maintain its feasibility. The objective function in Equation

(9) after exchanging j' and j'' in the solution becomes

$$\begin{aligned} & s_{j'} f_{j''}(\boldsymbol{z}) + s_{j''} f_{j'}(\boldsymbol{z}) + \sum_{j \in \mathcal{N}^+ \setminus \{j', j''\}} s_j f_j(\boldsymbol{z}) \\ = \; & s_{j'} f_{j'}(\boldsymbol{z}) + s_{j''} f_{j'}(\boldsymbol{z}) + s_{j'} \epsilon + \sum_{j \in \mathcal{N}^+ \setminus \{j', j''\}} s_j f_j(\boldsymbol{z}) \\ \leq \; & s_{j'} f_{j'}(\boldsymbol{z}) + s_{j''} f_{j'}(\boldsymbol{z}) + s_{j''} \epsilon + \sum_{j \in \mathcal{N}^+ \setminus \{j', j''\}} s_j f_j(\boldsymbol{z}) \\ = \; & s_{j'} f_{j'}(\boldsymbol{z}) + s_{j''} f_{j''}(\boldsymbol{z}) + \sum_{j \in \mathcal{N}^+ \setminus \{j', j''\}} s_j f_j(\boldsymbol{z}). \end{aligned}$$

The last expression is the objective function under the original solution without exchanging j' and j''. Thus, we can find a better solution to the problem by exchanging j' and j'' in the original solution, which leads to a contradiction. □

Appendix A.6. Proof of Corollary 2

Suppose $N = L$, $s_j = r_j$, and $x_{ij}^1 = 0$, for all $i \in \mathcal{M}, j \in \mathcal{N}^+$. From Lemma 2 we know that if $s_{j'} \leq s_{j''}$ then $f_{j'}(\boldsymbol{z}) \geq f_{j''}(\boldsymbol{z})$, $j', j'' \in \mathcal{N}$, for any realization $\boldsymbol{z}$ of the uncertain factors. Equation (6) implies that if $s_{j'} \leq s_{j''}$ then $f_{j'} \geq f_{j''}$, for $j', j'' \in \mathcal{N}$, where f_j is the visit frequency of location j. This implies that ranking the locations from the highest visit frequency to the lowest visit frequency is equivalent to ranking the locations from the smallest location travel cost to the largest location travel cost. Therefore, ranking by visit frequency and ranking by location travel cost are equivalent for such a warehouse. □

Appendix A.7. Proof of Lemma 3

Given that $x_{ij}^1 = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+$ and $\sum_{j \in \mathcal{N}^+} v_{ij}^t = a_i^t$, for $t \in \mathcal{T}_p$, there are two cases:

1. For $j \in \mathcal{N}$, given $v_{ij}^t = n' \leq 1$ and because each pallet has a duration of stay p, we have $w_{ij}^{t+p-1} = n' = v_{ij}^t$.
2. For $j = N + 1$, given $v_{i,N+1}^t = n' \leq a_i^t$ and because each pallet has a duration of stay p, we have $w_{i,N+1}^{t+p-1} = n' = v_{i,N+1}^t$.

This proves the lemma. □

Appendix A.8. Proof of Lemma 4

Let $\mathcal{T}_1 = \{1, 2, \ldots, p-1\}$ and $\mathcal{T}_2 = \{p, p+1, \ldots, T\}$. We have $\sum_{t\in\mathcal{T}_1}\sum_{i\in\mathcal{M}} w_{ij}^t = 0$. According to Lemma 3, we have $\sum_{t\in\mathcal{T}_p}\sum_{i\in\mathcal{M}} v_{ij}^t = \sum_{t\in\mathcal{T}_p}\sum_{i\in\mathcal{M}} w_{ij}^{t+p-1} = \sum_{\tau\in\mathcal{T}_2}\sum_{i\in\mathcal{M}} w_{ij}^{\tau}$. For $j \in \mathcal{N}^+$, we have

$$f_j^s = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} v_{ij}^t = \sum_{t\in\mathcal{T}_p}\sum_{i\in\mathcal{M}} v_{ij}^t + \sum_{t\in\mathcal{T}\setminus\mathcal{T}_p}\sum_{i\in\mathcal{M}} v_{ij}^t \geq \sum_{t\in\mathcal{T}_2}\sum_{i\in\mathcal{M}} w_{ij}^t = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} w_{ij}^t = f_j^r.$$

□

Appendix A.9. Proof of Lemma 5

Since $p = 1$ for a balanced case, we have $\mathcal{T}_p = \mathcal{T}$, and $\mathcal{T} \setminus \mathcal{T}_p = \emptyset$. Part (i) of Lemma 5 follows directly from Lemma 3.

To prove part (ii) of Lemma 5, we consider the third constraint of Problem (4), we have

$$\begin{aligned} & x_{ij}^{t+1} = x_{ij}^t + v_{ij}^t - w_{ij}^t,\ i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}; \\ \Rightarrow \quad & x_{ij}^{t+1} = x_{ij}^t,\ i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}. \end{aligned}$$

Together with the conditions $x_{ij}^1 = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+$, the last equality implies that $x_{ij}^t = 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}^+$. □

Appendix A.10. Proof of Lemma 6

According to part (i) of Lemma 5, we have $v_{ij}^t = w_{ij}^t$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$. Thus, $f_j^s = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} v_{ij}^t = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} w_{ij}^t = f_j^r$. The objective function of Problem (4) becomes

$$\begin{aligned} & \sum_{j\in\mathcal{N}^+} s_j \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} v_{ij}^t + \sum_{j\in\mathcal{N}^+} r_j \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} w_{ij}^t \\ = \quad & \sum_{j\in\mathcal{N}^+} (s_j + r_j) f_j^s. \end{aligned}$$

The first and the second constraints of Problem (4) reduce to the first constraint of Problem (10). From the fourth constraint of Problem (4) and part (ii) of Lemma 5, we have

$$\begin{aligned} & \sum_{i\in\mathcal{M}} \left(x_{ij}^t + v_{ij}^t\right) \leq 1,\ j \in \mathcal{N}, t \in \mathcal{T}; \\ \Rightarrow \quad & \sum_{i\in\mathcal{M}} v_{ij}^t \leq 1,\ j \in \mathcal{N}, t \in \mathcal{T}; \\ \Rightarrow \quad & f_j^s = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{M}} v_{ij}^t \leq T,\ j \in \mathcal{N}. \end{aligned}$$

Since $v_{ij}^t \geq 0$, for $i \in \mathcal{M}, j \in \mathcal{N}^+, t \in \mathcal{T}$, we have $f_j^s \geq 0$, for $j \in \mathcal{N}^+$. □

Appendix A.11. Proof of Corollary 3

Problem (10) is a continuous Knapsack Problem (Sahni (1975)). It can be solved as follows. We first index the classes from 1 to $N+1$ such that $s_1+r_1 \leq s_2+r_2 \leq \cdots \leq s_{N+1}+r_{N+1}$. Let $\hat{a} = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{M}} a_i^t$. For $j \leq \lfloor \hat{a}/T \rfloor$, set $f_j^s = T$. For $j = \hat{j} \equiv \lfloor \hat{a}/T \rfloor + 1$, set $f_j^s = \hat{a} \bmod T$. For $j > \hat{j}$, set $f_j^s = 0$.

Since each class contains only one location, the above procedure results in the first $\lfloor \hat{a}/T \rfloor$ most economic locations to be visited more frequently than other locations. Throughout the planning horizon, we store T pallets to and retrieve T pallets from each of these most economic locations. Thus, the solution to Problem (10) implies that for a balanced warehouse with no initial inventory, it is optimal to first index its storage locations from the smallest location travel cost to the largest location travel cost. We then visit a location with a smaller index more frequently than a location with a larger index. Under the ranking by visit frequencies, the location with the highest visit frequency corresponds to the location with the smallest location travel cost. Thus, ranking the locations by visit frequency and ranking them by location travel cost are equivalent for the warehouse. □

References

Gilboa, I., D. Schmeidler. 1989. Maximin expected utility theory with non-unique prior. *J. Math. Econ.* **18** 141–153.