Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 136047, 16 pages
http://dx.doi.org/10.1155/2014/136047

Research Article

Multi-Item Multiperiodic Inventory Control Problem with Variable Demand and Discounts: A Particle Swarm Optimization Algorithm

Seyed Mohsen Mousavi,[1] S. T. A. Niaki,[2] Ardeshir Bahreininejad,[1] and Siti Nurmaya Musa[1]

[1] *Department of Mechanical Engineering, Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia*
[2] *Department of Industrial Engineering, Sharif University of Technology, P.O. Box 11155-0414 Azadi Avenue, Tehran 1458889694, Iran*

Correspondence should be addressed to Ardeshir Bahreininejad; bahreininejad@um.edu.my

Received 21 April 2014; Accepted 28 May 2014; Published 30 June 2014

Academic Editor: Leopoldo Eduardo Cardenas-Barron

A multi-item multiperiod inventory control model is developed for known-deterministic variable demands under limited available budget. Assuming the order quantity is more than the shortage quantity in each period, the shortage in combination of backorder and lost sale is considered. The orders are placed in batch sizes and the decision variables are assumed integer. Moreover, all unit discounts for a number of products and incremental quantity discount for some other items are considered. While the objectives are to minimize both the total inventory cost and the required storage space, the model is formulated into a fuzzy multicriteria decision making (FMCDM) framework and is shown to be a mixed integer nonlinear programming type. In order to solve the model, a multiobjective particle swarm optimization (MOPSO) approach is applied. A set of compromise solution including optimum and near optimum ones via MOPSO has been derived for some numerical illustration, where the results are compared with those obtained using a weighting approach. To assess the efficiency of the proposed MOPSO, the model is solved using multi-objective genetic algorithm (MOGA) as well. A large number of numerical examples are generated at the end, where graphical and statistical approaches show more efficiency of MOPSO compared with MOGA.

1. Introduction and Literature Review

Most real-world problems in industries and commerce are studied as an optimization problem involving a single objective. The assumption that organizations always seek to maximize (or minimize) their profit (or their cost) rather than making trade-offs among multiple objectives has been censured for a long time. Generally, classical inventory models are developed under the basic assumption that the management purchases or produces a single product. However, in many real-life conditions, this assumption does not hold. Instead, many firms, enterprises, or vendors are motivated to store a number of products in their shops for more profitable business affairs. Another cause of their motivation is to attract the customers to purchase several items in one showroom or shop.

This work proposes a multiperiod inventory model for seasonal and fashion items. The multiperiodic inventory control problems have been investigated in depth in different research. Chiang [1] investigated a periodic review inventory model in which the period is partly long. The important aspect of his study was to introduce emergency orders to prevent shortages. He employed a dynamic programming approach to model the problem. Mohebbi and Posner [2] investigated an inventory system with periodic review, multiple replenishment, and multilevel delivery. Assuming a Poisson process for the demand, shortages were allowed in this research, and the lost sale policy could be employed. Lee and Kang [3] developed a model for managing inventory of a product in multiple periods. Their model was first derived for one item and then was extended for several products. Mousavi et al. [4] proposed a multiproduct multiperiod inventory control problem under time value of money and inflation where total storage space and budget were limited. They solved the problem using two metaheuristic algorithms,

that is, genetic algorithm and simulated annealing. Mirzapour Al-e-hashem and Rekik [5] presented a multiproduct multiperiod inventory routing problem, where multiple constrained vehicles distributed products from multiple suppliers to a single plant to meet the given demand of each product over a finite planning horizon. Janakiraman et al. [6] analyzed the multiperiodic newsvendor problem and proposed some new results.

The quantity discount is of increasing attention due to its practical importance in purchasing and control of items. Usually, one derives the better marginal cost of purchase/production by taking advantages of the chances of cost savings through bulk purchase/production. Furthermore, in supply chain environments, quantity discounts can be considered an inventory coordination mechanism between buyers and suppliers [7]. In the literature of quantity discounts, Benton [8] considered an inventory system with quantity discount with multiple price breaks and alternative purchasing and lot-sizing policy. Abad [9, 10] proposed models for joint price and lot size determination when supplier offers either incremental (IQD) or all unit (AUD) quantity discounts. K. Maiti and M. Maiti [11] developed a model for a multi-item inventory control system of breakable items with AUD and IQD (and a combination of the two policies) and proposed genetic algorithm to solve the model. Sana and Chaudhuri [12] extended an EOQ model by relaxation of the preassumptions related to payments, allowing delay on delivery and discounts. They used a mixed integer nonlinear programming technique to model the problem. Taleizadeh et al. [13] considered a genetic algorithm to optimize multiproduct multiconstraint inventory control systems with stochastic replenishment intervals and discount. Recently, several works such as the ones in [4, 14–16] have also spotted discounts in inventory control problems. Huang and Lin [17] addressed an integrated model that scheduled multi-item replenishment with uncertain demand to determine delivery routes and truck loads. In this study, the products are purchased in different periods under AUD and IQD policies.

Metaheuristic algorithms have been suggested to solve some of the existing developed inventory problems in the literature. Some of these algorithms are tabu search [18, 19, 31], genetic algorithms (GA) [20–22, 32], simulating annealing (SA) [23, 33, 34], evolutionary algorithm [24, 35], threshold accepting [30], neural networks [36], ant colony optimization [37], fuzzy simulation [25], and harmony search [26, 38–41].

Inspired by social behavior of bird flocking or fish schooling, particle swarm optimization (PSO) is also a population-based stochastic optimization metaheuristic developed by Kennedy and Eberhart [42]. Recently, researchers have employed this effective technique to find optimal or near optimal solutions of their inventory control problems. For example, Taleizadeh et al. [43] employed PSO to solve their integer nonlinear programming model of a constraint joint single buyer-single vendor inventory problem with changeable lead time and (r, Q) policy in supply chains with stochastic demand. Chen and Dye [44] solved an inventory problem with deteriorating products and variable demands using a PSO algorithm. Further, Taleizadeh et al. [27] modeled a chance–constraint supply chain problem with uniformly distributed stochastic demand, where an Ant Colony Bee and a PSO algorithm were utilized to solve the problem.

Instead of optimizing a single objective, some researchers tried to find Pareto front solutions for their multiple objective inventory planning problems that usually consist of multiple conflicting objectives. Agrell [45] proposed a multicriteria framework for inventory control problem, in which the solution procedure was an interactive method with preferences extracted gradually in decision analysis process to determine batch size and security stock. Roy and Maiti [28] presented a multiobjective inventory model of deteriorating items with stock-dependent demand under limited imprecise storage area and total cost budget. Tsou [46] developed a multiobjective reorder point and order size system and proposed a multiobjective PSO (MOPSO) to generate Pareto front solutions. He employed TOPSIS to sort the nondominated solutions. The objectives therein were to maximize the profit and to minimize the wastage cost where the profit goal, wastage cost, and storage area were fuzzy in nature. One of the successful applications of PSO to MOOPs is the seminal work of Coello Coello and Lechuga [47]. Yaghin et al. [29] first addressed an inventory-marketing system to determine the production lot size, marketing expenditure, and selling prices in which the model was formulated as a fuzzy nonlinear multiobjective program. Then, they converted the model to a classical single-objective one by a fuzzy goal programming method where an efficient solution procedure using PSO was provided to solve the resulting nonlinear problem. In their study, MOPSO is not only a viable alternative to solve MOOPs, but also the only one, compared with the nondominated sorting genetic algorithm-II (NSGA-II) [48], the Pareto archive evolutionary strategy (PAES) [49], and the microgenetic algorithm [50] for multiobjective optimization problems [51]. Table 1 shows the literature review of the works reviewed in this work where DOE is an abbreviation of term "design of experiments."

In this research, the contribution of the problem is considering a new biobjective multi-item multiperiodic inventory control model where some items are purchased under AUD and the other items are bought from IQD. The demands vary in different periods, the budget is limited, the orders are placed in batch sizes, and shortages in combination of backorder and lost sale are considered. The goal is to find the optimum inventory levels of the items in each period such that the total inventory cost and the total required warehouse space are minimized simultaneously. Since it is not easy for the managers to allocate the crisp values to the weights of the objectives in a decision making process, considering these weights as fuzzy numbers will be taken as an advantage.

In order to be more understanding of the problem, we try to explain the model with taking an example in the real world. We consider a company which produces some kinds of fashion clothes including trousers, t-shirt, and shirt in a certain period. The customers (wholesales) of this company with different demand rates make the orders and receive their products in the prespecific boxes, each one consisting of a known number of these clothes. Moreover, due to some unforeseen matters, such as production limitation, the

Table 1: Literature review of the related works.

References	Multiproduct	Multiperiod	Fuzzy multiobjective	Discount policy	Solving methodology	Shortages	DOE
[1]	—	√	—	—	B and B	—	—
[2]	—	—	—	—	Level-crossing	√	—
[3]	—	√	—	IQD	Numerical methods	—	—
[4]	√	√	—	IQD	GA	—	—
[5]	√	√	—	—	CPLEX	—	—
[6]	—	√	—	—	—	√	—
[7]	—	—	—	IQD and AUD	Simulation	—	—
[8]	—	—	—	AUD	Simulation	—	—
[9]	—	—	—	IQD	Numerical methods	—	—
[10]	—	—	—	IQD	Numerical methods	—	—
[11]	√	—	—	IQD and AUD	GA	—	—
[12]	√	—	—	AUD	Numerical methods	—	—
[13]	√	—	—	IQD and AUD	GA	√	—
[14]	—	—	—	IQD	Yager ranking	—	—
[15]	—	—	—	IQD	Excel macro	—	—
[16]	—	—	√	—	TOPSIS and GA	√	—
[17]	√	—	—	—	ACO	—	—
[18]	—	—	—	—	Tabu search and Lagrangian	—	—
[19]	—	√	—	—	Tabu search	—	—
[20]	—	—	—	IQD	Goal programming and GA	√	—
[21]	√	—	—	IQD	GA and fuzzy simulation	√	—
[22]	√	—	—	—	GA	√	—
[23]	√	—	—	—	SA and GA	√	—
[24]	—	—	√	—	TOPSIS and GA	√	—
[25]	√	—	—	IQD	fuzzy simulation	√	—
[26]	√	—	—	—	Harmony search	√	—
[27]	√	—	—	—	Bee colony and PSO	√	—
[28]	√	—	√	—	Fuzzy programming algorithm	—	—
[29]	—	√	√	—	Fuzzy method	—	—
This research	√	√	√	√	PSO and GA	√	Taguchi

companies are not responsive to all of the demands in a period and hence some customers must wait until the next period to receive their orders. Furthermore, it is assumed the company is going to extend the production part and therefore the owner has a plan to build and optimize a new storage subject to the available space.

The remainder of the paper is organized as follows. In Section 2, the problem along with its assumptions is defined. In Section 3, the defined problem of Section 2 is modeled. To do this, the parameters and the variables of the problem are first introduced. A MOPSO algorithm is presented in Section 4 to solve the model. Section 5 contains a numerical example for a problem with 5 items and 3 periods, for which a multiobjective genetic algorithm (MOGA) is also applied as benchmark for comparisons. Finally, conclusion and recommendations for future research comes in Section 6.

2. Problem Definition, Assumptions, and Notations

Consider a biobjective multi-item multiperiod inventory control problem, in which an AUD policy is used for some items and an IQD policy for some other items. The inventory control problem of this research is similar to the seasonal items problem where the planning horizon starts in a period (or season) and finish in a certain period (or season). The total available budget in the planning horizon is limited and fixed. Due to existing ordering limitations or production constraints, the order quantities of all items in different periods cannot be more than their predetermined upper bounds. The demands of the products are constant and distinct, and, in case of shortage, a fraction is considered backorder and a fraction lost sale. The costs associated with the inventory control system are holding, backorder, lost sale, and purchasing costs. Moreover, due to current managerial decision adaptations on production policies (i.e., setting up a new manufacturing line, extending the warehouse, or building a new storage area), minimizing the total storage space is required as well as minimizing the total inventory costs. Therefore, the goal is to identify the inventory levels of the items in each period such that the two objective functions, total inventory costs and total storage space, are minimized.

In order to simplify the modeling, the following assumptions are set to the problem at hand.

(1) The demand rate of an item is independent of the others and is constant in a period. However, it can be different in different periods.

(2) At most one order can be placed in a period. This order can include or exclude an item.

(3) The items are delivered in a special container. Thus, the order quantities must be a multiple of a fixed-sized batch.

(4) The vendor uses an AUD policy for some items and an IQD policy for others.

(5) A fraction of the shortages is considered backorder and a fraction lost sale.

(6) The initial inventory level of all items is zero.

(7) The budget is limited.

(8) The planning horizon is finite and known. In the planning horizon, there are N periods of equal duration.

(9) The order quantity on an item in a period is greater than or equal to its shortage quantity in the previous period (i.e., $Q_{i,j+1} \geq b_{i,j}$ defined below).

In order to model the problem at hand, in what comes next we first define the variables and the parameters. Then, the problem is formulated in Section 3.

For $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, N-1$ and $k = 1, 2, \ldots, K$ the variables and the parameters of the model are defined as follows:

N: number of replenishment cycles during the planning horizon,

m: number of items,

K: number of price break points,

S_i: required storage space per unit of the ith product,

T_j: total time elapsed up to and including the jth replenishment cycle,

$T'_{i,j}$: jth period in which the inventory level of item i is zero (a decision variable),

B_i: batch size of the ith product,

$V_{i,j}$: number of the packets for the ith product order in period j (a decision variable),

$D_{i,j}$: demand of the ith product in period j,

$Q_{i,j}$: purchase quantity of item i in period j (a decision variable where $Q_{i,j} = B_i V_{i,j}$),

A_i: ordering cost per replenishment of product i (If an order is placed for one or more items in period j, this cost appears in that period),

$b_{i,j}$: shortage quantity of the ith product in period j (a decision variable),

$X_{i,j}$: the beginning positive inventory level of the ith product in period j (in $j = 1$, the beginning positive inventory level of all items is zero) (a decision variable),

$I_{i,j}$: inventory position of the ith product in period j (it is $X_{i,j+1} + Q_{i,j+1}$, if $I_{i,j} \geq 0$, otherwise equals $b_{i,j}$),

$I_i(t)$: the inventory level of the ith item at time t,

H_i: unit inventory holding cost for item i,

$q_{i,k}$: kth discount point for the ith product ($q_{i,1} = 0$),

$m_{i,k}$: discount rate of item i in kth price break point ($m_{i,1} = 0$),

P_i: purchasing cost per unit of the ith product,

$P_{i,k}$: purchasing cost per unit of the ith product at the kth price break point,

$U_{i,j,k}$: a binary variable, set 1 if item i is purchased at price break point k in period j, and 0 otherwise,

$W_{i,j}$: a binary variable, set 1 if a purchase of item i is made in period j, and 0 otherwise,

$L_{i,j}$: a binary variable, set 1 if a shortage for item i occurs in period j, and 0 otherwise,

β_i: percentage of unsatisfied demands of the ith product, that is, back ordered,

$\pi_{i,j}$: backorder cost per unit demand of the ith product in period j,

$\widehat{\pi}_{i,j}$: shortage cost per unit of the ith product in period j, that is, lost,

Z_1: total inventory cost,

Z_2: total storage space,

TB: total available budget,

M_1: an upper bound for order quantity of the ith item in period j,

M_2: an upper bound for order quantities of all items in each period (the truck capacity),

TMF: objective function (the weighted combination of the total inventory cost and the total storage space),

w_1: a weight associated with the total inventory cost ($0 \leq w_1 \leq 1$),

w_2: a weight associated with the total storage space ($0 \leq w_2 \leq 1$).

3. Problem Formulation

A graphical representation of the inventory control problem at hand with 5 periods for item i is given in Figure 1 to obtain the inventory costs. At the beginning of the primary period (T_0), it is assumed the starting inventory level of item i is zero and that the order quantity has been received and is available. In the following periods, shortages can either occur or not. If shortage occurs, the corresponding binary variable is 1, otherwise it is zero. In the latter case, the inventory levels at the beginning of each period may be positive.

3.1. The Objective Functions. The first objective function of the problem, the total inventory cost, is obtained as

$$\begin{aligned} Z_1 &= \text{Total Inventory Cost} \\ &= \text{Total Ordering Cost} + \text{Total Holding Cost} \\ &\quad + \text{Total Shortage Cost} + \text{Total Purchasing Cost}, \end{aligned} \tag{1}$$

where each part is derived as follows.

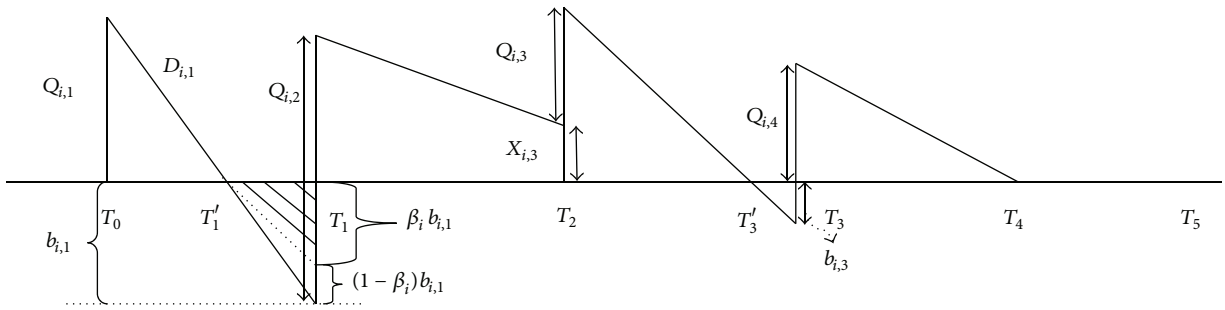


Figure 1: Some possible situations for the inventory of item i in 5 periods.

The ordering cost of an item in a period occurs when an order is placed for it in that period. Using a binary variable $W_{i,j}$, where it is 1 if an order for the ith product in period j is placed and zero otherwise, and knowing that orders can be placed in periods 1 to $N-1$ the total ordering cost is obtained as

$$\text{Total Ordering Cost} = \sum_{i=1}^{m}\sum_{j=1}^{N-1} A_i W_{i,j}. \tag{2}$$

Since it is assumed a shortage may occur for a product in a period or not, the holding cost derivation is not as straightforward as the ordering cost derivation. Taking advantage of a binary variable $L_{i,j}$, where it is 1 if a shortage for item i in period j occurs and otherwise zero, and using Figure 1, the holding cost for item i in the time interval $T_{j-1} \le t \le T_j(1-L_{i,j}) + T'_{i,j}L_{i,j}$ is obtained as

$$H_i \int_{T_{j-1}}^{T_j(1-L_{i,j})+T'_{i,j}L_{i,j}} I_i(t)\,dt, \tag{3}$$

where $I_i(t)$ is the inventory level of the ith item at time t. In (3), if a shortage for item i occurs, $L_{i,j}$ becomes 1 and the term $T_j(1-L_{i,j}) + T'_{i,j}L_{i,j}$ becomes $T'_{i,j}$. Otherwise, $L_{i,j} = 0$ and $T_j(1-L_{i,j})+T'_{i,j}L_{i,j} = T_j$. In Figure 1, the trapezoidal area above the horizontal timeline in each period when multiplied by the unit inventory holding cost of an item, H_i, represents the holding cost of the item in that period. In other word, since

$$I_{i,j+1} = I_{i,j} + Q_{i,j} - D_{i,j} \tag{4}$$

and if $I_{i,j+1} \ge 0$ then $I_{i,j+1} = X_{i,j+1}$, otherwise $I_{i,j+1} = b_{i,j}$, (3) becomes

$$H_i \int_{T_{j-1}}^{T_j(1-L_{i,j})+T'_{i,j}L_{i,j}} I_i(t)\,dt = \frac{X_{i,j}+Q_{i,j}-D_{i,j}}{2}\left(T_j\left(1-L_{i,j}\right) + T'_{i,j}L_{i,j} - T_{j-1}\right)H_i. \tag{5}$$

Therefore, the total holding cost is obtained in

$$\text{Total Holding Cost} = \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{X_{i,j}+Q_{i,j}+X_{i,j+1}}{2}\right) \times \left(T_j\left(1-L_{i,j}\right) + T'_{i,j}L_{i,j} - T_{j-1}\right)H_i. \tag{6}$$

The total shortage cost consists of two parts: the total backorder cost and the total lost sale cost. In Figure 1, the trapezoidal area underneath the horizontal timeline in each period (shown for the primary period) when multiplied by the backorder cost per unit demand of the ith product in period j, $\pi_{i,j}$, is equal to the backorder cost of the item in that period. Therefore, the total backorder cost will be

$$\text{Total Backorder Cost} = \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{\pi_{i,j}b_{i,j}}{2}\left(T_j - T'_{i,j}\right)\beta_i\right). \tag{7}$$

Furthermore, since $(1-\beta_i)$ represents the percentage demands of the ith product, that is, lost sale, the total lost sale becomes

$$\text{Total Lost Sale Cost} = \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{\hat{\pi}_{i,j}b_{i,j}}{2}\left(T_j - T'_{i,j}\right)(1-\beta_i)\right) \tag{8}$$

in which $T_j - T'_{i,j} = b_{i,j}/D_{i,j}$.

The total purchase cost also consists of two AUD and IQD costs. The purchasing offered by AUD policy is modeled by

$$P_i = \begin{cases} P_{i,1}; & 0 < Q_{i,j} \le q_{i,2} \\ P_{i,2}; & q_{i,2} < Q_{i,j} \le q_{i,3} \\ \vdots & \\ P_{i,K}; & q_{i,K} < Q_{i,j}. \end{cases} \tag{9}$$

Hence, the purchasing cost of this policy is obtained as

$$\text{AUD Purchasing Cost} = \sum_{i=1}^{m}\sum_{j=1}^{N-1}\sum_{k=1}^{K} P_{i,k}Q_{i,j}U_{i,j,k}. \tag{10}$$

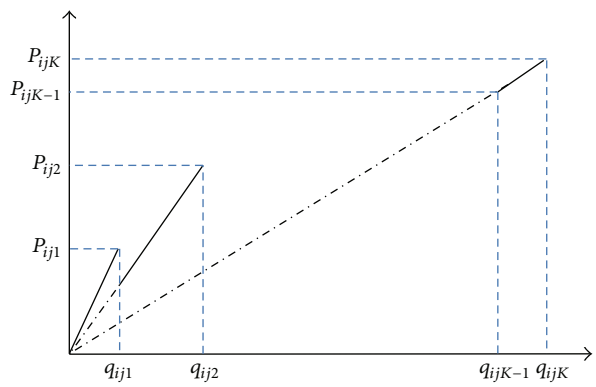


Figure 2: AUD policy for purchasing the products in different periods.

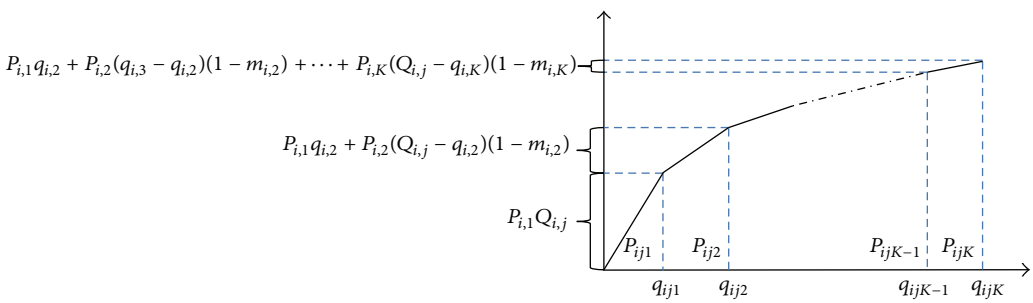


Figure 3: IQD policy for purchasing the products in different periods.

A graphical representation of the AUD policy employed to purchase the products in different periods is shown in Figure 2. In this Figure, the relation between the price break points and the purchasing costs is demonstrated clearly. Moreover, $U_{i,j,k}$ is a binary variable, set 1 if the ith item is purchased with price break k in period j and 0 otherwise.

In the IQD policy, the purchasing cost per unit of the ith product depends on its order quantity. Therefore, for each price break point we have

$$\begin{aligned} &P_{i,1}Q_{i,j}; \quad 0 < Q_{i,j} \le q_{i,2} \\ &P_{i,1}q_{i,2} + P_{i,2}\left(Q_{i,j} - q_{i,2}\right)\left(1 - m_{i,2}\right); \quad q_{i,2} < Q_{i,j} \le q_{i,3} \\ &\vdots \\ &P_{i,1}q_{i,2} + P_{i,2}\left(q_{i,3} - q_{i,2}\right)\left(1 - m_{i,2}\right) \\ &\quad + \cdots + P_{i,K}\left(Q_{i,j} - q_{i,K}\right)\left(1 - m_{i,K}\right); \\ &\quad q_{i,K} < Q_{i,j}. \end{aligned} \tag{11}$$

Hence, the total purchasing cost under the IQD policy is obtained as

IQD Purchasing Cost

$$= \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left\{\left(Q_{i,j} - q_{i,K}\right)P_iU_{i,j,K}\left(1 - m_{i,K}\right) + \sum_{k=1}^{K-1}\left(q_{i,k+1} - q_{i,k}\right)P_i\left(1 - m_{i,k}\right)\right\}. \tag{12}$$

Figure 3 graphically depicts the IQD policy for each product in different periods.

Thus, the first objective function of the problem at hand becomes

$$\begin{aligned} Z_1 = &\sum_{i=1}^{m}\sum_{j=1}^{N-1}A_iW_{i,j} + \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{X_{i,j} + Q_{i,j} + X_{i,j+1}}{2}\right) \\ &\times\left(T_j\left(1 - L_{i,j}\right) + T'_{i,j}L_{i,j} - T_{j-1}\right)H_i \\ &+ \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{\pi_{i,j}b_{i,j}}{2}\left(T_j - T'_{i,j}\right)\beta_i\right) \\ &+ \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left(\frac{\widehat{\pi}_{i,j}b_{i,j}}{2}\left(T_j - T'_{i,j}\right)\left(1 - \beta_i\right)\right) \\ &+ \sum_{i=1}^{m}\sum_{j=1}^{N-1}\sum_{k=1}^{K}Q_{i,j}P_{i,k}U_{i,j,k} \\ &+ \sum_{i=1}^{m}\sum_{j=1}^{N-1}\left\{\left(Q_{i,j} - q_{i,K}\right)P_iU_{i,j,K}\left(1 - m_{i,K}\right) \right. \\ &\qquad \left. + \sum_{k=1}^{K-1}\left(q_{i,k+1} - q_{i,k}\right)P_i\left(1 - m_{i,k}\right)\right\}. \end{aligned} \tag{13}$$

The second objective of the problem is to minimize the total required storage space. Since in each period, order quantities $Q_{i,j}$ enter the storage and the beginning inventory

of a period is the remainder inventory of the previous period, $X_{i,j}$, the second objective function of the problem is modeled by

$$Z_2 = \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left(X_{i,j} + Q_{i,j}\right) S_i. \tag{14}$$

Finally, the fitness function is defined as the weighted combination of the total inventory cost and the required storage space as

$$\text{TMF} = w_1 Z_1 + w_2 Z_2. \tag{15}$$

3.2. The Constraints. In real-world inventory planning problems, due to existing constraints on either supplying or producing goods (e.g., budget, labor, production, carrying equipment, and the like), objectives are not met simply. This section presents formulations for some real-world constraints.

The first limitation is given in (4), where it relates the beginning inventory of the items in a period to the beginning inventory of the items in the previous period plus the order quantity of the previous period minus the demand of the previous period.

The second limitation is due to delivering the items in packets of batches. Since $Q_{i,j}$ represents the purchase quantity of item i in period j; denoting the batch size by B_i and the number of packets by $V_{i,j}$, we have

$$Q_{i,j} = B_i V_{i,j}. \tag{16}$$

Furthermore, since $Q_{i,j}$ can only be purchased based on one price break point, the following constraint must hold:

$$\sum_{k=1}^{K} U_{i,j,k} = 1. \tag{17}$$

The prerequisite of using this strategy is that the lowest $q_{i,k}$ in the AUD table must be zero (i.e., $q_{i,1} = 0$).

Since the total available budget is TB, the unit purchasing cost of the product is P_i, and the order quantity is $Q_{i,j}$, the budget constraint will be

$$\sum_{i=1}^{m} \sum_{j=1}^{N-1} Q_{i,j} P_i \le \text{TB}. \tag{18}$$

In real-world environments, the order quantity of an item in a period may be limited. Defining M_1 an upper bound for this quantity, for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, N-1$ we have

$$Q_{i,j} \le M_1. \tag{19}$$

Moreover, due to transportation contract and the truck capacity, the number of product orders and the total order quantities in a period are limited as well. Hence, for $j = 1, 2, \ldots, N-1$, we have

$$\sum_{i=1}^{m} Q_{i,j} W_{i,j} \le M_2, \tag{20}$$

where if an order occurs for item i in period j, $W_{i,j} = 1$, otherwise $W_{i,j} = 0$. Further, M_2 is an upper bound on the total number of orders and the total order quantities in a period.

As a result, the complete mathematical model of the problem is

$$\text{Min} \quad \text{TMF} = w_1 Z_1 + w_2 Z_2 \tag{21}$$

subject to

$$\begin{aligned}
Z_1 = {} & \sum_{i=1}^{m} \sum_{j=1}^{N-1} A_i W_{i,j} + \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left(\frac{X_{i,j} + Q_{i,j} + X_{i,j+1}}{2} \right) \\
& \times \left(T_j \left(1 - L_{i,j}\right) + T'_{i,j} L_{i,j} - T_{j-1} \right) H_i \\
& + \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left(\frac{\pi_{i,j} b_{i,j}}{2} \left(T_j - T'_{i,j}\right) \beta_i \right) \\
& + \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left(\frac{\widehat{\pi}_{i,j} b_{i,j}}{2} \left(T_j - T'_{i,j}\right) (1 - \beta_i) \right) \\
& + \sum_{i=1}^{m} \sum_{j=1}^{N-1} \sum_{k=1}^{K} Q_{i,j} P_{i,k} U_{i,j,k} \\
& + \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left\{ \left(Q_{i,j} - q_{i,K}\right) P_i U_{i,j,K} \left(1 - m_{i,K}\right) \right. \\
& \qquad \left. + \sum_{k=1}^{K-1} \left(q_{i,k+1} - q_{i,k}\right) P_i \left(1 - m_{i,k}\right) \right\},
\end{aligned}$$

$$Z_2 = \sum_{i=1}^{m} \sum_{j=1}^{N-1} \left(X_{i,j} + Q_{i,j}\right) S_i,$$

$$I_{i,j+1} = I_{i,j} + Q_{i,j} - D_{i,j};$$

$$(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1),$$

$$I_{i,j+1} = \begin{cases} X_{i,j+1}; & I_{i,j+1} \ge 0 \\ b_{i,j}; & I_{i,j+1} < 0; \end{cases}$$

$$(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1),$$

$$Q_{i,j} = B_i V_{i,j};$$

$$(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1),$$

$$\sum_{k=1}^{K} U_{i,j,k} = 1;$$

$$(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1),$$

$$\sum_{i=1}^{m} \sum_{j=1}^{N-1} Q_{i,j} P_i \le \text{TB}$$

$$Q_{i,j} \le M_1;$$

$$
\begin{gathered}
(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1) \\
W_{i,j} \in \{0, 1\}; \quad (j = 1, 2, \ldots, N-1) \\
U_{i,j,k} \in \{0, 1\}; \\
(i = 1, 2, \ldots, m), \quad (j = 1, 2, \ldots, N-1), \ (k = 1, 2, \ldots, K) \\
\sum_{i=1}^{m} Q_{i,j} W_{i,j} \le M_2; \quad (j = 1, 2, \ldots, N-1) \\
Q_{i,j+1} \ge b_{i,j}.
\end{gathered} \tag{22}
$$

In most inventory-planning models that have been developed so far, researchers have imposed some unrealistic assumptions such that the objective function of the model becomes concave and the model can easily be solved by some mathematical approaches like the Lagrangian or the derivative methods. However, since the model in (22), which is obtained based on assumptions that are more realistic, is an integer nonlinear programming mixed with binary variables, reaching an analytical solution (if any) to the problem is difficult. In addition, efficient treatment of integer nonlinear optimization is one of the most difficult problems in practical optimization [52]. As a result, in the next section a metaheuristic algorithm is proposed to solve the model in (22).

4. The Proposed Multiobjective Particle Swarm Optimization Algorithm

Many researchers have successfully used metaheuristic methods to solve complicated optimization problems in different fields of scientific and engineering disciplines; among them, the particle swarm optimization (PSO) algorithm is one of the most efficient methods. That is why this approach is taken in this research to solve the model in (22). The structure of the proposed MOPSO that is based on the PSO algorithm for the multiobjective inventory planning problem at hand is given as follows.

4.1. Generating and Initializing the Particles Positions and Velocities. PSO is initialized by a group of random particles (solutions) called generation and then searches for optima by updating generations. The initial population is constructed by randomly generated R particles (similar to the chromosomes of a genetic algorithm). In a d-dimensional search space, let $\vec{x}_k^i = \{x_{k,1}^i, x_{k,2}^i, \ldots, x_{k,d}^i\}$ and $\vec{v}_k^i = \{v_{k,1}^i, v_{k,2}^i, \ldots, v_{k,d}^i\}$ be, respectively, the position and the velocity of particle i at time k. Then, (23) is applied to generate initial particles, in which $x_{\min}$ and $x_{\max}$ are the lower and the upper bounds on the design variable values and RAND is a random number between 0 and 1. Consider

$$
\begin{aligned}
x_0^i &= x_{\min} + \mathrm{RAND}\left(x_{\max} - x_{\min}\right) \\
v_0^i &= x_{\min} + \mathrm{RAND}\left(x_{\max} - x_{\min}\right).
\end{aligned} \tag{23}
$$

An important note for the generating and initializing step of the PSO is that solutions must be feasible and must satisfy the constraints. As a result, if a solution vector does not satisfy a constraint, the related vector solution will be penalized by a big penalty on its fitness.

4.2. Selecting the Best Position and Velocity. For every particle, denote the best solution (fitness) that has been achieved so far as

$$
\overrightarrow{p\mathrm{best}_k^i} = \left\{p\mathrm{best}_{k,1}^i, p\mathrm{best}_{k,2}^i, \ldots, p\mathrm{best}_{k,d}^i\right\}, \tag{24}
$$

$$
\overrightarrow{g\mathrm{best}_k^i} = \left\{g\mathrm{best}_{k,1}^i, g\mathrm{best}_{k,2}^i, \ldots, g\mathrm{best}_{k,d}^i\right\}, \tag{25}
$$

where $\overrightarrow{p\mathrm{best}_k^i}$ in (25) is the best position already found by particle i until time k and $\overrightarrow{g\mathrm{best}_k^i}$ in (24) is the best position already found by a neighbor until time k.

4.3. Velocity and Position Update. The new velocities and positions of the particles for the next fitness evaluation are calculated using [53, 54]

$$
\begin{aligned}
v_{k+1,d}^i &= w \cdot v_{k,d}^i + C_1 \cdot r_1 \cdot \left(p\mathrm{best}_{k,d}^i - x_{k,d}^i\right) \\
&\quad + C_2 \cdot r_2 \cdot \left(g\mathrm{best}_{k,d}^i - x_{k,d}^i\right), \\
x_{k+1,d}^i &= x_{k,d}^i + v_{k+1,d}^i,
\end{aligned} \tag{26}
$$

where r_1 and r_2 are random numbers between 0 and 1, coefficients C_1 and C_2 are given acceleration constants towards $\overrightarrow{p\mathrm{best}}$ and $\overrightarrow{g\mathrm{best}}$, respectively, and w is the inertia weight. Introducing a linearly decreasing inertia weight into the original PSO significantly improves its performance through the parameter study of inertia weight [55, 56]. Moreover, the linear distribution of the inertia weight is expressed as follows [55]:

$$
w = w_{\max} - \frac{w_{\max} - w_{\min}}{\mathrm{iter\_max}} \mathrm{iteration}, \tag{27}
$$

where iter_max is the maximum number of iterations and iteration is the current number of iteration. Equation (27) presents how the inertia weight is updated, considering $w_{\max}$ and $w_{\min}$ are the initial and the final weights, respectively. The parameters $w_{\max} = 0.9$ and $w_{\min} = 0.4$ that were previously investigated by Naka et al. [55] and Shi and Eberhart [56] are used in this research as well.

4.4. Stopping Criterion. Achieving a predetermined solution, steady-state mean, and standard deviations of a solution in several consecutive generations, stopping the algorithm at a certain computer CPU time, or stopping when a maximum number of iterations is reached are usual stopping rules that have been used so far in different research works. In current research, the PSO algorithm stops when the maximum number of iterations is reached.

Pseudocode 1 shows the pseudocode of the proposed MOPSO algorithm. Moreover, since the problem and hence

```
for i = 1 to Pop
    initialize position (i)
    initialize velocity (i)
    if position (i) and velocity (i) be a feasible candidate solution
        penalty = 0
    else penalty = a positive number
    end if
end for
w = [0.4, 0.9]
do while Iter <= Gen
    for j = 1 to Pop
        Calculate new velocity of the particle
        Calculate new position of the particle
        pbest(iter) = min(pbest(i))
    end for
    gbest(iter) = min(gbest)
    w = w_max − ((w_max − w_min)/iter_max) × iter
    modifying the velocity and position of the particle
end while
```

Pseudocode 1: The pseudocode of MOPSO algorithm.

```
Begin
Set P_C, P_m, Pop and Gen
S → 0
initialize Population (S)
evaluate Population (S)
while (non terminating condition)
repeat
S → S + 1
select Population (S) from Population with roulette wheel (S − 1)
uniform crossover
one point mutation
evaluate Population (S)
end repeat
print optimum result
end
```

Pseudocode 2: The pseudocode of MOGA algorithm.

the model is new and there is no other available algorithm to compare the results, a multiobjective genetic algorithm (MOGA) is developed in this research for validation and benchmarking. MOGA was coded using roulette wheel in selection operator, population size of 40, uniform crossover with probability of 0.64, one-point random mutation with probability 0.2, and a maximum number of 500 iterations. Pseudocode 2 shows the pseudocode of the proposed MOGA algorithm. The computer programs of the MOPSO and MOGA algorithms were developed in MATLAB software and are executed on a computer with 2.50 GHz of core 2 CPU and 3.00 GB of RAM. Furthermore, all the graphical and statistical analyses are performed in MINITAB 15.

In the next section, some numerical examples are given to illustrate the application of the proposed MOPSO algorithm in real-world environments and to evaluate and compare its performances with the ones obtained by a MOGA method.

5. Numerical Illustrations

The decision variables in the inventory model (22) are $Q_{i,j}$, $X_{i,j}$, $V_{i,j}$, and $b_{i,j}$. We note that the determination of the order quantity of the items in different periods, that is, $Q_{i,j}$, results in the determination of the other decision variables as well. Hence, we first randomly generate $Q_{i,j}$, that is, modeled by the particles' position and velocity. Equation (28) shows a pictorial representation of the matrix Q for a problem with 4 items in 4 periods, where rows and columns correspond to the items and the periods, respectively.

The structure of a particle

$$Q_{4,4} = \begin{bmatrix} 124 & 116 & 50 & 0 \\ 205 & 190 & 58 & 0 \\ 114 & 68 & 107 & 0 \\ 43 & 87 & 210 & 0 \end{bmatrix}. \tag{28}$$

Table 2: Different problems and their optimal TMF values obtained by the two algorithms.

Problem	m	N	M_1	TB	Objective values		MOPSO				MOGA			
					MOPSO	MOGA	C_1	C_2	Pop	Gen	P_C	P_m	Pop	Gen
1	1	3	2000	30000	18510	18510	2	2.5	30	200	0.6	0.1	40	500
2	2	3	3000	85000	44622	44943	1.5	2.5	20	200	0.5	0.08	30	200
3	3	3	5000	170000	85830	86089	1.5	2	30	500	0.6	0.08	50	300
4	2	4	3500	130000	92758	93423	2	2	30	100	0.7	0.2	50	200
5	3	4	5000	240000	147480	146450	1.5	2.5	40	100	0.6	0.2	50	200
6	4	3	6000	260000	132910	133400	2.5	1.5	20	200	0.6	0.08	40	500
7	5	3	9000	370000	153840	154550	2	2.5	40	200	0.6	0.2	50	200
8	6	3	8800	360000	214620	215510	1.5	2.5	30	500	0.5	0.1	30	200
9	7	3	10500	400000	265020	266020	2	2	30	200	0.7	0.1	40	300
10	8	5	12000	470000	322850	328250	2.5	2.5	30	500	0.6	0.08	50	200
11	8	8	15000	550000	431245	442100	2	2	20	200	0.7	0.2	30	300
12	9	5	15000	530000	438790	449835	1.5	2.5	40	100	0.7	0.2	50	300
13	9	8	18000	600000	495470	513725	2	2	40	500	0.7	0.08	50	500
14	9	9	20000	630000	553276	571250	1.5	2.5	20	100	0.5	0.1	40	500
15	10	5	20000	620000	579030	593167	2.5	2.5	20	200	0.5	0.2	30	200
16	10	8	25000	700000	642870	665890	1.5	2.5	40	200	0.7	0.08	30	200
17	10	10	34000	780000	710035	731280	1.5	1.5	30	200	0.5	0.08	50	500
18	10	15	45000	900000	823210	852400	2	2.5	20	500	0.6	0.1	40	500
19	11	10	40000	850000	827659	859080	1.5	2.5	30	500	0.6	0.1	30	500
20	11	15	45000	870000	867840	897500	2.5	2	40	100	0.7	0.2	50	200
21	12	10	48000	870000	902720	948956	1.5	2.5	30	200	0.5	0.2	50	300
22	12	12	53000	900000	932760	974380	1.5	2	30	100	0.6	0.2	40	200
23	12	15	58000	950000	965470	1023950	2	2.5	20	100	0.5	0.2	30	200
24	13	10	52000	890000	973200	1043569	1.5	1.5	20	500	0.7	0.08	40	300
25	13	13	55000	930000	985439	1089210	1.5	2.5	40	500	0.6	0.1	30	300
26	13	15	62000	980000	1056810	1104325	2	2.5	40	500	0.5	0.1	40	500
27	15	8	57000	900000	1059835	1110360	2	2	20	200	0.6	0.08	50	500
28	15	10	63000	900000	1095430	1176509	2.5	2.5	30	100	0.6	0.2	30	200
29	15	12	68000	950000	1198720	1332900	2.5	1.5	30	500	0.5	0.1	40	200
30	15	15	75000	1000000	1256980	1447905	1.5	2	20	500	0.6	0.08	50	200
31	16	12	70000	940000	1298750	1473400	2	1.5	40	100	0.7	0.1	50	500
32	16	15	80000	1050000	1454328	1772349	1.5	2.5	40	500	0.7	0.2	40	200
33	17	15	87000	1100000	1543890	1809850	2	2	40	200	0.7	0.1	50	500
34	17	17	93000	1140000	1630215	1865780	2.5	2.5	30	500	0.6	0.1	40	300
35	18	10	80000	1000000	1678345	1890437	1.5	1.5	30	100	0.7	0.08	50	300
36	18	15	98000	1150000	1768950	1924670	2	2.5	20	500	0.5	0.08	40	300
37	18	18	103000	1230000	1832450	1987320	1.5	2	30	100	0.7	0.1	40	200
38	20	10	100000	1200000	1876895	1998230	2.5	2.5	20	500	0.6	0.1	50	500
39	20	15	108000	1260000	1904564	2035689	1.5	2	30	500	0.7	0.2	50	500
40	20	20	115000	1330000	1987350	2154670	1.5	1.5	30	100	0.7	0.08	30	500
Mean	—	—	—	—	**897145**	**971541**	—	—	—	—	—	—	—	—
St. Dev	—	—	—	—	**581013**	**652778**	—	—	—	—	—	—	—	—

Table 2 shows partial data for 40 different problems with different sizes along with their near optimal solutions obtained by MOPSO and MOGA. In these problems, the number of items varies between 1 and 20 and the number of periods takes values between 3 and 15. In addition, the total available budgets and the upper bounds for the order quantities (M_1) are given in Table 1 for each problem.

In order to illustrate how the results are obtained, consider a typical problem with 5 items and 3 periods (the seventh row in Table 2), for which the complete input data is given in Table 3. The parameters of the MOPSO and MOGA algorithms are set by Taguchi method where C_1, C_2 the number of populations (Pop) and number of generations (Gen) are the parameters of MOPSO and crossover probability and their level values are shown in Table 4. Furthermore, the rest of MOPSO's parameters are set as $w_{\min} = 0.4, w_{\max} = 0.9$ and the time periods $T_j = 3$ for $j = 0, 1, 2, 3$. The above parameter settings are obtained performing intensive runs. Furthermore, the amount of $V_{i,j}$ will be obtained automatically after gaining the order quantity $Q_{i,j}$.

Table 3: The general data for a problem with 5 items and 3 periods.

Product	$D_{i,1}$	$D_{i,2}$	$\pi_{i,1}$	$\pi_{i,2}$	$\hat{\pi}_{i,1}$	$\hat{\pi}_{i,2}$	B_i	H_i	A_i	β_i	S_i
1	1200	800	20	18	9	10	3	5	20	0.5	4
2	1300	900	20	18	9	10	7	5	15	0.5	6
3	1500	1200	11	14	8	12	5	6	25	0.8	7
4	2100	2000	11	14	8	12	8	6	18	0.8	5
5	1800	1600	12	15	9	11	7	7	19	0.6	6

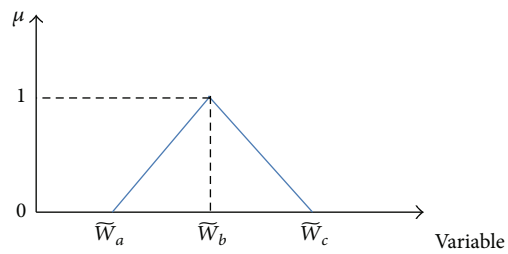


Figure 4: The triangular fuzzy numbers.

Table 4: The parameters of the two algorithms and their levels.

Algorithms	Factors	Levels [1, 2, 3]
MOPSO	C_1 (A)	[1.5, 2, 2.5]
	C_2 (B)	[1.5, 2, 2.5]
	Pop(C)	[20, 30, 40]
	Gen(D)	[100, 200, 500]
MOGA	P_C (A)	[0.5, 0.6, 0.7]
	P_m (B)	[0.08, 0.1, 0.2]
	Pop(C)	[30, 40, 50]
	Gen(D)	[200, 300, 500]

The weights associated with the objectives are as triangular fuzzy number $\widetilde{w} = [\widetilde{w}_a, \widetilde{w}_b, \widetilde{w}_c]$ shown in Figure 4 where membership function of variable x is given by

$$\widetilde{\mu}(x) = \begin{cases} 0 & x < \widetilde{w}_a \\ \dfrac{x - \widetilde{w}_a}{\widetilde{w}_b - \widetilde{w}_a} & \widetilde{w}_a < x < \widetilde{w}_b \\ \dfrac{\widetilde{w}_c - x}{\widetilde{w}_c - \widetilde{w}_b} & \widetilde{w}_b < x < \widetilde{w}_c \\ 0 & \widetilde{w}_c < x. \end{cases} \tag{29}$$

Now, in order to get crisp interval by α-cut operation, interval $\widetilde{w}_\alpha$ can be obtained as follows ($\forall \alpha \in [0, 1]$):

$$\frac{\widetilde{w}_a^{(\alpha)} - \widetilde{w}_a}{\widetilde{w}_b - \widetilde{w}_a} = \alpha, \qquad \frac{\widetilde{w}_c - \widetilde{w}_c^{(\alpha)}}{\widetilde{w}_c - \widetilde{w}_b} = \alpha. \tag{30}$$

We have

$$\begin{aligned} \widetilde{w}_a^{(\alpha)} &= (\widetilde{w}_b - \widetilde{w}_a)\alpha + \widetilde{w}_a; \\ \widetilde{w}_c^{(\alpha)} &= \widetilde{w}_c - (\widetilde{w}_c - \widetilde{w}_b)\alpha. \end{aligned} \tag{31}$$

Therefore,

$$\begin{aligned} \widetilde{w}_\alpha &= \left[\widetilde{w}_a^{(\alpha)}, \widetilde{w}_c^{(\alpha)}\right] \\ &= \left[(\widetilde{w}_b - \widetilde{w}_a)\alpha + \widetilde{w}_a, \widetilde{w}_c^{(\alpha)} = \widetilde{w}_c - (\widetilde{w}_c - \widetilde{w}_b)\alpha\right], \end{aligned} \tag{32}$$

where $\widetilde{w}_1 = [0.3, 0.5, 0.7]$, $\widetilde{w}_2 = [0.2, 0.3, 0.6]$, and $\alpha = 0.5$.

Table 5: The Taguchi L_9 design along with objective values of the algorithms.

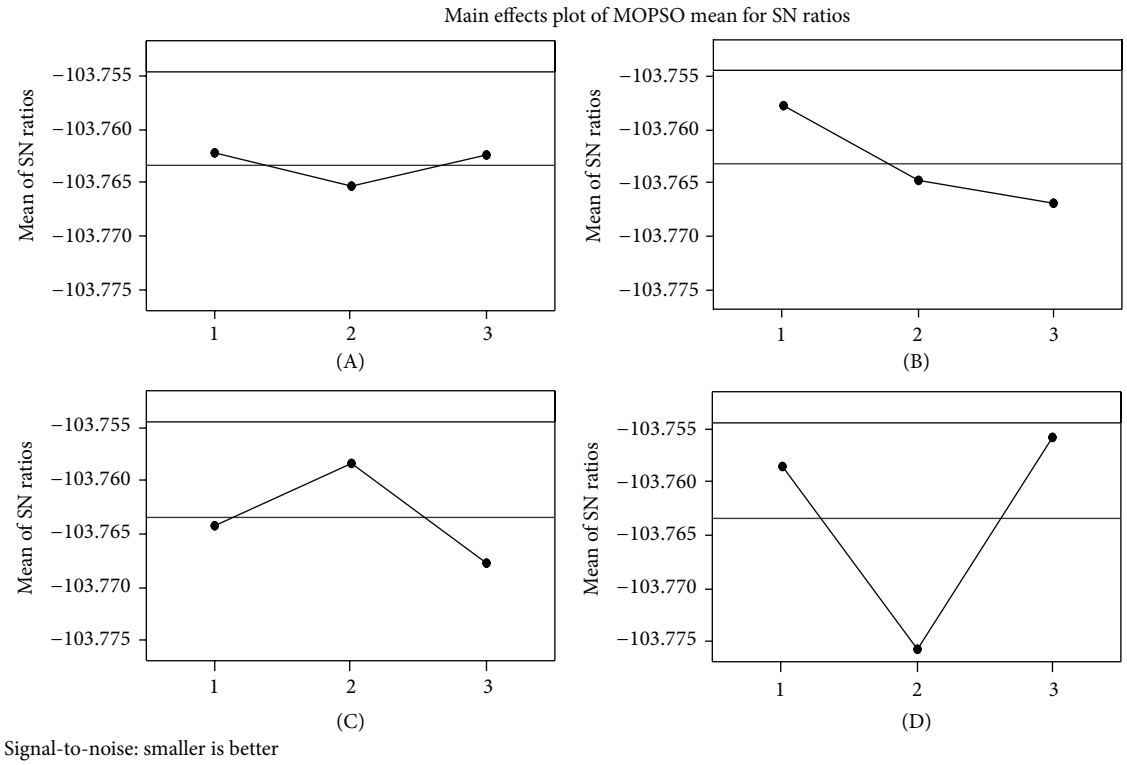
Run number	A	B	C	D	MOPSO	MOGA
1	1	1	1	1	154040	154980
2	1	2	2	2	154367	154760
3	1	3	3	3	154220	155075
4	2	1	2	3	153944	154875
5	2	2	3	1	153985	155230
6	2	3	1	2	154568	155102
7	3	1	3	2	154215	154780
8	3	2	1	3	154320	154750
9	3	3	2	1	154100	155111

Table 6: The optimal levels of the algorithms' parameters for problem 7 of Table 2.

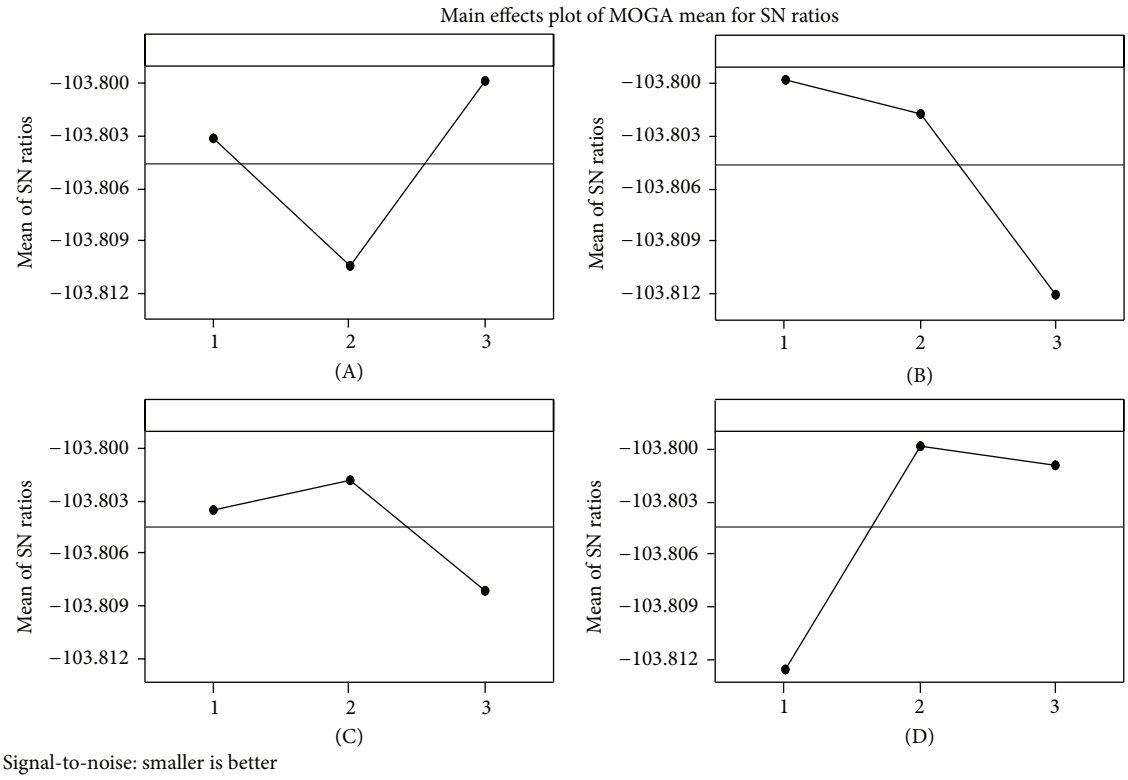
Algorithms	Factors	Optimal levels
MOPSO	C_1	2
	C_2	2.5
	Pop	40
	Gen	200
MOGA	P_C	0.6
	P_m	0.2
	Pop	50
	Gen	200

To perform Taguchi approach in this paper, a L_9 design is utilized, based on which the results for problem 7 described in Table 2 are shown in Table 5 as an example. The optimal values of the levels of the algorithms' parameters shown in Table 5 are represented by Table 6. Figure 5 depicts the mean S/N ratio plot each level of the factors of MOPSO and MOGA for problem 7 in Table 2.

Tables 7 and 8 show the best result obtained by MOPSO and MOGA for the problem with 5 items and 3 periods (problem 7), respectively, including the amounts of decision variables and the optimal objective values. In these tables,



(a)



(b)

Figure 5: The mean S/N ratio plot for parameter levels of MOPSO and MOGA in problem 7 of Table 2.

Table 7: The best result of the MOPSO algorithm.

Product	$Q_{i,1}$	$Q_{i,2}$	$X_{i,2}$	$X_{i,3}$	$V_{i,1}$	$V_{i,2}$	$b_{i,1}$	$b_{i,2}$	TMF
1	1215	159	15	0	405	53	0	626	153840
2	1162	252	0	0	166	36	138	786	
3	1555	190	55	0	311	38	0	955	
4	1360	864	0	0	170	108	740	1876	
5	1435	420	0	0	205	60	365	1545	

Table 8: The best result of the MOGA algorithm.

Product	$Q_{i,1}$	$Q_{i,2}$	$X_{i,2}$	$X_{i,3}$	$V_{i,1}$	$V_{i,2}$	$b_{i,1}$	$b_{i,2}$	TMF
1	1221	168	21	0	407	56	0	611	154550
2	959	392	0	0	137	56	341	849	
3	1220	390	0	0	244	78	280	1090	
4	1168	960	0	0	146	120	932	1972	
5	168	2254	0	0	24	322	1632	978	

Table 9: The ANOVA analysis of the performances.

Source	DF	SS	MS	F	P value
Factor	1	$1.11E+11$	$1.11E+11$	0.28	0.6
Error	78	$3.11E+13$	$3.99E+11$	—	—
Total	79	$3.12E+13$	—	—	—

TMF is the best value of the biobjective inventory planning problem, which is given in the last two columns of Table 2. Similarly, the best TMF for the other problems is obtained and is summarized in Table 2.

To compare the performances of the MOPSO and MOGA, several statistical and graphical approaches are employed. A one-way ANOVA analysis of the means of the algorithms in confidence 0.95% is used to compare and evaluate the objective values of the generated 40 problems. Table 9 shows the ANOVA analysis of the results of the two algorithms that demonstrates no significant difference between both algorithms. Moreover, the mean and standard deviation (Std. Dev) of the objective values of the 30 generated problems shows that the MOPSO has the better performance in terms of the objective values in comparison with the MOGA. In addition, a pictorial presentation of the performances of the two algorithms shown by Figure 6 displays that the MOPSO is more efficient than the MOGA algorithm in the large number of the problems.

Figure 7 depicts the boxplot and the individual value plot and Figure 8 shows the residual plots for the algorithms.

A comparison of the results in Table 2 shows that the MOPSO algorithm performs better than the MOGA in terms of the fitness functions values.

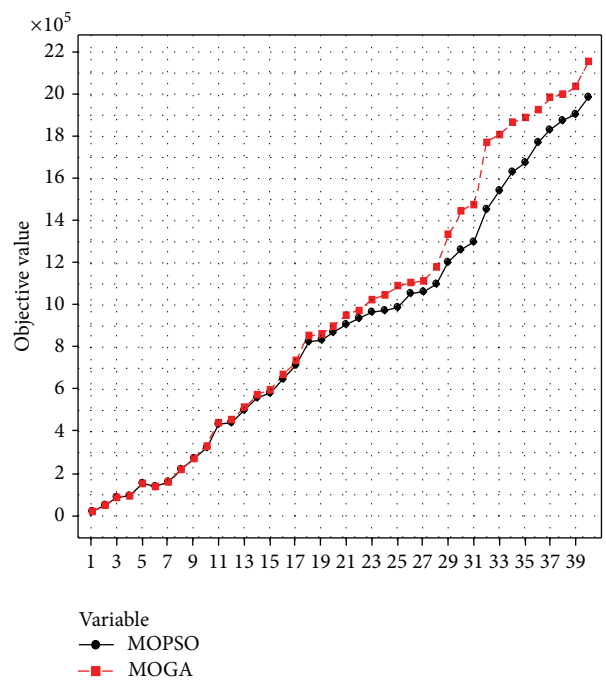


Figure 6: The pictorial representation of the performances of the algorithms.

6. Conclusion and Recommendations for Future Research

In this paper, a biobjective multi-item multiperiod inventory planning problem with total available budget under all unit discount for some items and incremental quantity discount for other items was considered. The orders were assumed to be placed in batch sizes and the order quantities at the end period were zeros. Shortages were allowed and contain backorder and lost sale. It was assumed that the beginning inventory level in primary period was zeros and the order quantity in each period was more than the shortage quantity in the previous period. Due to adopting decisions related to a certain department of production planning (extending warehouse or building a new manufacturing line), the manager decided to build a new warehouse for the ordered items. The objectives were to minimize both the total inventory costs

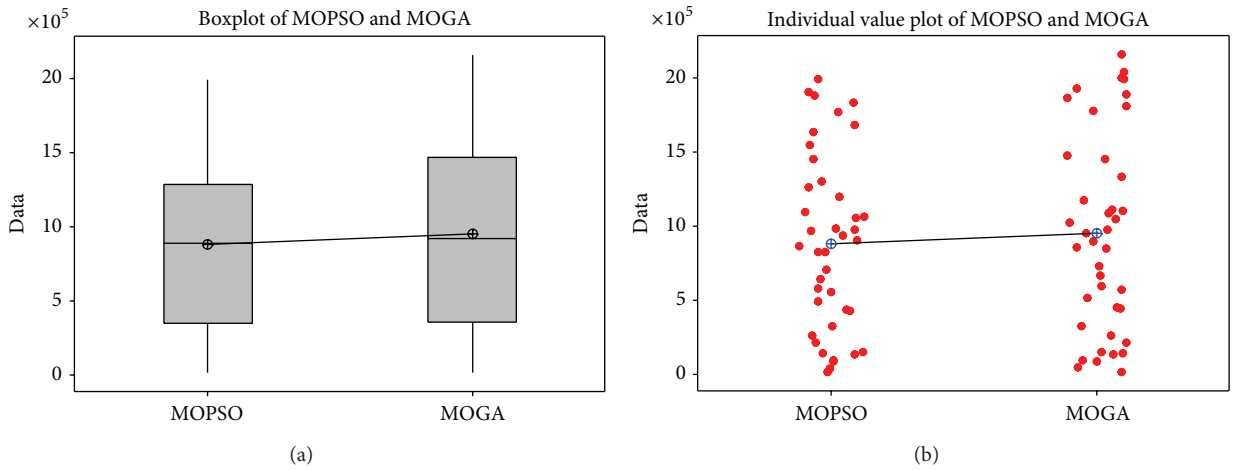


Figure 7: The boxplot and the individual value plot of the performances of the algorithms.

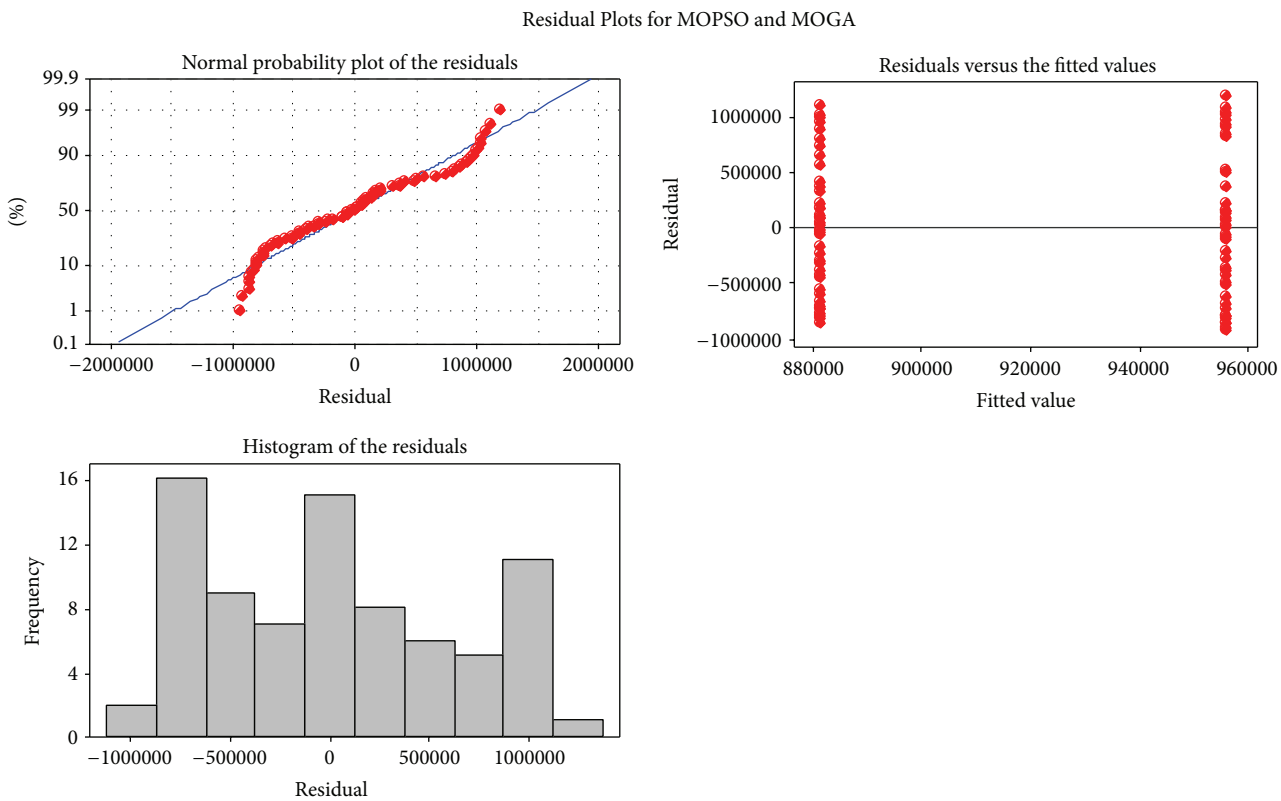


Figure 8: The residual plots of the algorithms.

and the total required storage space, for which a weighted combination was defined as the objective function. The aim of the study was to determine the optimal order quantity and the shortage quantity of each product in each period such that the objective function is minimized and the constraints hold. The developed model of the problem was shown to be an integer nonlinear programming mixed with binary variables. To solve the model, both a multiobjective particle swarm optimization and a multiobjective genetic algorithm were applied. The results showed that for the 10 specific problems the MOPSO performs better than the MOGA in terms of the fitness function values.

Some recommendations for future works are to expand the model to cover a supply chain environment, to consider fuzzy or stochastic demands, and/or to take into account the inflation and the time value of the money.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to acknowledge the University of Malaya for providing the necessary facilities and resources for this research. This research was fully funded by the Ministry of Higher Education, Malaysia, with the high impact research (HIR) Grant no. HIR-MOHE-16001-D000001.

References

[1] C. Chiang, "Optimal replenishment for a periodic review inventory system with two supply modes," *European Journal of Operational Research*, vol. 149, no. 1, pp. 229–244, 2003.

[2] E. Mohebbi and M. J. M. Posner, "Multiple replenishment orders in a continuous-review inventory system with lost sales," *Operations Research Letters*, vol. 30, no. 2, pp. 117–129, 2002.

[3] A. H. I. Lee and H.-Y. Kang, "A mixed 0-1 integer programming for inventory model: a case study of TFT-LCD manufacturing company in Taiwan," *Kybernetes*, vol. 37, no. 1, pp. 66–82, 2008.

[4] S. M. Mousavi, V. Hajipour, S. T. A. Niaki, and N. Alikar, "Optimizing multi-item multi-period inventory control system with discounted cash flow and inflation: two calibrated meta-heuristic algorithms," *Applied Mathematical Modelling*, vol. 37, no. 4, pp. 2241–2256, 2013.

[5] S. M. J. Mirzapour Al-e-hashem and Y. Rekik, "Multi-product multi-period Inventory Routing problem with a transshipment option: a green approach," *International Journal of Production Economics*, 2013.

[6] G. Janakiraman, S. J. Park, S. Seshadri, and Q. Wu, "New results on the newsvendor model and the multi-period inventory model with backordering," *Operations Research Letters*, vol. 41, no. 4, pp. 373–376, 2013.

[7] H. J. Shin and W. C. Benton, "Quantity discount-based inventory coordination: effectiveness and critical environmental factors," *Production and Operations Management*, vol. 13, no. 1, pp. 63–76, 2004.

[8] W. C. Benton, "Multiple price breaks and alternative purchase lot-sizing procedures in material requirements planning systems," *International Journal of Production Research*, vol. 23, no. 5, pp. 1025–1047, 1985.

[9] P. L. Abad, "Joint price and lot size determination when supplier offers incremental quantity discount," *Journal of the Operational Research Society*, vol. 39, no. 6, pp. 603–607, 1988.

[10] P. L. Abad, "Determining optimal selling price and lot size when the supplier offers all unit quantity discount," *Decision Science*, vol. 19, no. 6, pp. 632–634, 1988.

[11] A. K. Maiti and M. Maiti, "Discounted multi-item inventory model via genetic algorithm with roulette wheel selection, arithmetic crossover and uniform mutation in constraints bounded domains," *International Journal of Computer Mathematics*, vol. 85, no. 9, pp. 1341–1353, 2008.

[12] S. S. Sana and K. S. Chaudhuri, "A deterministic EOQ model with delays in payments and price-discount offers," *European Journal of Operational Research*, vol. 184, no. 2, pp. 509–533, 2008.

[13] A. A. Taleizadeh, S. T. A. Niaki, M.-B. Aryanezhad, and A. F. Tafti, "A genetic algorithm to optimize multiproduct multiconstraint inventory control systems with stochastic replenishment intervals and discount," *International Journal of Advanced Manufacturing Technology*, vol. 51, no. 1–4, pp. 311–323, 2010.

[14] S.-P. Chen and Y.-H. Ho, "Optimal inventory policy for the fuzzy newsboy problem with quantity discounts," *Information Sciences*, vol. 228, pp. 75–89, 2013.

[15] M. Khan, M. Y. Jaber, and A.-R. Ahmad, "An integrated supply chain model with errors in quality inspection and learning in production," *Omega*, vol. 42, no. 1, pp. 16–24, 2014.

[16] A. A. Taleizadeh, S. T. A. Niaki, M.-B. Aryanezhad, and N. Shafii, "A hybrid method of fuzzy simulation and genetic algorithm to optimize constrained inventory control systems with stochastic replenishments and fuzzy demand," *Information Sciences*, vol. 220, pp. 425–441, 2013.

[17] S.-H. Huang and P.-C. Lin, "A modified ant colony optimization algorithm for multi-item inventory routing problems with demand uncertainty," *Transportation Research E: Logistics and Transportation Review*, vol. 46, no. 5, pp. 598–611, 2010.

[18] K. Li, B. Chen, A. I. Sivakumar, and Y. Wu, "An inventory-routing problem with the objective of travel time minimization," *European Journal of Operational Research*, vol. 236, no. 3, pp. 936–945, 2014.

[19] L. Qin, L. Miao, Q. Ruan, and Y. Zhang, "A local search method for periodic inventory routing problem," *Expert Systems with Applications*, vol. 41, no. 2, pp. 765–778, 2014.

[20] A. A. Taleizadeh, S. T. A. Niaki, and V. Hosseini, "The multi-product multi-constraint newsboy problem with incremental discount and batch order," *Asian Journal of Applied Science*, vol. 1, no. 2, pp. 110–122, 2008.

[21] A. A. Taleizadeh, S. T. A. Niaki, and V. Hoseini, "Optimizing the multi-product, multi-constraint, bi-objective newsboy problem with discount by a hybrid method of goal programming and genetic algorithm," *Engineering Optimization*, vol. 41, no. 5, pp. 437–457, 2009.

[22] S. H. R. Pasandideh, S. T. A. Niaki, and M. Hemmati far, "Optimization of vendor managed inventory of multiproduct EPQ model with multiple constraints using genetic algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 71, no. 1–4, pp. 365–376, 2014.

[23] A. A. Taleizadeh, M.-B. Aryanezhad, and S. T. A. Niaki, "Optimizing multi-product multi-constraint inventory control systems with stochastic replenishments," *Journal of Applied Sciences*, vol. 8, no. 7, pp. 1228–1234, 2008.

[24] A. A. Taleizadeh, S. T. A. Niaki, and M.-B. Aryanezhad, "A hybrid method of Pareto, TOPSIS and genetic algorithm to optimize multi-product multi-constraint inventory control systems with random fuzzy replenishments," *Mathematical and Computer Modelling*, vol. 49, no. 5-6, pp. 1044–1057, 2009.

[25] A. A. Taleizadeh, S. T. A. Niaki, and M.-B. Aryaneznad, "Multi-product multi-constraint inventory control systems with stochastic replenishment and discount under fuzzy purchasing price and holding costs," *The American Journal of Applied Sciences*, vol. 6, no. 1, pp. 1–12, 2009.

[26] A. A. Taleizadeh, S. T. A. Niaki, and F. Barzinpour, "Multiple-buyer multiple-vendor multi-product multi-constraint supply chain problem with stochastic demand and variable lead-time: a harmony search algorithm," *Applied Mathematics and Computation*, vol. 217, no. 22, pp. 9234–9253, 2011.

[27] A. A. Taleizadeh, S. T. A. Niaki, and H.-M. Wee, "Joint single vendor-single buyer supply chain problem with stochastic demand and fuzzy lead-time," *Knowledge-Based Systems*, vol. 48, pp. 1–9, 2013.

[28] T. K. Roy and M. Maiti, "Multi-objective inventory models of deteriorating items with some constraints in a fuzzy environment," *Computers and Operations Research*, vol. 25, no. 12, pp. 1085–1095, 1998.

[29] R. G. Yaghin, S. M. T. Fatemi Ghomi, and S. A. Torabi, "A possibilistic multiple objective pricing and lot-sizing model with multiple demand classes," *Fuzzy Sets and Systems*, vol. 231, pp. 26–44, 2013.

[30] G. Dueck, "Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing," *Journal of Computational Physics*, vol. 90, no. 1, pp. 161–175, 1990.

[31] S. J. Joo and J. Y. Bong, "Construction of exact D-optimal designs by Tabu search," *Computational Statistics and Data Analysis*, vol. 21, no. 2, pp. 181–191, 1996.

[32] J. Sadeghi, S. Sadeghi, and S. T. A. Niaki, "Optimizing a hybrid vendor-managed inventory and transportation problem with fuzzy demand: an improved particle swarm optimization algorithm," *Information Sciences*, vol. 272, pp. 126–144, 2014.

[33] E. H. L. Aarts and J. H. M. Korst, *Simulated Annealing and Boltzmann Machine: A stochastic Approach to Computing*, John Wiley & Sons, Chichester, UK, 1st edition, 1989.

[34] A. Varyani and A. Jalilvand-Nejad, "Determining the optimum production quantity in three-echelon production system with stochastic demand," *International Journal of Advanced Manufacturing Technology*, vol. 72, no. 1–4, pp. 119–133, 2014.

[35] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[36] A. R. Gaiduk, Y. A. Vershinin, and M. J. West, "Neural networks and optimization problems," in *Proceedings of the IEEE International Conference on Control Applications*, vol. 1, pp. 37–41, September 2002.

[37] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, Cambridge, Mass, USA, 2004.

[38] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[39] K. S. Lee and Z. W. Geem, "A new structural optimization method based on the harmony search algorithm," *Computers and Structures*, vol. 82, no. 9-10, pp. 781–798, 2004.

[40] A. A. Taleizadeh, H. Moghadasi, S. T. A. Niaki, and A. Eftekhari, "An economic order quantity under joint replenishment policy to supply expensive imported raw materials with payment in advance," *Journal of Applied Sciences*, vol. 8, no. 23, pp. 4263–4273, 2008.

[41] F. Fu, "Integrated scheduling and batch ordering for construction project," *Applied Mathematical Modelling*, vol. 38, no. 2, pp. 784–797, 2014.

[42] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Wash, USA, 1995.

[43] A. A. Taleizadeh, S. T. A. Niaki, N. Shafii, R. G. Meibodi, and A. Jabbarzadeh, "A particle swarm optimization approach for constraint joint single buyer-single vendor inventory problem with changeable lead time and (r,Q) policy in supply chain," *International Journal of Advanced Manufacturing Technology*, vol. 51, no. 9–12, pp. 1209–1223, 2010.

[44] Y.-R. Chen and C.-Y. Dye, "Application of particle swarm optimisation for solving deteriorating inventory model with fluctuating demand and controllable deterioration rate," *International Journal of Systems Science*, vol. 44, no. 6, pp. 1026–1039, 2013.

[45] P. J. Agrell, "A multicriteria framework for inventory control," *International Journal of Production Economics*, vol. 41, no. 1–3, pp. 59–70, 1995.

[46] C.-S. Tsou, "Multi-objective inventory planning using MOPSO and TOPSIS," *Expert Systems with Applications*, vol. 35, no. 1-2, pp. 136–142, 2008.

[47] C. A. Coello Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization," in *Proceedings of the IEEE Congress on Computational Intelligence*, vol. 2, pp. 1051–1056, Honolulu, Hawaii, USA, May 2002.

[48] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[49] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[50] C. A. Coello Coello and G. T. Pulido, "Multiobjective optimization using a micro-genetic algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '01)*, pp. 274–282, San Francisco, Calif, USA, 2001.

[51] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[52] L. El-Sharkawi, *Modern Heuristic Optimization Techniques*, Wiley InterScience, New Jersey, NJ, USA, 1st edition, 2008.

[53] H. Shayeghi, H. A. Shayanfar, S. Jalilzadeh, and A. Safari, "A PSO based unified power flow controller for damping of power system oscillations," *Energy Conversion and Management*, vol. 50, no. 10, pp. 2583–2592, 2009.

[54] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[55] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, "Practical distribution state estimation using hybrid particle swarm optimization," in *Proceedings of the IEEE Power Engineering Society Winter Meeting*, vol. 2, pp. 815–820, Columbus, Ohio, USA, February 2001.

[56] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation*, Washington, DC, USA, July 1999.