


Article

An Efficient Algorithm for the Joint Replenishment Problem with Quantity Discounts, Minimum Order Quantity and Transport Capacity Constraints

Shiyu Liu ¹ , Ou Liu ^{2,*} and Xiaoming Jiang ²¹ School of Economics and Management, Beihang University, Beijing 100191, China² Wenzhou Institute, University of Chinese Academy of Sciences, Wenzhou 325001, China

* Correspondence: oliu@ucas.ac.cn

Abstract: The joint replenishment problem has been extensively studied and the joint replenishment strategy has been adopted by a large variety of retailers in recent years. However, the joint replenishment problem under minimum order quantity and other constraints does not receive sufficient attention. This paper analyzes a retailing supply chain involving a supplier that provides quantity discount schedules and limits the order quantity. The order quantity constraints include minimum order requirements for each item and as to the total quantity; additionally, the latter cannot exceed the transport capacity constraint. These are common constraints in the retail industry today and create greater complexity and difficulty in the retailer's decision-making. To analyze the problem, an integer nonlinear programming model is set up to maximize retailers' profit with all practical constraints. A two-layer efficient algorithm named the Marginal and Cumulative Profit-Based Algorithm (MCPB) is then proposed to find whether to order and the optimal order quantity for each item. The results of computational experiments show that the proposed algorithm can find near-optimal solutions to the problem efficiently and is a reference for retailers to solve practical joint replenishment problems.

Keywords: order strategy; joint replenishment problem; quantity discount; minimum order quantity; transport capacity

MSC: 90B05; 90C10

Citation: Liu, S.; Liu, O.; Jiang, X. An Efficient Algorithm for the Joint Replenishment Problem with Quantity Discounts, Minimum Order Quantity and Transport Capacity Constraints. *Mathematics* **2023**, *11*, 1012. <https://doi.org/10.3390/math11041012>

Academic Editor: Andreas C. Georgiou

Received: 31 December 2022

Revised: 12 February 2023

Accepted: 15 February 2023

Published: 16 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The joint replenishment problem (JRP) is a cooperative strategy to reduce the total cost of procurement by sharing fixed costs through an optimal grouping of different types of purchased items and joint procurement of items in the same group to take advantage of the economic scale effect [1]. Currently, joint replenishment strategy, both in theory and practice, has significantly reduced company replenishment and inventory costs [2]. The joint replenishment strategy has been adopted by a large variety of supermarket chains, such as Wal-Mart, Carrefour, and other well-known companies. A famous Chinese hypermarket chain has planned to implement a joint replenishment strategy through an efficient algorithm. As the problem continues to be studied, researchers have relaxed many assumptions to make the problem scenario more relevant to reality, such as the consideration of positive lead time and shortages. Furthermore, there are various constraints being taken into account to meet the need of practical applications.

The classical joint replenishment problems were motivated by the fixed shared costs and the separated individual fixed costs [1,3]. However, grocery retailers or supermarket retailers have no fixed costs for an individual item ordered. Instead, suppliers constrain minimum order quantities (MOQ) to ensure their profits for each order and delivery. Minimum order quantity restrictions are widely used in business. For example, Sports Obermeyer, a fashion ski-wear distributor, requires MOQ from buyers, and Walmart and

Alibaba are similarly constrained by their suppliers [4]. Tuncel et al. demonstrate that MOQ contracts are very popular for suppliers in actual practice [5]. Therefore, retailers need to develop their inventory optimization algorithms to fit supplier MOQ constraints.

Nevertheless, the joint replenishment problem study under MOQ constraints has still not received sufficient attention [6]. To the best of our knowledge, only a few papers have addressed the joint replenishment problem with MOQ constraints. For example, Porras and Dekker applied a global optimization procedure to solve the JRP with MOQ to derive the bounds for a primary order cycle [7]. Noh proposed an efficient method to determine a base cycle length and safety factor that minimizes the buyer's total cost with quantity discounts and minimum order constraints [6]. Muriel focused on the optimal order strategy with a fixed joint-order cycle [8]. To enrich the study of JRP with MOQ, we establish a mathematical model to describe it and propose an efficient algorithm to solve it.

Unlike most studies on JRP, our model takes the objective function of maximizing the retailer's profits and accounts for out-of-stock items. For out-of-stock items, a unit shortage cost represents the retailer's potential loss. Additionally, beyond the minimum order quantities for each item, we should also consider total minimum order quantity and transport capacity constraints of the total replenishment. More interestingly, although there are a set of joint replenishment items, not all items need to be ordered at the same order point. Therefore, in our model, the decision variables include whether each item is ordered and the amount of replenishment if it is ordered.

The contributions of this study are as follows: (1) We extend the literature of JRP research by considering shortage, quantity discounts, MOQ for both single item and total replenishment, and transport capacity at the same time in the model to solve more realistic issues; (2) We propose an efficient algorithm to solve this JRP in a shorter time and determine the order quantity for each item to a near-optimal degree.

This paper is organized as follows. Section 2 introduces the related literature and summarizes the position of our work in the existing literature. Section 3 models a nonlinear integer programming of the JRP with shortage and MOQ and transport capacity constraints. The proposed marginal profit-based algorithm is described in Section 4. The computational experiments and the evaluation of the proposed algorithm are illustrated in Section 5. The paper is concluded in Section 6.

2. Literature Review

Over the past few decades, researchers have shown increasing attention to JRP, and different approaches have been proposed to solve both the classical JRP and the constrained JRP. The JRP is an NP-hard problem, and it is unlikely that a polynomial-time algorithm exists to solve JRP [9]. Goyal has proposed an algorithm to determine the order quantities for items in a JRP [10]. Moreover, an efficient heuristic algorithm was developed to solve the JRP problem [11], which was improved by Goyal and Belton [12] and Kaspi and Rosenblatt [13]. Since then, more and more heuristics have been developed, such as RAND [14], C-RAND [15], genetic algorithm (GA) [16–18], evolutionary algorithm (EA) [19–22] and simulated annealing (SA) [23]. Among the heuristics, GA and EA have been proven effective in solving the classical JRP [24].

With the continuous study of JRPs, there are more and more factors and constraints being considered in the problems. First, shortage is an important issue in JRPs. To address this important issue, several studies [25–27] employ a penalty cost per unit because of the shortages, representing the probable loss of profit. Furthermore, some extensions of the analysis that are closely related to this paper are quantity discounts, minimum order quantity constraints, and transport capacity constraints. Cha and Moon [28] firstly considered all-units quantity discounts for each item in JRP under constant demand. Moon [29] extended this research to multiple suppliers offering quantity discounts and proposed a hybrid genetic algorithm with resource constraints. Duran and Pérez Pozo [30] used techniques based on particle swarm optimization and a genetic algorithm to deal with a JRP regarding spare parts.

Furthermore, offering a quantity discount is a common approach of suppliers to encourage retailers to order more, and there are several researchers who have considered this issue in the study of inventory management. Paul [31] investigated two different models, with and without price discount, to determine the length of the family's cycle and the integer number of intervals that the replenishment quantity of each item will last. Cui [32] simultaneously considered two quantity discounts, an all-unit quantity discount and an incremental quantity discount, in the JRP and solved it by the locust swarms algorithm. Ai [33] constructed a mathematical model with a supplier selection system and a JRP where suppliers have different quantity discount schemes.

MOQ constraint was perhaps first considered by Fisher [34]. They considered a two-phase problem for multiple projects with a realistic setup in which MOQ constrains low-cost procurement. Robb [35] studied a periodic review system with MOQ constraints and proposed heuristic approaches for when the recommended order quantity is less than the specified minimum quantity. Similarly, Zhou [36] considered a single-item, periodic-review inventory system with MOQ and tested a simple heuristic policy specified by only two parameters (s, t) which was demonstrated to consistently outperform the best feasible (s, S) policy. Based on the same inventory system settings, Kiesmüller [37] proposed a periodic review policy, called (R, S, Q_{\min}) policy, where the order quantity of a single item was calculated as the inventory on hand plus orders on hand minus stalled orders, equal to or greater than level S . A more straightforward one-parameter policy called S policy was proposed when considering a single-item inventory system with both MOQ and batch orders [38]. Shen studied a two-echelon inventory system with one warehouse and multiple retailers [39]. In particular, the warehouse had a minimum order quantity requirement according to the supplier's regulations. They assumed that retailers had adopted the base-stock policy and designed a new heuristic ordering for the warehouse. All the above studies were conducted for single-item inventory systems with MOQ constraints. Only a few articles have investigated joint replenishment systems with MOQ constraints [6–8]. Therefore, to fill the research gap in this area, this paper focuses on joint replenishment systems with MOQ and derives the optimal order quantity of each item.

In addition, there are several other constraints besides MOQ constraints. For example, Moon and Cha [15] introduced resource constraints to the JRP. Houque considered the capacity and budget constraints of the joint replenishment with a shortage [40]. Shipment constraints and defective items that cannot be delivered together have also been studied [17]. Similarly, Otero-Palencia considered real-life capacity constraints such as finite storage and transport and solved them via genetic algorithms [41].

This study aims to model a constrained JRP for multiple items, in which demand is stochastic and shortage is allowed, with individual MOQ constraints for each item and total MOQ and transport capacity for total replenishment. This extended JRP is very critical for the retail industry during COVID-19. For example, this problem can be found in a supermarket where the same brand of goods is ordered from the same supplier as a group of joint replenishment items with MOQ constraints and transport capacity constraints from suppliers with different quantity discount schedules.

Table 1 summarizes how our work compares to other works and its position in the existing literature. It shows that there are several papers that have studied the relevant constraints in JRP. A few papers including [6,7] have analyzed JRP with MOQ constraint, even though the decision variables are not the order quantity. Muriel et al. analyzed JRP with MOQ constraint and aimed to get the optimal order quantity in the model, but the transport capacity and quantity discounts were not considered [8]. On the contrary, this paper simultaneously analyzes an extended JRP with stochastic customer demands, a discount schedule for each item, MOQ constraints for each item and as to total quantity, and transport capacity constraints at the. Thus, the current work can extend the literature of JRP research by allowing for these four realistic conditions and focusing on the optimal order quantity for each item.

Table 1. Position of Our Work in the Existing Literature.

Research Paper	JRP	MOQ	Transport Capacity	Quantity Discount	Shortage Allowed	Optimal Order Quantity	Stochastic Demand
Goyal & Belton [12]	✓					✓	
Robb & Silver [35]		✓				✓	
Cha & Moon [28]	✓			✓			
Porras & Dekker [7]	✓	✓					
Moon & Cha [15]	✓		✓				
Zhou et al. [36]		✓				✓	✓
Moon et al. [29]	✓			✓			
Kiesmüller et al. [37]		✓				✓	✓
Zhu et al. [38]		✓				✓	✓
Cui et al. [32]	✓			✓			
Ongkunaruk et al. [17]	✓		✓				
Chen et al. [25]	✓		✓		✓		
Noh et al. [6]	✓	✓		✓	✓		✓
Ai et al. [33]	✓			✓			
Muriel et al. [8]	✓	✓				✓	
This paper	✓	✓	✓	✓	✓	✓	✓

3. Problem Description and Model Formulation

Large supermarket chains such as Wal-Mart and Yonghui sell various items from suppliers. They usually prefer joint replenishment strategies to reduce shipping costs by regularly ordering multiple items from each supplier simultaneously. This study considers a grocery supply chain with a single retailer and a single supplier who provides quantity discount schedules and has MOQ constraints for each item and as to total replenishment. Furthermore, transport capacity is an upper limit for total replenishment. The order for each item can be fulfilled when the retailer places the order with the supplier. The main goal of this paper is to determine the optimal joint replenishment policy for this retailer, i.e., the items to be ordered in each order point and the order quantity of each item with the constraints above, such that the total expected profit of the retailer is maximized.

It is assumed that the order cycle and the lead time of items from the same supplier are stable, and that the retailer can only order at the order point. There are no supply constraints on suppliers, and retailers can have their orders fully met. Moreover, shortages are allowed, and out-of-stock items are lost without backlogging. The demand for each item is assumed to be independent, and the expected probability distribution of each item can be obtained (through machine learning, for example). As grocery supply chains are concentrated, there are no individual item setup costs, since minimum order quantities ensure that orders can cover any existing fixed costs [8].

To model the proposed JRP, indices, parameters, decision variables, and auxiliary variables are defined in Table 2.

The retailer’s expected profit is determined based on expected demand and available inventory. Inventory is fully used to meet demand, without inventory hoarding. If there is a surplus of inventory after demand is fulfilled, inventory holding costs are incurred; if demand is not fully met, out-of-stock losses result. Similar to the periodic strategies, it is believed that the replenishment quantity should meet the total demand for the order cycle and lead time, which means $D_{i_j} = d_{i_j} * (0 + L)$, and this is used as a benchmark to measure the out-of-stock loss. Thus, if $D_{i_j} > (I_i + q_i)$, which implies the item is out-of-stock, the retailer will have a unit shortage cost s_i for each out-of-stock item. Similarly, if $D_{i_j} \leq (I_i + q_i)$, items not sold will be placed in the warehouse and incur the unit-holding cost h_i . Meanwhile, the retailer will have a corresponding revenue p_i for each item sold and pay a purchase cost for each item ordered. This paper assumes that the supplier is providing a full-unit quantity discount schedule.

Table 2. Description of symbols.

Indices:	
i	item index, $i \in N$, where N is the set of items
i_j	the probable demand index of item i ($0 < p_{i_j} \leq 1$)
k	The price break index
Parameters:	
O	the order interval of joint replenishment
L	the lead time of joint replenishment
d_{i_j}	the average probable demand rate of item i during the order cycle and lead time
Pro_{i_j}	the probability of d_{i_j} of item i
I_i	the existing inventory of item i at the order point
p_i	the unit revenue of item i
h_i	the unit holding cost of item i
s_i	the unit shortage cost of item i
$b_{i,k}$	k th price break quantity required of item i
$c_{i,k}$	the unit purchase cost with price break k under all-unit quantity discounts
moq_i	the minimum order quantity of item i
Moq	the total minimum order quantity for all items
Tr	the transport capacity
Decision variables:	
q_i	The order quantity of item i (integer variable)
y_i	$= \begin{cases} 1, & \text{if item } i \text{ is ordered} \\ 0, & \text{otherwise} \end{cases}$
Auxiliary variable:	
X_{i_j}	$= \begin{cases} 1, & \text{if item } i \text{ is out of stock} \\ 0, & \text{otherwise} \end{cases}$

Furthermore, there is a probability Pro_{i_j} connected to each demand D_{i_j} . Thus, the expected profit of each kind of item in an order quantity q_i and a probable demand D_{i_j} can be calculated as follows:

$$R_i(q_i, D_{i_j}) = \begin{cases} Pro_{i_j} \left(p_i(q_i + I_i) - s_i \left(D_{i_j} - (I_i + q_i) \right) \right) - C_i q_i, & D_{i_j} \geq I_i + q_i \\ Pro_{i_j} \left(p_i D_{i_j} - h_i \left(I_i + q_i - D_{i_j} \right) \right) - C_i q_i, & D_{i_j} < I_i + q_i \end{cases} \quad (1)$$

The discount schedule C_i is described as follows:

$$C_i = \begin{cases} c_{i,1}, & \text{if } b_{i,1} \leq q_i < b_{i,2} \\ c_{i,2}, & \text{if } b_{i,2} \leq q_i < b_{i,3} \\ \dots & \\ c_{i,k}, & \text{if } b_{i,k} \leq q_i \end{cases} \quad (2)$$

where q_i is the order quantity and $b_{i,1} = moq_i$. In the schedule, the first price break $b_{i,1}$ is the minimum order quantity constraint, and it is assumed that $c_{i,k} < c_{i_k} < \dots < c_{i_2} < c_{i_1}$ [6]. Thus, the total expected profit of each item is a function of the order quantity q_i as:

$$R_i(q_i) = \sum_j R_i(q_i, D_{i_j}) \quad (3)$$

The MOQ constraint includes the MOQ constraint for a single item moq_i and the MOQ for total replenishment Moq . For a single item, the MOQ constraint can be understood

as being if an item is ordered, it must meet the minimum order quantity requirement. Otherwise, it will not be ordered, which can be symbolized as follows:

$$\begin{cases} \text{if } y_i = 1 \text{ then } q_i \geq moq_i \\ \text{if } y_i = 0 \text{ then } q_i = 0 \end{cases} \quad (4)$$

Based on the discussion above, the proposed JRP with MOQ and transport capacity constraints can be formulated as follows:

$$Max \sum_i \sum_j Pro_{ij} \left(\begin{aligned} & \left((1 - X_{ij}) (p_i(q_i + I_i) - s_i(D_{ij} - (I_i + q_i))) \right) \\ & + X_{ij} (p_i D_{ij} - h_i(I_i + q_i - D_{ij})) \end{aligned} \right) - C_i q_i \quad (5)$$

$$s.t. \begin{cases} My_i - q_i \geq 0 & (6) \\ M(1 - y_i) + q_i \geq moq_i & (7) \\ D_{ij} - (I_i + q_i y_i) \leq M(1 - X_{ij}) & (8) \\ D_{ij} - (I_i + q_i y_i) \geq -MX_{ij} & (9) \\ Moq \leq \sum_i q_i & (10) \\ \sum_i q_i \leq Tr & (11) \\ \sum_j Pro_{ij} = 1 & (12) \\ q_i \geq 0, q_i \in \mathbb{Z} \forall i & (13) \\ y_i \in \{0, 1\}, \forall i & (14) \\ X_{ij} \in \{0, 1\}, \forall i, j & (15) \end{cases}$$

The objective function Equation (5) represents the total expected profit of the retailer, where X_{ij} is an auxiliary variable defined to distinguish the two scenes in (1) as follows:

$$X_{ij} = \begin{cases} 1, & \text{if } D_{ij} > (I_i + q_i) \\ 0, & \text{if } D_{ij} \leq (I_i + q_i) \end{cases} \quad (16)$$

Equations (6) and (7) represent the single-item MOQ constraints for each kind of item and are consistent with the relationship in Equation (4) where M is a finitely large enough positive number. Equations (8) and (9) ensure that the value of X_{ij} meets the constraints in Equation (16). The total MOQ and transport capacity constraints are shown in Equations (10) and (11), respectively. Equation (12) is the expression of the probability distribution, in which the sum of probabilities is one. Equations (13)–(15) imply that q_i must be a positive integer and that y_i and X_{ij} are binary.

4. Marginal and Cumulative Profit-Based Algorithm (MCPB)

The traditional joint replenishment problem is defined as an NP-hard problem [9], and it is difficult to solve. In previous papers, most integer programming problems with constraints are solved by heuristic algorithms such as genetic algorithms [16–18]. However, heuristic algorithms have limitations in that the performance of the algorithms is unstable. In our model, there are multiple breakpoints in the profit function due to the minimum starting order constraint for a single item and the quantity discount. Thus, this paper proposes a Marginal and Cumulative Profit-Based Joint Replenishment Algorithm (MCPB). The main idea of this algorithm is to minimize marginal and cumulative profit loss principles to deal with the discontinuous profit function.

Based on Equations (5)–(15), it is easy to split the problem into several subproblems, as Equations (17) and (18), to find the optimal order quantity of each item without the total

constraints of MOQ and transport capacity. The optimal solutions are the initial input of the proposed algorithm to determine the ideal profit without any resource constraints.

$$\max R_i(q_i, y_i) = \sum_j \text{Pro}_{ij} \left(\left((1 - X_{ij}) \left((q_i + I_i) p_i - s_i (D_{ij} - (I_i + q_i)) \right) \right) + X_{ij} (D_{ij} p_i - h_i (I_i + q_i - D_{ij})) \right) - C_i q_i \quad (17)$$

$$\text{s.t.} \begin{cases} My_i - q_i \geq 0 \\ M(1 - y_i) + q_i \geq moq_i \\ D_{ij} - (I_i + q_i y_i) \leq M(1 - X_{ij}) \\ D_{ij} - (I_i + q_i y_i) \geq -MX_{ij} \end{cases} \quad (18)$$

Property 1. *If quantity discounts are not considered, the optimal replenishment amount of each subproblem is*

$$q_i = D_{ij} - I_i$$

Proof of Property 1. For each D_{ij} , the objective function of each subproblem can be rewritten as:

$$\max R_i(q_i) = \begin{cases} p_i(q_i + I_i) - s_i(D_{ij} - (I_i + q_i)) - cq_i, & q_i \leq D_{ij} - I_i \\ D_{ij} p_i - h_i(I_i + q_i - D_{ij}) - cq_i, & q_i > D_{ij} - I_i \end{cases} \quad (19)$$

For the first scenario,

$$\frac{dR_i(q_i)}{dq_i} = p_i + s_i - c \quad (20)$$

It is assumed that, if $p_i + s_i - c > 0$, then $R_i(q_i)$ is an increasing function with respect to q_i . Therefore, based on the range of values of q_i , the maximum value of the objective function is obtained at $q_i^* = D_{ij} - I_i$.

Similarly, for the second one,

$$\frac{dR_i(q_i)}{dq_i} = -(h_i + c) \quad (21)$$

Therefore, $R_i(q_i)$ is a decreasing function and the optimal solution is $q_i^* = D_{ij} - I_i$. Thus, $q_i^* = D_{ij} - I_i$ is the global optimal solution if there are no quantity discounts. \square

According to Property 1, for each $c_{i,k}$ in discount schedules C_i , the optimal solution is $q_i^* = D_{ij} - I_i$, regardless of price break. Then, comparing q_i^* with price interval, the optimal solution for each price interval can be written as:

$$q_{i,k}^* = \begin{cases} b_{i,2} - 1, & q_i^* > b_{i,2} \\ \dots \\ q_i^*, & b_{i,m} \leq q_i^* \leq b_{i,m+1} \\ \dots \\ b_{i,k}, & q_i^* < b_{i,k} \end{cases} \quad (22)$$

Thus, the global optimal solution for each item with a quantity discount is:

$$q_i^* = \max \left(R_i \left(q_{i,k}^* \right) \text{ for } k = 1, 2, 3, \dots, R_i(0) \right) \quad (23)$$

The profit of each item can be easily calculated by Equation (17), and the marginal profit loss of each item at a specific order quantity is the loss of profit when increasing or

decreasing one order relative to the current order quantity. If the order quantity is at the boundary of zero quantity and its individual minimum order quantity, the marginal profit loss is the loss of profit when increasing to the individual MOQ or decreasing to zero. What is more, the total MOQ and transport capacity constraints should be considered. Equations (24) and (25) show the marginal profit when the total MOQ constraint is not satisfied and the total transport capacity is not satisfied, respectively, where *Sum* refers to the current sum of order quantities.

$$mar_i = \begin{cases} R_i(q_i, y_i) - R_i(q_i + 1, y_i), & \text{if } q_i \neq 0 \text{ and } q_i + 1 \leq Tr \\ R_i(q_i, y_i) - R_i(moq_i, y_i), & \text{if } q_i = 0 \text{ and } moq_i + Sum \leq Tr \\ M, & \text{else} \end{cases} \quad (24)$$

$$mar_i = \begin{cases} R_i(q_i, y_i) - R_i(q_i - 1, y_i), & \text{if } q_i \neq 0 \text{ and } q_i - 1 \leq Moq \\ R_i(q_i, y_i) - R_i(0, y_i), & \text{if } q_i = moq_i \text{ and } Sum - moq_i \geq Moq \\ M, & \text{else} \end{cases} \quad (25)$$

Considering this JRP, a two-layer optimization method can be designed to solve it. The first layer’s optimization is based on minimizing marginal profit loss, regarded as the Marginal Profit-Based Algorithm (MPB). The second layer is designed to evaluate the result from the MPB by the principle of minimizing cumulative profit loss, and is named the Cumulative Profit-Based Algorithm (CPB). The whole algorithm, called the Marginal and Cumulative Profit-Based Joint Replenishment Algorithm (MCPB), controls the iterative process of MPB and CPB to satisfy all constraints and achieve global optimization. The process of the first layer optimization MPB can be described in Algorithm 1 in Python syntax for matrices.

Algorithm 1. Process of Marginal Profit-Based Algorithm (MPB).

- Step 0: Input current order strategy set, the sum of total order quantities set and the profit set
 - Step 1: Calculate the marginal profit of each item
 - Step 1.1: If the MOQ constraint is not satisfied, use Equation (24)
 - Step 1.2: If the transport capacity constraint is not satisfied, use Equation (25)
 - Step 2: Select an item to update order quantity based on the principle of minimal loss of marginal profit as $j = \min(mar[i], i = 0, 1, 2, \dots, n)$
 - Step 3: Update the order strategy
 - Step 3.1: If the MOQ constraint is not satisfied:
 - Step 3.1.1: If $q[j] = 0$, set $q[j] \leftarrow moq[j]$
 - Step 3.1.2: If $q[j] \geq moq[j]$, set $q[j] \leftarrow q[j] + 1$
 - Step 3.2: If the transport capacity constraint is not satisfied:
 - Step 3.2.1: If $q[j] = moq[j]$, set $q[j] \leftarrow 0$
 - Step 3.2.2: If $q[j] \geq moq[j]$, set $q[j] \leftarrow q[j] - 1$
 - Step 3.3: Update the total order quantities *Sum*, the order strategy $\{y[i][Sum], i \in N\}$ and the profit $P[Sum]$
 - Step 4: Update the breakpoints set *A*
 - If the item selected is in *A*, delete it
 - Step 5: Output the updated order strategy and the breakpoints set into CPB and stop
-

The input of MPB is the current order strategy and corresponding parameters. Step 1 calculates the marginal profit loss of each item according to different scenarios in Step 1.1 and Step 1.2. Step 2 is to select an item with the lowest marginal profit loss. Based on the result of Step 2, Step 3 updates the new order quantity, as determined by the equations in Step 3.1 and Step 3.2. Furthermore, in order to evaluate the result, Step 3.3 calculates the new total order quantities and profit. Then, Step 4 deletes the item if it is in the breakpoints set. Finally, the updated order strategy is output into the second layer algorithm CPB in Step 5.

After optimizing the first layer, the second layer algorithm, CPB, can be implemented. During the following layer algorithm, the result will be evaluated by the principle of minimizing cumulative profit loss. According to Figure 1, there are several breakpoints in profit functions. Thus, there are certain limitations to its results which can be solved by the next layer algorithm CPB. Algorithm 2 is the formalized scheme of the procedure in Python syntax for matrices.

Algorithm 2. Process of Cumulative Profit-Based Algorithm (CPB).

- Step 0: Input the result of the first layer MPB
- Step 1: Check if there are better solutions for the current total order quantities
 - Step 1.1: Check the item and order quantity in the breakpoints set A if cumulative loss of profit exceeds marginal profit in A , go to step 2
 - Step 1.2: else, go to step 4
 - Step 1.3: else, go to step 4
- Step 2: Select an item m to update the order quantity with the lowest cumulative profit loss
- Step 3: Update the order strategy
 - Step 3.1: Set $q[i] \leftarrow q[i][Sum - (A[m] - q[m])]$, $i \in N/m$
 - Step 3.2: Set $q[m] \leftarrow A[m]$
 - Step 3.3: Update the total order quantities Sum and the profit $P[Sum]$
- Step 4: Output the order strategy into the whole algorithm MCPB and stop

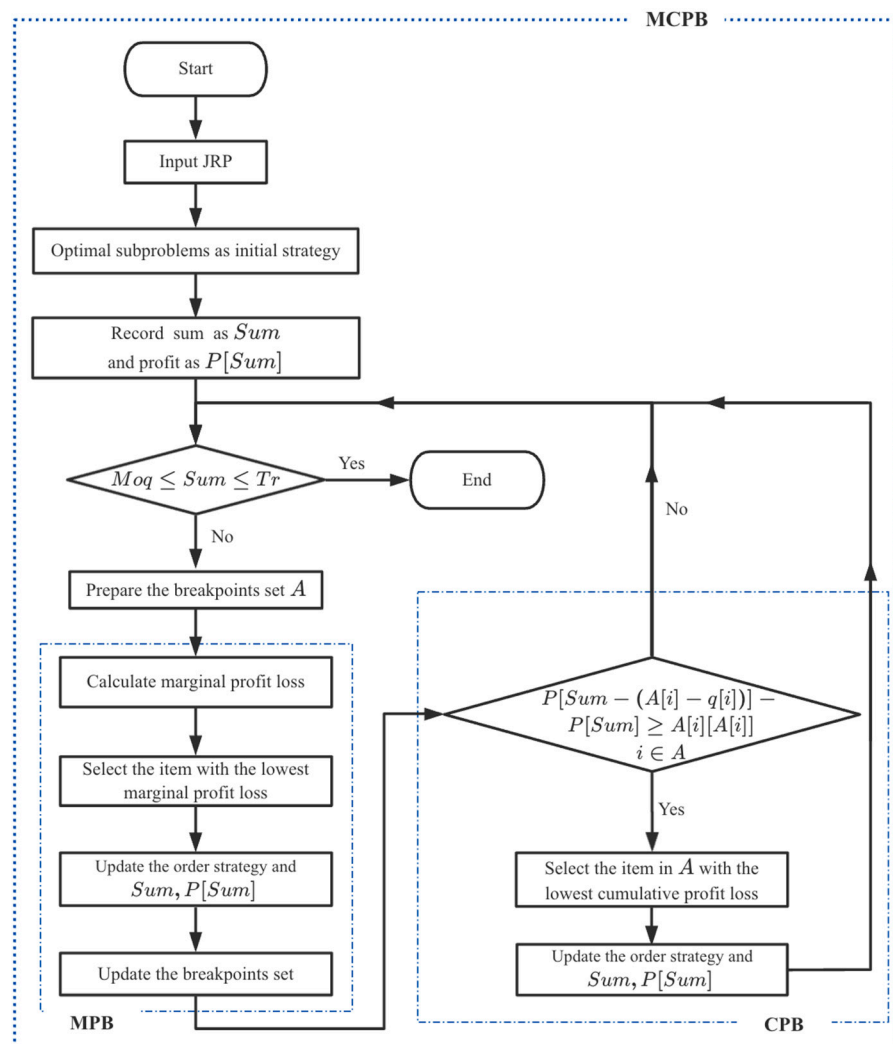


Figure 1. Flow chart of MCPB.

In Step 0, the result of MPB is input. Step 1 checks if the input is the optimized solution of the current total order quantities based on minimizing the cumulative profit loss. In this process, the cumulative reduction of total order quantities and the cumulative profit loss are compared with the order quantities and cumulative profit loss in the breakpoints set. If there exists any item satisfying both conditions, it means the current result is not optimal and needs to be updated later, while if it does not exist, it goes to Step 4 directly. Step 2 chooses the item with the lowest profit loss for those satisfied items. Then, Step 3 updates the optimal solution by dating back to the previous state. For example, suppose the MOQ constraint is not satisfied now. In that case, all the order quantities except the item selected will be updated by the order strategy in which the sum is $Sum - moq[m]$ and the item selected is replaced by zero. The process is similar when the transport capacity is unsatisfied, as shown in Step 3.2. After running the CPB, the final result is transferred to the whole MCPB algorithm in Algorithm 3 in Python syntax for matrices.

Algorithm 3. Process of Marginal Profit-Based and Cumulative Profit-Based Algorithm (MCPB).

- Step 0: Input the joint replenishment problem:
 Input joint replenishment item group $N = \{i, i = 0, 1, 2, \dots, n\}$
 Input parameters $(d[i][j], Pro[i], I[i], p[i], C[i], S[i], moq[i])$ of each item and (O, L, Moq, Tr) of the inventory system
 - Step 1: Initialization
 - Step 1.1: Initialize order strategy by maximizing subproblems based on Equation (23)
 - Step 1.2: Record the sum of initial order quantities as Sum_o and the sum of maximum profit by the dictionary as $P[Sum_o]$
 - Step 1.3: Record decision variables by dictionary form as $\{y[i][Sum_o], i \in N\}$ where i and Sum are the key values
 - Step 2: Check the MOQ and transport capacity constraints
 - Step 2.1: If both are satisfied, output the order strategy and stop
 - Step 2.2: If either of them is unsatisfied:
 prepare the breakpoints set A in Equation (22) and go to Step 3
 - Step 2.2.1: If the MOQ constraint is not satisfied:
 $A = \{j \mid q[j][Sum] = 0 \text{ and } Sum_o + moq[j] \leq Tr\} \cup \{j \mid \max(Pro[j][q], q \in [b_{i,0+z}, b_{i,k}], z = 0, 1, \dots, k)\}$
 - Step 2.2.2: If the transport capacity constraint is not satisfied:
 $A = \{j \mid q[j][Sum] = moq[j] \text{ and } Sum_o - moq[j] \geq Moq\} \cup \{j \mid \max(Pro[j][q], q \in [b_{i,0+z}, b_{i,k}], z = 0, 1, \dots, k)\}$
 - Step 3: Perform MPB with current order strategy
 - Step 4: Perform CPB with the updated order strategy obtained from Step 3 above
 - Step 5: Save the current order strategy and go to Step 2
-

MCPB is the main algorithm, synthesizing the previous two functions. Step 0 inputs the basic parameters of the joint replenishment problem. In Step 1, the order strategy is initialized by the subproblems' results of each item, and the order quantities, sum quantity, and profit are recorded in dictionary form with unique keys. Step 2 checks the constraints to control the iterative of MPB and CPB. If all constraints are satisfied in Step 2, the result will be output as the optimal solution. If not, a breakpoint set is set up, one which contains breakpoints for minimum order quantities, optimal solutions for each quantity discount interval of each item, and the corresponding profit loss at each breakpoint. Then, Step 3 and Step 4 are implemented, and unless the termination condition is met, the process will return to Step 2 and continue to the next iteration. The flow chart of the whole MCPB is shown in Figure 1.

The main contribution of the proposed algorithm is the combination of marginal profit loss and cumulative profit loss. The minimum margin profit principle ensures that the solution produced in every loop is nearly optimal, and the cumulative profit principle modifies the calculation for possible deviations from the minimum marginal profit principle

given the gaps in the profit functions because of the quantity discounts. Thus, the proposed marginal profit-based algorithm can be used to solve these joint replenishment problems with constraints to maximize retailers’ profits. In the next section, several computational experiments are designed to check the efficiency and accuracy of the MCPB.

5. Computational Experiments

Our experiments are designed based on a practical application scenario referencing a large supermarket chain in China. The supermarket sells a large number of items from different suppliers and requires a replenishment order regularly from the supplier, which provides a set of items with minimum order quantities, transport capacity constraints at a fixed cycle replenishment point, and quantity discount schedules for the purchase cost.

We have conducted three experiments to check the performance of the proposed solutions in various settings. To evaluate the performance of the proposed algorithm, solver CPLEX 22.1.0.0 is used as the benchmark. CPLEX is able to solve integer programming problems and has been proven to outperform other solvers to obtain the primary optimal solution [42,43]. The measurement of the accuracy of MCPB solutions can be defined as a percent deviation = $100(TP - TP^*)/TP$, where TP denotes the total profit obtained from the CPLEX and TP^* denotes the total profit of the solution from MCPB. All experiments are programmed in Python 3.9 and implemented on a PC (CPU: Apple M1; RAM: 16 GB; OS: macOS Monterey 12.5.1).

5.1. Experiment I: A Small-Scale Sample of the MCPB Algorithm

To evaluate the accuracy of the proposed algorithm, a joint replenishment example with 10 items is conducted first. The order interval is 3 days and the lead time is 2 days. The other values are in Table 3. Table 4 specifies the price discount schedule of each item. The total MOQ constraint and the maximum transport capacity vary to test the accuracy of the proposed algorithm in different scenarios.

Table 3. Values of parameters of item i in Experiment I.

Item (i)	1	2	3	4	5	6	7	8	9	10
I_i	5	2	3	20	0	6	5	4	3	0
p_i	30	50	40	35	25	30	30	20	30	35
s_i	15	25	20	18	15	10	12	10	15	12
h_i	2	5	4	1	2	3	2.5	2	1.5	1
$c_{i,1}$	10	20	15	10	12	18	16	10	12	14
moq_i	60	50	30	70	80	30	40	70	44	50

Table 4. Discount schedule.

Price Break	Purchase Cost
$moq_i \leq q_i < 1.5moq_i$	$c_{i,1}$
$1.5moq_i \leq q_i < 2moq_i$	$0.9c_{i,1}$
$2moq_i \leq q_i < 2.5moq_i$	$0.8c_{i,1}$
$q_i \geq 2.5moq_i$	$0.7c_{i,1}$

There are eight to-be-tested pairs of MOQ and transport capacity in Experiment I, and all eight results are shown in Table 5. Calculating the percent deviation between MCPB and CPLEX shows that the MCPB algorithm can achieve the same results in six scenarios, and in the remaining two scenarios, the maximum percent deviation is no more than 1%. In the scenario where MOQ is 300 and transport capacity is 600, we find that the deviation between MCPB and CPLEX mainly comes from the selection of replenishment items. In the scenario where MOQ is 500 and transport capacity is 800, the total order quantity of MCPB and CPLEX is different because of the termination conditions set by MCPB to meet the

constraints. Thus, MCPB can successfully generate nearly optimal solutions for small-scale problems.

Table 5. The comparison of output under different scenarios.

<i>Moq/Tr</i>	Total Order Quantity		Total Profit		Percent Deviation
	MCPB	CPLEX	MCPB	CPLEX	
300/600	600	600	10,667.61	10,766.34	0.926
400/700	700	700	12,719.18	12,719.18	0.000
500/800	800	777	13,060.18	13,151.43	0.694
800/1000	847	847	13,208.13	13,208.13	0.000
950/1200	950	950	12,592.55	12,592.55	0.000
1000/1500	1000	1000	12,283.56	12,283.56	0.000
1100/1400	1100	1100	11,665.15	11,665.15	0.000
1200/1500	1200	1200	10,965.15	10,965.15	0.000

5.2. Experiment II: A Large-Scale Sample of the MCPB Algorithm

To test our algorithm for more realistic problems, a second experiment is designed in which the problem size varies from 15 to 30. There is a mild assumption that, in practice, the unit lost-sales penalty cost usually exceeds the unit holding cost (i.e., $s_i > h_i$) and that the selling price must exceed the unit ordering cost (i.e., $p_i > c_{i,1}$) so that the retailer can make a profit [44]. The discount schedule of each item still follows the principle in Table 4. Three pairs of minimum order quantity (*Moq*) and transport capacity (*Tr*) constraints are set for each problem size. Thus, a total of 12 different problems are designed in the test. For each type of problem, five sets of items’ parameters are prepared by generating random numbers from the probability distributions given in Table 6. Thus, the total number of computational experiments is 60. The primary results of all computational experiments including total order quantity and total profit are summarized in Table A1 in Appendix A.

Table 6. Parameters setting of Experiment II.

I_i	p_i	s_i	h_i	$c_{i,1}$	moq_i	d_i
$[0, 20]^a$	$[20, 50]^a$	$[10, 25]^a$	$[1, 5]^a$	$[10, 30]^a$	$[30, 150]^a$	$[5, 50]^a$

^a Uniform distribution.

The average, maximum, minimum, and standard deviation of the five percent deviations between MCPB and CPLEX for each problem size and each set of constraints are summarized in Table 7. It shows that the average percent deviation ranges from 0.000% to 0.744% in 20-item problem sets. In all experiments, the MCPB finds the exact solutions 33 times out of 60 cases, and the worst deviation from CPLEX is no larger than 0.8%, which can be considered a near-optimum solution. Encouragingly, the standard deviations of the percentage deviations are minimal, ranging from 0.000% to 0.383%, which implies that MCPB has robust performance and is minimally affected by the randomness of the input data. More importantly, the deviation of MCPB does not become more extensive with the increase in the problem size. Furthermore, to test the performance on large problems, the problem size was expanded to 100 to 140 in the extended experiments and the results are summarized in Table A2. The results show that the percent deviation is no greater than 0.9% within a shorter time, as compared with CPLEX. The feature of MCPB is very promising for real-world applications, as MCPB’s error for a sizeable real-world problem is expected to be within a practically-acceptable limit.

Table 7. Output of percent deviation in Experiment II.

Problem Size	Moq/Tr	Percent Deviation			
		Average	Maximum	Minimum	Standard Deviation
15	1200/1800	0.417	0.740	0.000	0.383
	1500/2000	0.000	0.000	0.000	0.000
	2000/2500	0.003	0.016	0.000	0.007
20	2000/2500	0.166	0.744	0.000	0.325
	2500/3000	0.000	0.000	0.000	0.000
	3000/3500	0.000	0.000	0.000	0.000
25	2500/3000	0.124	0.394	0.000	0.172
	3000/3500	0.000	0.000	0.000	0.000
	3500/4000	0.003	0.009	0.000	0.004
30	3000/3500	0.385	0.628	0.000	0.236
	3500/4000	0.054	0.245	0.000	0.108
	4000/4500	0.025	0.127	0.000	0.057

5.3. Experiment III: Comparison of MCPB and CPLEX

The final experiment is designed to test the speed of MCPB. Based on the above experimental scenarios, the average solution times for MCPB and CPLEX are calculated for each problem size and summarized in Table 8 and Figure 2. MCPB can solve a given problem in less than 0.8 s and CPLEX consumes over 0.3 s with the simplest problem. It is clear from Figure 2 that there is a significant rise in CPLEX’s time when the problem size exceeds 15 items, while MCPB remains at a low level for all problem sizes. The experimental results clearly show that MCPB can be a very effective tool for retailers facing a variety of real-world JRPs with price discount schedules, minimum order quantities, and transport capacity constraints.

Table 8. The comparison of average computational time for MCPB and CPLEX.

Problem Size	Average Computational Time (in s)		Average Percent Deviation
	MCPB	CPLEX	
10	0.072	0.376	0.025
15	0.098	0.376	0.053
20	0.115	0.481	0.055
25	0.139	0.580	0.042
30	0.168	0.619	0.155
100	0.253	1.262	0.000
110	0.339	1.694	0.046
120	0.355	1.602	0.169
130	0.423	1.615	0.275
140	0.738	1.743	0.309

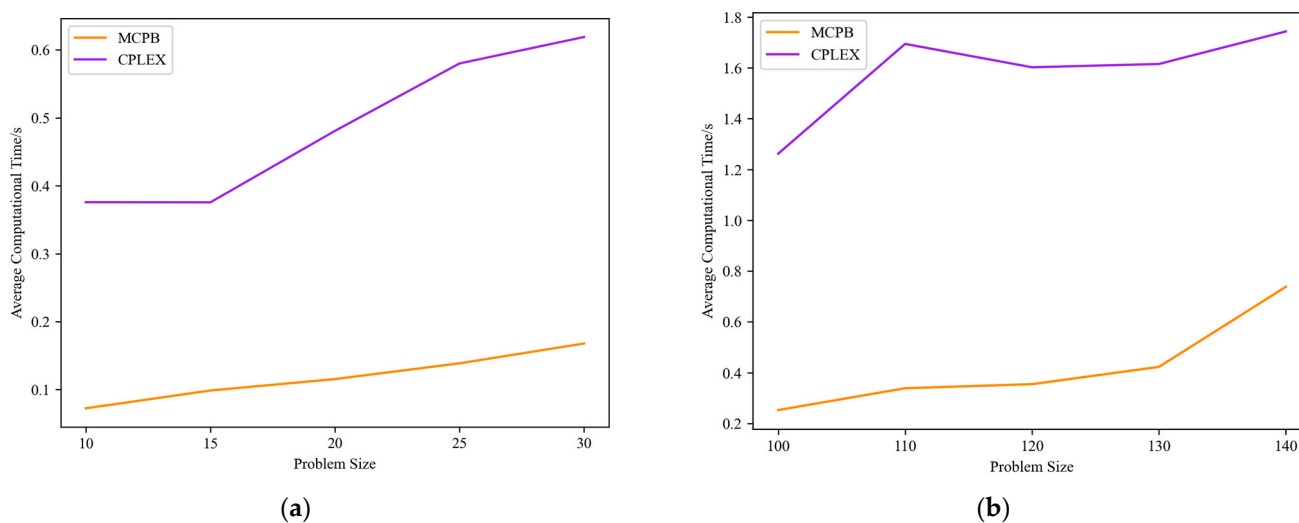


Figure 2. The comparisons of average computational time for MCPB and CPLEX: (a) problem size ranging from 10 to 30; and (b) problem size ranging from 100 to 140.

6. Conclusions

This paper analyzes a retail supply chain involving a supplier that provides many types of items and a retailer that orders items from suppliers on a known fixed cycle based on the demand of each item. Suppliers provide quantity discount schedules and accept orders that meet pre-set minimum order requirements for each item and as to the total quantity, which additionally does not exceed the transport capacity constraint. The retailer then tries to determine whether to order each item and the optimal order quantity to maximize its profit, which becomes an extended form of the JRP.

To solve the JRP, an integer nonlinear programming model is established. Based on the properties of the model, a two-layer algorithm called MCPB is proposed. To test the performance of the MCPB algorithm, we conduct small-scale and large-scale experiments. The results show that MCPB can obtain a nearly optimal solution with a bias below 1% both in the small-scale experiments and the large-scale experiments and achieve a speed about four times faster than CPLEX. Thus, the computational results show that the MCPB can find near-optimum solutions in a short time. The significance of our study is that retailers faced with similar JRP can use MCPB to decide whether to order items and the corresponding order quantity to maximize their profits. A large supermarket chain has now tested the algorithm proposed in this paper in China in a real-world scenario. It worked well during the test period and effectively improved the efficiency of joint replenishment and the retailer's revenue.

From a theoretical perspective, our research effectively enriches the field of research on JRP with constraints such as MOQ and transport capacity. From a practical perspective, our proposed algorithm can help retailers solve practical joint replenishment order quantity decision problems with several practical constraints in an efficient way. As to the future, there are several ways to further extend the study. One promising approach is to incorporate multiple suppliers into the current problem, thus making it a problem of collaborative replenishment between multiple suppliers. Another interesting approach might be to extend JRP by considering time-invariant dynamic joint replenishment.

Author Contributions: Conceptualization and methodology, S.L. and O.L.; validation, investigation, writing—original draft preparation, visualization, S.L.; validation and writing—review and editing, O.L. and X.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

The total order quantity and total profit of all scenarios in Experiment II are shown in Table A1.

Table A1. The results of all scenarios in Experiment II.

Problem Size	Moq/Tr	Total Order Quantity		Total Profit	
		MCPB	CPLEX	MCPB	CPLEX
15	1200/1800	1800	1800	41,336.09	41,336.1
		1800	1800	60,134.2	60,521.6
		1800	1800	32,418.88	32,649.5
		1800	1800	58,393.4	58,393.4
		1800	1800	40,696.89	41,000.3
	1500/2000	1851	1851	41,977.70	41,977.7
		2000	2000	66,475.92	66,475.9
		2000	2000	38,004.12	38,004.1
		1941	1941	61,186.40	61,186.4
		1971	1971	44,734.39	44,734.4
	2000/2500	2000	2000	41,589.70	41,589.70
		2145	2145	68,419.10	68,419.09
		2155	2155	40,505.59	40,505.59
		2000	2000	60,806.00	60,806.00
		2000	2007	44,507.80	44,514.80
20	2000/2500	2500	2500	62,100.1	62,100.1
		2500	2483	79,029.9	79,621.89
		2500	2500	45,700.95	45,739.93
		2500	2500	86,557.75	86,557.75
		2450	2450	51,923.39	51,923.39
	2500/3000	2519	2519	62,264.7	62,264.7
		2563	2563	80,354.4	80,354.4
		2825	2825	52,507.75	52,507.75
		2795	2795	92,607.14	92,607.14
		2500	2500	51,609.39	51,609.39
3000/3500	3000	3000	60,136.7	60,136.7	
	3000	3000	77,582.7	77,582.7	
	3000	3000	51,418.75	51,418.75	
	3000	3000	91,526.35	91,526.35	
	3000	3000	46,847.79	46,847.79	
25	2500/3000	3000	2961	74,716.1	74,727.69
		3000	2995	89,460.9	89,641.4
		3000	3000	53,976.07	54,189.38
		3000	3000	99,468.76	99,478.18
		3000	3000	67,422.3	67,422.3
	3000/3500	3085	3085	75,595.7	75,595.7
		3228	3228	93,368.4	93,368.4
		3500	3500	65,901.47	65,901.46
		3470	3470	110,258.95	110,258.94
		3052	3052	68,290.9	68,290.9
3500/4000	3500	3500	74,071.69	74,071.69	
	3500	3500	91,742.19	91,742.19	
	3530	3530	66,164.25	66,164.25	
	3500	3500	110,155.85	110,165.55	
	3500	3503	64,589.3	64,592.29	

Table A1. Cont.

Problem Size	Moq/Tr	Total Order Quantity		Total Profit	
		MCPB	CPLEX	MCPB	CPLEX
30	3000/3500	3500	3500	82,927.1	83,346.1
		3500	3495	105,211.4	105,659
		3500	3500	72,782.99	73,054.9
		3500	3500	118,282.42	119,030
		3500	3500	80,054.3	80,054.3
	3500/4000	3957	3957	90,636.2	90,636.2
		3722	3722	108,890	108,890
		4000	3997	83,873.89	84,080.3
		4000	3999	129,375.69	129,405
		3552	3552	80,922.9	80,922.9
	4000/4500	4000	4055	90,486.7	90,602.2
		4000	4000	107,644.9	107,645
		4068	4068	84,892.5	84,887.6
		4010	4010	129,414.47	129,414
		4000	4000	78,722.3	78,722.3

Appendix B

In order to reflect more realistic problems, the problem size is expanded to 100 to 140 in the extended experiment. Following the setting of Experiment II, each item’s data is designed by generating random numbers from the probability distributions specified in Table 6 and the discount schedule of each item still follows the principle in Table 4. For each problem size, three pairs of constraints are set. The total profit and the computational time of MCPB and CPLEX are shown in Table A2.

Table A2. The results of extended experiments on larger scales.

Size	Moq/Tr	MCPB		CPLEX		Percent Deviation
		Total Profit	Time (in s)	Total Profit	Time (in s)	
100	11,000/12,000	228,662.6	0.254	228,663	1.089	0.000
	12,000/13,000	228,590	0.137	228,590	1.348	0.000
	13,000/14,000	225,831.7	0.368	225,832	1.348	0.000
110	12,000/13,000	24,759.2	0.298	247,935	1.872	0.138
	13,000/14,000	248,189.5	0.368	248,190	1.721	0.000
	14,000/15,000	24,648.67	0.352	246,487	1.490	0.000
120	12,000/13,000	253,828.45	0.691	255,105	1.562	0.500
	13,000/14,000	267,000	0.247	267,019	1.684	0.007
	14,000/15,000	268,088.86	0.126	268,089	1.561	0.000
130	13,000/14,000	268,940.95	0.424	271,097	1.623	0.795
	14,000/15,000	282,736.07	0.381	282,818	1.724	0.029
	15,000/16,000	284,296.57	0.465	284,297	1.497	0.000
140	14,000/15,000	284,689.63	0.720	287,051	1.823	0.823
	15,000/16,000	300,190.61	0.741	300,507	1.714	0.105
	16,000/17,000	305,011.06	0.752	305,011	1.691	0.000

References

- Goyal, S.K.; Satir, A.T. Joint replenishment inventory control: Deterministic and stochastic models. *Eur. J. Oper. Res.* **1989**, *38*, 2–13. [CrossRef]
- Hsu, S.-L. Optimal joint replenishment decisions for a central factory with multiple satellite factories. *Expert Syst. Appl.* **2009**, *36*, 2494–2502. [CrossRef]

3. Khouja, M.; Goyal, S. A review of the joint replenishment problem literature: 1989–2005. *Eur. J. Oper. Res.* **2008**, *186*, 1–16. [[CrossRef](#)]
4. Zhao, Y.; Katehakis, M.N. On the structure of optimal ordering policies for stochastic inventory systems with minimum order quantity. *Probab. Eng. Inf. Sci.* **2006**, *20*, 257–270. [[CrossRef](#)]
5. Tuncel, O.; Taneri, N.; Hasija, S. Why are minimum order quantity contracts popular in practice? A behavioral investigation. *Manuf. Serv. Oper. Manag.* **2022**, *24*, 2166–2182. [[CrossRef](#)]
6. Noh, J.S.; Kim, J.S.; Sarkar, B. Stochastic joint replenishment problem with quantity discounts and minimum order constraints. *Oper. Res.* **2019**, *19*, 151–178. [[CrossRef](#)]
7. Porras, E.; Dekker, R. An efficient optimal solution method for the joint replenishment problem with minimum order quantities. *Eur. J. Oper. Res.* **2006**, *174*, 1595–1615. [[CrossRef](#)]
8. Muriel, A.; Chugh, T.; Prokle, M. Efficient algorithms for the joint replenishment problem with minimum order quantities. *Eur. J. Oper. Res.* **2022**, *300*, 137–150. [[CrossRef](#)]
9. Arkin, E.; Joneja, D.; Roundy, R. Computational complexity of uncapacitated multi-echelon production planning problems. *Oper. Res. Lett.* **1989**, *8*, 61–66. [[CrossRef](#)]
10. Goyal, S.K. Determination of optimum packaging frequency of items jointly replenished. *Manag. Sci.* **1974**, *21*, 436–443. [[CrossRef](#)]
11. Silver, E.A. A simple method of determining order quantities in joint replenishments under deterministic demand. *Manag. Sci.* **1976**, *22*, 1351–1361. [[CrossRef](#)]
12. Goyal, S.K.; Belton, A.S. Note on “A simple method of determining order quantities in joint replenishments under deterministic demand”. *Manag. Sci. (Pre-1986)* **1979**, *25*, 604.
13. Kaspi, M.; Rosenblatt, M.J. An improvement of Silver’s algorithm for the joint replenishment problem. *AIIE Trans.* **1983**, *15*, 264–267. [[CrossRef](#)]
14. Li, Q. Solving the multi-buyer joint replenishment problem with the RAND method. *Comput. Ind. Eng.* **2004**, *46*, 755–762. [[CrossRef](#)]
15. Moon, I.K.; Cha, B.C. The joint replenishment problem with resource restriction. *Eur. J. Oper. Res.* **2006**, *173*, 190–198. [[CrossRef](#)]
16. Hong, S.-P.; Kim, Y.-H. A genetic algorithm for joint replenishment based on the exact inventory cost. *Comput. Oper. Res.* **2009**, *36*, 167–175. [[CrossRef](#)]
17. Ongkunaruk, P.; Wahab, M.I.M.; Chen, Y. A genetic algorithm for a joint replenishment problem with resource and shipment constraints and defective items. *Int. J. Prod. Econ.* **2016**, *175*, 142–152. [[CrossRef](#)]
18. Zapata-Cortes, J.A.; Arango-Serna, M.D.; Saldarriaga-Romero, V.J. The constrained joint replenishment problem using direct and indirect grouping strategies with genetic algorithms. In *Best Practices in Manufacturing Processes*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 239–259.
19. Olsen, A.L. An evolutionary algorithm to solve the joint replenishment problem using direct grouping. *Comput. Ind. Eng.* **2005**, *48*, 223–235. [[CrossRef](#)]
20. Qu, H.; Ai, X.-Y.; Wang, L. Optimizing an integrated inventory-routing system for multi-item joint replenishment and coordinated outbound delivery using differential evolution algorithm. *Appl. Soft Comput.* **2020**, *86*, 105863. [[CrossRef](#)]
21. Wang, L.; Dun, C.-X.; Bi, W.-J.; Zeng, Y.-R. An effective and efficient differential evolution algorithm for the integrated stochastic joint replenishment and delivery model. *Knowl.-Based Syst.* **2012**, *36*, 104–114. [[CrossRef](#)]
22. Zeng, Y.-R.; Peng, L.; Zhang, J.; Wang, L. An effective hybrid differential evolution algorithm incorporating simulated annealing for joint replenishment and delivery problem with trade credit. *Int. J. Comput. Intell. Syst.* **2016**, *9*, 1001–1015. [[CrossRef](#)]
23. Mohammaditabar, D.; Ghodspour, S.H. A supplier-selection model with classification and joint replenishment of inventory items. *Int. J. Syst. Sci.* **2016**, *47*, 1745–1754. [[CrossRef](#)]
24. Olsen, A.L. Inventory replenishment with interdependent ordering costs: An evolutionary algorithm solution. *Int. J. Prod. Econ.* **2008**, *113*, 359–369. [[CrossRef](#)]
25. Chen, Y.; Yang, L.; Jiang, Y.; Wahab, M.I.M.; Yang, J. Joint replenishment decision considering shortages, partial demand substitution, and defective Items. *Comput. Ind. Eng.* **2019**, *127*, 420–435. [[CrossRef](#)]
26. Devy, N.L.; Ai, T.J.; Astanti, R.D. A joint replenishment inventory model with lost sales. In *Proceedings of the IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2018; Volume 337, p. 012018.
27. Rojas, F. A joint replenishment supply model for multi-products grouped by several variables with random and time dependence demand. *J. Model. Manag.* **2019**, *15*, 276–296. [[CrossRef](#)]
28. Cha, B.C.; Moon, I.K. The joint replenishment problem with quantity discounts under constant demand. *OR Spectr.* **2005**, *27*, 569–581. [[CrossRef](#)]
29. Moon, I.K.; Goyal, S.K.; Cha, B.C. The joint replenishment problem involving multiple suppliers offering quantity discounts. *Int. J. Syst. Sci.* **2008**, *39*, 629–637. [[CrossRef](#)]
30. Duran, O.; Pérez Pozo, L. Solution of the spare parts joint replenishment problem with quantity discounts using a discrete particle swarm optimization technique. *Stud. Inform. Control.* **2013**, *22*, 319–328. [[CrossRef](#)]
31. Paul, S.; Wahab, M.I.M.; Ongkunaruk, P. Joint replenishment with imperfect items and price discount. *Comput. Ind. Eng.* **2014**, *74*, 179–185. [[CrossRef](#)]
32. Cui, L.; Deng, J.; Wang, L.; Xu, M.; Zhang, Y. A novel locust swarm algorithm for the joint replenishment problem considering multiple discounts simultaneously. *Knowl.-Based Syst.* **2016**, *111*, 51–62. [[CrossRef](#)]

33. Ai, X.; Yue, Y.; Xu, H.; Deng, X. Optimizing multi-supplier multi-item joint replenishment problem for non-instantaneous deteriorating items with quantity discounts. *PLoS ONE* **2021**, *16*, e0246035. [[CrossRef](#)] [[PubMed](#)]
34. Fisher, M.; Raman, A. Reducing the cost of demand uncertainty through accurate response to early sales. *Oper. Res.* **1996**, *44*, 87–99. [[CrossRef](#)]
35. Robb, D.J.; Silver, E.A. Inventory management with periodic ordering and minimum order quantities. *J. Oper. Res. Soc.* **1998**, *49*, 1085–1094. [[CrossRef](#)]
36. Zhou, B.; Zhao, Y.; Katehakis, M.N. Effective control policies for stochastic inventory systems with a minimum order quantity and linear costs. *Int. J. Prod. Econ.* **2007**, *106*, 523–531. [[CrossRef](#)]
37. Kiesmüller, G.P.; De Kok, A.G.; Dabia, S. Single item inventory control under periodic review and a minimum order quantity. *Int. J. Prod. Econ.* **2011**, *133*, 280–285. [[CrossRef](#)]
38. Zhu, H.; Liu, X.; Chen, Y.F. Effective inventory control policies with a minimum order quantity and batch ordering. *Int. J. Prod. Econ.* **2015**, *168*, 21–30. [[CrossRef](#)]
39. Shen, H.; Tian, T.; Zhu, H. A two-echelon inventory system with a minimum order quantity requirement. *Sustainability* **2019**, *11*, 5059. [[CrossRef](#)]
40. Hoque, M.A. An optimal solution technique for the joint replenishment problem with storage and transport capacities and budget constraints. *Eur. J. Oper. Res.* **2006**, *175*, 1033–1042. [[CrossRef](#)]
41. Otero-Palencia, C.; Amaya-Mier, R.; Yie-Pinedo, R. A stochastic joint replenishment problem considering transportation and warehouse constraints with gainsharing by shapley value allocation. *Int. J. Prod. Res.* **2019**, *57*, 3036–3059. [[CrossRef](#)]
42. Anand, R.; Aggarwal, D.; Kumar, V. A comparative analysis of optimization solvers. *J. Stat. Manag. Syst.* **2017**, *20*, 623–635. [[CrossRef](#)]
43. Belotti, P.; Bonami, P.; Fischetti, M.; Lodi, A.; Monaci, M.; Nogales-Gómez, A.; Salvagnin, D. On handling indicator constraints in mixed integer programming. *Comput. Optim. Appl.* **2016**, *65*, 545–566. [[CrossRef](#)]
44. Keskin, N.B.; Li, Y.; Song, J.-S. Data-driven dynamic pricing and ordering with perishable inventory in a changing environment. *Manag. Sci.* **2022**, *68*, 1938–1958. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.