Optimizing a hybrid vendor-managed inventory and transportation problem with fuzzy demand: An improved particle swarm optimization algorithm

Javad Sadeghi, Saeid Sadeghi, Seyed Taghi Akhavan Niaki

a Young Researchers and Elite Club, Qazvin Branch, Islamic Azad University, Qazvin, Iran
b Young Researchers and Elite Club, South Tehran Branch, Islamic Azad University, Tehran, Iran
c Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran

Abstract

Vendor-managed inventory (VMI) is a popular policy in supply chain management (SCM) to decrease bullwhip effect. Since the transportation cost plays an important role in VMI and because the demands are often fuzzy, this paper develops a VMI model in a multi-retailer single-vendor SCM under the consignment stock policy. The aim is to find optimal retailers' order quantities so that the total inventory and transportation cost are minimized while several constraints are satisfied. Because of the NP-hardness of the problem, an algorithm based on particle swarm optimization (PSO) is proposed to find a near optimum solution, where the centroid defuzzification method is employed for defuzzification. Since there is no benchmark available in the literature, another meta-heuristic, namely genetic algorithm (GA), is presented in order to verify the solution obtained by PSO. Besides, to make PSO faster in finding a solution, it is improved by a local search. The parameters of both algorithms are calibrated using the Taguchi method to have better quality solutions. At the end, conclusions are made and future research is recommended.

Keywords: Vendor managed inventory; Consignment stock; Fuzzy demand; Particle swarm optimization (PSO); Genetic algorithm (GA); Taguchi method.

1 Corresponding author
E-mail addresses: Keeyarash@gmail.com, Sadeqi@qiau.ac.ir (J. Sadeghi); Saeed.sadeghi.85@gmail.com (S. Sadeghi); Niaki@Sharif.edu (S.T.A. Niaki)
1. Introduction

Companies with integrated supply chains are shown to be more competitive than the others in today's market environments in the sense of reducing total cost. Supply chain management (SCM) involves integrated decisions on transportation, location, inventory, and production to give the best mix of efficiency and responsiveness to the market being served [21]. Inventory management plays an important role in SCM, where academic and industrial communities made many strategies in order to reduce total inventory cost. These strategies are used in making inventory-related decisions that coordinate suppliers and retailers. The vendor managed inventory (VMI), which is the most popular strategy in the inventory management area, was successfully implemented by retailers such as JC-Penney and Wal-Mart [11, 45]. In the VMI approach, the vendor (or supplier) makes a decision for a chain consisting of some retailers and a vendor. The retailers share the inventory and sales information in order to cooperate with the vendor who determines the replenishment frequency and the order quantity. In general, the VMI approach can reduce the demand variability in order to reduce the total inventory cost.

The consignment stock (CS), which defines the stock ownership in a supply chain, is a relatively new approach in VMI modeling, in which retailers pay for the goods that are sold only. In other words, the unsold stock belongs to the vendor who is the legal owner of the goods although a retailer holds the stock.

Customer demand is difficult to predict and is uncertain in most situations. While some researchers modeled this uncertainty using stochastic approaches, many others employed non-deterministic methods such as fuzzy set theory. The main reason to use fuzzy demand is unexpected changes in customer demand [45].

In an attempt to reduce transportation cost by determining the shortest route to deliver goods, this paper extends the VMI model of Zavanella and Zanoni [55] for a supply chain.
The transportation cost corresponds to deliveries of the goods to the retailers located in different places, where optimizing the delivery routes of the vendor's vehicle is similar to the one in the well-known traveling salesman problem (TSP). Besides, a trapezoidal fuzzy number is used to model fuzzy demand. Moreover, there are three constraints; 1) the retailers' warehouse is limited; 2) the vendor has a limit on his average inventory; 3) there is an upper bound on the total number of replenishments. The aim is to find the order size, the replenishment frequency of the retailers, and the shortest route, such that the total inventory and transportation cost are minimized. We will show that the developed VMI model of the problem at hand belongs to the class of NP-hard problems. Accordingly, a hybrid particle swarm optimization (PSO) algorithm with a local searcher is utilized to find a near optimum solution, where the centroid defuzzification method is employed for defuzzification. Since there is no benchmark available in the literature, another meta-heuristic, namely genetic algorithm (GA) is presented in order to verify the solution obtained by PSO. Moreover, to find better quality solutions, the parameters of both algorithms are calibrated using the Taguchi method.

The structure of the remainder of the paper is as follows. The motivation and contribution of the paper is provided in the next section. Section 3 illustrates a brief review on the VMI problem. Section 4 presents the proposed VMI model along with the notations used and the assumptions made. Meta-heuristics and parameter tuning come in Sections 5 and 6, respectively. Section 7 presents the analysis of the solutions. Finally, conclusions and recommendations for future research are given in Section 8.

2. Motivation and contribution

VMI is one of the most popular strategies in retailer-supplier partnerships to decrease bullwhip effect as well as to reduce total inventory cost. Several successful retailers such as
Wal-Mart and JC-Penny took advantages of the VMI policy [44]. The bullwhip effect, which is an observed phenomenon in forecast-driven distribution channels of SCM, concludes to greater safety stocks with increased total cost. Thus, it is important to employ supply chain policies that reduce demand variability. To reduce demand variability and hence to cope with the bullwhip effect in a supply chain, in this research the demand is considered fuzzy. Moreover, transportation cost has a key role in total cost of the chain. For example, the freight transportation costs of General Motors with a large production and distribution networks were about $4.1 billion in 1984 [44].

Reducing demand variability using fuzzy numbers and considering transportation cost of a supply chain that operates under CS and VMI policies are the two motivations of this research. This research is probably one of the first under fuzzy demands that considers a closer to reality VMI problem in which the consignment stock approach is employed, the transportation cost is taken into account, and that there are several constraints. Another novelty of this paper comes from proposing an algorithm based on PSO named hybrid PSO that not only utilizes the centroid defuzzification method to defuzzify membership functions, but also a local searcher to find better solution. This algorithm tries to find a near optimum solution of an integer nonlinear programming that belongs to the class of NP-hard problems. Furthermore, in order to improve the quality of the solution obtained, the parameters of the meta-heuristic are calibrated using the Taguchi method.

3. Literature survey

Since a VMI model based on the economic production quantity (EPQ) policy is developed in this paper, among many research works proposed in the literature of VMI, those that use either EPQ or the economic order quantity (EOQ) are first reviewed in the following two subsections. Then, relevant literature on fuzzy VMI modeling is surveyed in Subsection
3.3. The meta-heuristic algorithms proposed in VMI environments are next reviewed in Subsection 3.4.

3.1. VMI literature based on the EOQ policy

Regarding the EOQ policy, Yao et al. [52] proposed a VMI model in a supply chain involving a single vendor and a single retailer. Afterwards, Darwish and Odah [10] first developed a VMI model for a multi-retailer single-vendor supply chain problem with a similar interval of consumption, and then applied a heuristic algorithm to solve the problem. Next, Pasandideh et al. [36] studied a VMI model with several constraints for a single-supplier single-retailer in which shortages were backordered. They proposed a genetic algorithm (GA) with tuned parameters to find a near-optimum solution of the problem. However, they did not concentrate on the maximum available inventory level in their modeling. As a result, Cárdenas-Barrón et al. [4] first extended their model to contain the maximum available inventory, and then used a better heuristic algorithm to solve the problem. Nonetheless, both ignored a closer to reality assumption on the replenishment frequencies. Since the vendor replenishes the retailers based on certain frequencies, Sadeghi et al. [39] improved the model taking into account the replenishment frequencies. To investigate unequal replenishment intervals in the VMI model, Hariga et al. [17] extended the model presented by Darwish and Odah [10] and utilized a simple heuristic to solve the problem.

3.2. VMI literature based on the EPQ policy

In the EPQ policy, Goyal [15] first wrote a note on the Banerjee's model for a single-vendor single-purchaser supply chain, and then developed a model for the multi-buyer case [16]. However, he did not consider the effect of production rate on the average inventory.
Moreover, Goyal [15, 16] and Lu [26] proposed several models based on an equal shipment size. Later, Hill [19] considered condition in which shipment sizes might vary. In addition, Hill and Omar [20] first made investigation on several existing models and then presented a model in which shipments might differ. The model proposed by Hill [19] was then extended by Braglia and Zavanella [3] based on the CS policy for a single-buyer case. This model was further extended by Zavanella and Zanoni [55] for a multi-buyer case, in which a sensitivity analysis on the parameters was made.

Chung [6] extended the VMI model previously presented by Yang et al. [51] for a single-vendor single-buyer SC such that the buyer’s shortage was allowed. He optimized the replenishment frequency with the cost-difference rate comparison method.

Regarding CS agreement, Hariga and Al-Ahmari [18] studied a VMI problem including a supplier and a retailer facing a stock-dependent demand for a single item. In their proposed model, by sharing data with the retailer, the supplier decides on the maximum shelf space to be reserved for the item, the sizing of each order, and the reorder point. They also made an extensive sensitivity analysis to elaborate the impact of key parameters on both parties’ net profits and on the problem decision variables.

3.3. The fuzzy VMI models

In order to deal with the uncertainties involved predicting customer demands in supply chains, many researchers proposed various approaches. To name a few, Kazemi et al. [22] presented an EOQ model, in which the demand and the holding cost were modeled using triangular and trapezoidal fuzzy numbers, respectively. They analytically solved the problem using the Kuhn–Tucker conditions. Mondal and Maiti [29] proposed fuzzy inventory models involving several items and illustrated the performance of GA in solving the problem. They showed GA had a better performance than a fuzzy non-linear programming (FNLP) method.
based on Zimmermann’s approach. In VMI models, in order to reduce the bullwhip effect, Lin et al. [25] developed a simulation method under fuzzy condition for an inventory model in a supply chain. They proposed a GA to search optimal fuzzy parameters. Later, Kristianto et al. [24] presented an adaptive fuzzy VMI control to improve the performance of the VMI approach in reducing the bullwhip effect. Regarding production problems, Aliev et al. [1] first developed an integrated production–distribution planning model in which customer demand and capacities were assumed trapezoidal fuzzy number, and then obtained a general near-optimal plan by a GA. Moreover, Chen and Ho [5] presented an inventory problem with fuzzy demand for the single-order newsboy problem confronted with quantity discounts, and then provided optimal inventory policies. To overcome supply chain outsourcing risk, Wu et al. [50] considered both random and fuzzy uncertainty to model a stochastic fuzzy multi-objective programming problem and proposed a heuristic algorithm to solve the extended problem.

3.4. Meta-heuristics for the VMI models

Evolutionary approaches were applied repetitively in the last decades to find near-optimum solutions of some integer nonlinear programming (INLP) problems. In the VMI environment, for example, some researchers such as Nachiappan and Jawahar [32] and Pasandideh et al. [36] first developed INLP models of VMI problems in supply chains and then employed GA to solve them.

Considering inventory problems, Taleizadeh et al. [49] provided a hybrid method of fuzzy simulation (FS) and GA to optimize a multiproduct inventory model by assuming independent random variables for the times between two replenishments. Besides, Boussaïd et al. [2] carried out a review on main meta-heuristics such as single-solution-based and population-based algorithms, and then presented their similarities and differences.
Table 1 presents a summary of some relevant research-works provided in the VMI problems. According to this table, it can be concluded that both the transportation cost and fuzzy demand has not been considered in a VMI environment yet.

In the next section, the mathematical formulation of the VMI problem is presented.

<table>
<thead>
<tr>
<th>Study</th>
<th>R</th>
<th>D</th>
<th>CS</th>
<th>TC</th>
<th>SM</th>
<th>TP</th>
<th>Policy</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goyal [15]</td>
<td>Single</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Heuristic</td>
<td>No</td>
<td>EPQ</td>
<td>No</td>
</tr>
<tr>
<td>Lu [26]</td>
<td>Multi</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Heuristic</td>
<td>No</td>
<td>EPQ</td>
<td>No</td>
</tr>
<tr>
<td>Hill [19]</td>
<td>Single</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Heuristic</td>
<td>No</td>
<td>EPQ</td>
<td>No</td>
</tr>
<tr>
<td>Braglia and Zavanella [3]</td>
<td>Single</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Optimal Solution</td>
<td>No</td>
<td>EPQ</td>
<td>No</td>
</tr>
<tr>
<td>Nachiappan and Jawahar [32]</td>
<td>Multi</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>GA</td>
<td>No</td>
<td>EOQ</td>
<td>Storage, Sales quantity</td>
</tr>
<tr>
<td>Yao, Evers and Dresner [52]</td>
<td>Single</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Optimal Solution</td>
<td>No</td>
<td>EOQ</td>
<td>No</td>
</tr>
<tr>
<td>Zavanella and Zanoni [55]</td>
<td>Multi</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Optimal Solution</td>
<td>No</td>
<td>EPQ</td>
<td>No</td>
</tr>
<tr>
<td>Lin et al. [22]</td>
<td>Single</td>
<td>Fuzzy</td>
<td>No</td>
<td>No</td>
<td>GA</td>
<td>No</td>
<td>Production</td>
<td>No</td>
</tr>
<tr>
<td>Darwish and Odah [10]</td>
<td>Multi</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Heuristic</td>
<td>No</td>
<td>EOQ</td>
<td>Overstock Penalty</td>
</tr>
<tr>
<td>Pasandideh et al. [36]</td>
<td>Single</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>GA</td>
<td>No</td>
<td>EOQ</td>
<td>Storage, Number of orders</td>
</tr>
<tr>
<td>Sadeghi, Sadeghi and Saidi-Mehrabad [39]</td>
<td>Single</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>GA</td>
<td>RSM</td>
<td>EOQ</td>
<td>Storage, Number of orders, Budget, available inventory</td>
</tr>
<tr>
<td>Sue-Ann et al. [46]</td>
<td>Multi</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>GA, PSO, GA-AIS</td>
<td>No</td>
<td>EOQ</td>
<td>Storage, Sales quantity</td>
</tr>
<tr>
<td>Hariga and Al-Ahmari [18]</td>
<td>Single</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Optimal Solution</td>
<td>No</td>
<td>EOQ</td>
<td>Shelf space</td>
</tr>
<tr>
<td>Proposed Model</td>
<td>Multi</td>
<td>Fuzzy</td>
<td>Yes</td>
<td>Yes</td>
<td>GA, Hybrid PSO</td>
<td>Taguchi</td>
<td>EPQ</td>
<td>Storage, Number of replenishments, Average inventory</td>
</tr>
</tbody>
</table>

R=Retailer; D=Demand; CS=Consignment Stock; TC=Transportation Cost; SM=Solution Method; TP=Tuning Parameters

4. Model derivation

The notations that are used in this paper are as follows.
$r$  Number of retailers

$i$  An index used for a retailer; $i = 1, 2, ..., r$

$A_i$  Ordering cost of retailer $i$

$A$  Ordering cost of the vendor

$h_i$  The unit holding cost of $i^{th}$ retailer's product per year

$H$  The unit holding cost of the vendor's products per year

$P$  Vendor production rate per year

$I_v$  Vendor average inventory

$I_r$  Retailer average inventory

$n_i$  Number of replenishment of retailer $i$ by vendor (a decision variable)

$\tilde{d}_i$  The total annual triangular fuzzy demand of $i^{th}$ retailer

$\tilde{D}$  The total annual triangular fuzzy demand of the vendor

$q_i$  Order quantity for retailer $i$ (a decision variable)

$Q_v$  Total vendor's order quantity

$Z$  Upper bound on the vendor's average inventory level

$f$  Space required storing one unit of the product

$F$  Total available storage space for retailers

$N$  Upper bound on the number of replenishments

$TC_v$  The vendor's total inventory cost

$tc$  The vendor's transportation cost

$TC_r$  The retailers' total inventory cost

$TC_{VMI}$  Total inventory cost in the VMI system

$TC$  Total cost in the VMI system
$S_{ik}$  Transportation cost from retailer $i$ to retailer $k$

$x_{ik}$  Is 1 if retailer $i$ is reached from retailer $k$, 0 otherwise (a decision variable)

$T_i$  Ordering cycle times of $i^{th}$ retailer

$T_v$  Production cycle times of the vendor

In EPQ of the VMI system, the vendor produces items in batches. In other words, the stock is transferred discretely in the chain and the vendor replenishes $i^{th}$ retailer $q_i$ items per delivery. This assumption affects the vendor average inventory and hence affects the holding cost. Considering the CS strategy, the retailer's warehouse is used in order to hold items, whereas, the vendor's warehouse is utilized to store materials in a No-CS condition (or traditional condition of VMI). Figs. 1 and 2 illustrate these two policies.

![Stock levels in Hill’s model](image)

**Fig. 1:** Stock levels in Hill’s model [3]
In Fig. 1, the behavior of the vendor's inventory is displayed in the traditional VMI policy so that the vendor keeps all ordered products to send to retailers periodically. Thus, the vendor has a gradual increase and a steep decrease in productive and non-productive conditions, respectively. Similarly, Fig. 2 shows the behavior of retailer's stock in CS-VMI strategy. In this policy, the retailer keeps all ordered products. There are two behaviors at the retailer's inventory level according to Fig. 2. First, a gradual consumption with immediate increases in inventory occurs when the vendor is replenishing the retailer. Second, a gradual consumption takes place for which the vendor does not replenish the retailer. Thus, the CS policy changes the location of stocks from vendor to retailer so that retailer pays for goods only when they are consumed. Considering an annual demand ($D$) for a single product, some
researchers such as [19] and [55] assumed that the unit vendor holding cost ($H$) to be less than that of the retailer's ($h$). As a result, it is logical to hold stocks in the vendor's warehouse due to low-cost bulk storage facilities. Hence, on the one hand the average vendor stock is obtained as

$$I_v = \text{average system stock} - \text{average retailer stock}$$

(1)

On the other hand,

$$I_v = \left( Q_v \left( 1 - \frac{D}{P} \right) \right) / 2 + qD/P - q/2$$

(2)

where $qD/P$ is the stock ($q$) that is calculated based on $D/P$ intervals and $Q_v \left( 1 - \frac{D}{P} \right)$ is the maximum system stock without reduction of stock made in batches.

As mentioned, $H > h$ in the CS policy [3]. Hence, the average retailer stock is calculated as

$$I_r = \left( Q_r \left( 1 - \frac{D}{P} \right) \right) / 2 + qD/P$$

(3)

With the above mentioned descriptions, we employ CS approaches to formulate the problem based on Zavanella and Zanoni [55].

Assuming the retailers share a unique consumption interval, we have,

$$T_i = \frac{n_i q_i}{d_i} = \frac{n_i q_i}{d_i} ; \quad i = 1, 2, ..., r$$

(4)

Here, $\sim$ denotes the trapezoidal fuzzy number for demands such that its membership function, which is defined by four parameters $\{a, b, c, d\}$, is as follows.

$$\mu_f(x) = \begin{cases} 0, & x < a \\ \frac{(x-a)}{(b-a)}, & a \leq x \leq b \\ 1, & b \leq x < c \\ \frac{(d-x)}{(d-c)}, & c \leq x < d \\ 0, & x > d \end{cases}$$

(5)

In (5) $x$ is a fuzzy member and $\mu_f(x)$, which takes values between 0 and 1, is the
membership degree of \( x \) in a fuzzy set. The value 0 for \( \mu_T(x) \) means that \( x \) is not a member of the fuzzy set; while the value 1 means that \( x \) is fully a member of the fuzzy set. Fig. 3 depicts a trapezoidal fuzzy number.

Thus, the order quantity for the \( i^{th} \) retailer is,

\[
q_i = \tilde{a}_i n_i q_i / (\tilde{a}_i n_i) ; \quad i = 1, 2, ..., r
\]

(6)

Moreover, since the production cycle time of the vendor is equal to the ordering cycle time of the retailers, we have,

\[
T_i = T_r ; \quad i = 1, 2, ..., r
\]

(7)

Besides, the total vendor's order quantity is,

\[
Q_v = \sum_{i=1}^{r} n_i q_i
\]

(8)

As a result, the total inventory cost of the vendor is:

\[
TC_v = \sum_{i=1}^{r} A \tilde{d}_i / (n_i q_i) + \sum_{i=1}^{r} H q_i \tilde{d}_i / (2P)
\]

(9)

where the first term on the right hand side is the total ordering cost and the second is the total holding cost of the vendor.

Similarly, the total inventory cost of the retailers can be written as

\[
TC_r = \sum_{i=1}^{r} A \tilde{d}_i / q_i + \sum_{i=1}^{r} h_i \left( n_i q_i \left(1 - \frac{\tilde{d}_i}{P}\right) + \frac{q_i \tilde{d}_i}{P}\right) / 2
\]

(10)
where the total ordering cost of the retailers is the first term on the right hand side and the second term (divided by 2) is the total holding cost of the retailers. Furthermore, according to Zavanella and Zanoni's model [55], the average inventory cost of the VMI system is,

\[ TC_{VMI} = TC_V + TC_R \]  \hspace{1cm} (11)

Assuming that the vendor delivers the goods to all the retailers using a single vehicle, the transportation cost of the vendor becomes

\[ tc = \sum_{i=1}^{r'} \sum_{k=1}^{r'} S_{ik} x_{ik} \]  \hspace{1cm} (12)

where \( S_{ik} \) is the transportation cost from \( i^{th} \) to \( k^{th} \) retailer and \( x_{ik} \) is 1 if retailer \( i \) is reached from retailer \( k \), otherwise 0. Consequently, the total cost in the VMI system is obtained as

\[ TC = TC_{VMI} + tc \]  \hspace{1cm} (13)

There are three constraints; first, the total available storage space for retailers is restricted to \( F \); second, the average inventory is limited to \( Z \); third, the maximum number of replenishments is \( N \). Inequalities (14), (15), and (16) can easily formulate these constraints, respectively.

\[ \sum_{i=1}^{r'} f \left( n_i q_i \left( 1 - \frac{\tilde{d}_i}{P} \right) + \frac{\tilde{d}_i q_i}{P} \right) \leq F \]  \hspace{1cm} (14)

\[ \sum_{i=1}^{r'} q_i \tilde{d}_i / (2P) \leq Z \]  \hspace{1cm} (15)

\[ \sum_{i=1}^{r'} n_i \leq N \]  \hspace{1cm} (16)

In short, with the objective of minimizing the total cost of the VMI system and based on the above derivations, the proposed model of the problem becomes

\[
\text{Min } TC = \frac{A \tilde{d}_i}{n_i q_i} \sum_{i=1}^{r'} \frac{\tilde{d}_i}{d_i} + \frac{H n_i q_i}{2d_i P} \sum_{i=1}^{r'} d_i n_i + \frac{\tilde{d}_i q_i}{n_i q_i} \sum_{i=1}^{r'} A_i \tilde{d}_i n_i + \\
\frac{n_i q_i}{2d_i} \sum_{i=1}^{r'} h_i \tilde{d}_i \left( 1 - \frac{\tilde{d}_i}{P} + \frac{\tilde{d}_i}{n_i P} \right) + \sum_{j=1}^{r'} \sum_{k=1}^{r'} S_{ik} x_{ik};
\]  \hspace{1cm} (17)
\begin{align*}
\sum_{i=1}^{r} \tilde{d}_i n_i q_i \left( 1 - \frac{\tilde{d}_i}{P} + \frac{\tilde{d}_i}{n_i P} \right) \tilde{d}_i & \leq F \quad \text{(18)} \\
n_i q_i \left( \sum_{i=1}^{r} \frac{\tilde{d}_i^2}{n_i} \right) / (2 \tilde{d}_i P) & \leq Z \quad \text{(19)} \\
\sum_{i=1}^{r} n_i & \leq N \quad \text{(20)} \\
\sum_{i=1}^{r} x_{ik} & = 1 \quad ; \quad k = 1, \ldots, r \quad \text{(21)} \\
\sum_{k=1}^{r} x_{ik} & = 1 \quad ; \quad i = 1, \ldots, r \quad \text{(22)} \\
S_{ik} & = \infty \text{ for all } i = k \quad ; \quad i = 1, \ldots, r \quad ; \quad k = 1, \ldots, r \quad \text{(23)}
\end{align*}

where Eqs. (21) and (22) do not permit the vendor's vehicle to pass from a retailer's location twice and the decision variables are the order quantities of the retailers, $q_i$, the number of replenishments for the retailers, $n_i$, and the route taken by the vendor's vehicle to reach retailers is modeled by the binary $x_{ik}$ variable.

5. The solution method

Computational complexity theory, which is a major branch in the theory of computation, focuses on sorting problems with respect to their inherent difficulty. According to the difficulty levels of solving problems, there are four classes: P, NP, NP-complete, and NP-hard. Besides, there are not provably efficient algorithms for NP-hard problems, which are most of the real-world optimization problems. Studies illustrate that meta-heuristics can be used to solve this class of problems [48].

The mathematical formulation derived in Section 4 is of a non-linear integer programming (INLP) type with a part that models a travelling salesman problem (TSP) to obtain the optimal vehicle route. Besides, it has been shown in [27] that Hamiltonian cycle
problem is NP-complete, which infers NP-hardness of TSP. Since TSP is NP-hard itself, therefore, the formulation corresponds to an NP-hard problem and exact methods such as branch and bound cannot obtain the optimal solution in normal time for even a medium size problem. Hence, this paper uses two meta-heuristics namely PSO and GA to find a near optimum solution of the problem modeled in Section 4. These algorithms are first presented in this section to show how they can be used to solve the problem. Then in Section 6, the parameters of both algorithms are calibrated using the Taguchi method. Next, the algorithms are employed in Section 7 to find near optimum solutions. At the end, the PSO is improved by a local searcher (resulting in a hybrid PSO) in Subsection 7.1. In other words, a PSO is first developed to find a near-optimum solution. Then, a GA is utilized to validate the results obtained. Since PSO shows to be slower than GA, it is improved by a local searcher.

In order to compare the solution proposal with some other relevant methods proposed in the literature qualitatively, we could mention the followings.

a) Considering the studies in [10, 16, 20, 22, 26], since an INLP problem is modeled in this paper, the Lagrange multiplier methods, which calculates a system of inequalities called the Karush–Kuhn–Tucker conditions, cannot be utilized to solve the problem. Instead, it seems reasonable to use the solution proposal, a powerful mathematical optimization tool to solve the presented model. In other words, the solution proposal does not need to satisfy any optimality condition.

b) Since meta-heuristic algorithms randomly produce initial solutions, their results require validation. Despite Sadeghi et al. [39] who did not verify the results obtained, the near-optimum solutions of the solution proposal are validated using another meta-heuristic.

c) The values of parameters in meta-heuristic algorithms impress the quality of the solution obtained; thus, it seems reasonable to tune their parameters using statistical methods.
Therefore, the solution proposal, calibrated by the Taguchi method, may end up with better solutions than the ones obtained by the algorithms developed in [4, 25, 36].

d) In comparison with the algorithms in [4, 25, 36], the solution proposal employs hybridization in order to find better solution of the VMI problem at hand.

5.1. The meta-heuristics

Since annual demands are assumed triangular fuzzy numbers (TrFN), the fitness function ($TC$) will be TrFN. This function is impressed by the decision variables (the order quantities of the retailers, $q_j$, the number of replenishments for the retailers, $n_j$, and the route taken by the vendor's vehicle to reach retailers, $x_{ik}$). The general overview of using the solution algorithms that aim to obtain a near-optimum solution is as follows. The meta-heuristics first create a solution candidate with respect to their optimization approach. Then, the TrFN fitness function is evaluated. Next, the meta-heuristics find solution for the best fuzzy number of the fitness function. Finally, meta-heuristics defuzzify TrFNs to present the best solution. To defuzzify fuzzy output functions, note that while there are several other defuzzification methods such as the max-membership principle and the weighted average method, the centroid method (also called center of gravity) is used for defuzzification. The details of the two algorithms are described in the following two subsections.

5.2. Genetic algorithm

GA is one of the meta-heuristics that has been frequently utilized to find near-optimum solutions of many combinational problems. It shows a good near-optimal solution for INLP problems [28, 53]. John Holland introduced this algorithm in 1960 based on the concept of the Darwin's theory of evolution. Afterwards, his student Goldberg extended GA in the 1989 [14]. A solution in the GA is called a chromosome that contains a set of genes.
Fig. 4 presents a chromosome of a five retailers-one vendor supply chain.

![Optimal Tour](image)

**Fig. 4:** The representation of a chromosome

To start, GA generates an initial population containing $N_{pop}$ chromosomes. Then, the chromosomes are evaluated based on their fitness function values, where the ones with the better fitness values are selected as candidates for the next population. Note that since the annual demands are assumed TrFN, there are four fitness function values for each chromosome. Thus, fuzzy fitness function values are defuzzified in order to evaluate the fitness. Next, GA operators, namely crossover and mutation, generate the next population based on the crossover rate of $P_c$ and mutation rate of $P_m$, for which the roulette wheel selection method is employed. By repeating the above stages, GA terminates to a near-optimum solution. To stop GA, we used a fixed number of generations that is tuned by the Taguchi method.

Since a chromosome consists of a permutation solution, a partial-mapped crossover (PMX) is used for the crossover operator [14]. In this operator, based on the two-point crossover, two crossover points are first selected (see the first step in Fig. 5). Then, the two middle strings are exchanged with each other to create offspring (see the second step in Fig. 5). Note that the offspring with repeated genes are repaired while the replaced string remain untouched throughout the repairing process (see the third step in Fig. 5).

The mutation operator acts like as reversion, swap, and insertion operators. Moreover, the death penalty approach is utilized to make sure all chromosomes are feasibly (satisfying the constraints) generated [14]. Note again that to show the feasibility of a chromosome with
respect to the constraints, similar to the fitness function they are defuzzified. In addition, the parameters all are calibrated using the Taguchi approach described in Section 6.

Fig. 5: A partial-mapped crossover (PMX) for a tour with seven retailers

5.3. Particle swarm optimization algorithm

Stimulated by social behavior of fish school or bird flock, Kennedy and Eberhart [23] were the ones who introduced the particle swarm optimization (PSO) in 1995. Similar to GA, PSO is a population-based search algorithm. Many researchers utilized this algorithm to optimize discrete combinatorial optimization problems such as TSP. PSO is usually applied as a powerful tool to optimize problems with little changes [41]. Epitropakis et al. [12] hybridized a PSO with a differential evolution algorithm (DEA) to significantly improve it.

In PSO, the particles, which are similar to chromosomes in GA, change their position to another with respect to simple mathematical formulae. The particles have two specifications that determine their movements; first, the position (or location); second, the velocity. The velocity is impressed by the previous best position of particles ($p_i$) and the historically best position of particles ($p_g$), where the best position is determined using the fitness function.
5.3.1. Particle specifications

The velocity of \(i^{th}\) particle in iteration \(gen + 1 \ (gen = 1, \ldots, It)\) is as follows

\[
v_i^{gen+1} = w v_i^{gen} + c_1 r_1 \left( p_i - x_i^{gen} \right) + c_2 r_2 \left( p_g - x_i^{gen} \right)
\]  

(25)

where \(i\) presents the number of a particle, \(x_i^{gen}\) is the \(i^{th}\) particle location in its iteration \(gen\), and \(0 < w < 1\) is the inertia weight that is fixed here, to balance between the local and the global solution [42, 43]. Further, \(c_1 > 0\) and \(c_2 > 0\) are the acceleration coefficients and \(r_1\) and \(r_2\) are independent uniform random variates in \((0, 1)\) [54]. In Eq. (25), the first term (multiplied by \(w\)) and the second term (multiplied by \(c_1\)) on the right hand side are called the momentum component and the cognitive component, respectively. The third part is the social component (multiplied by \(c_2\)) [42, 54]. Moreover, the \(i^{th}\) particle moves to its next location with respect to its previous position and velocity based on Eq. (26).

\[
x_i^{gen+1} = x_i^{gen} + v_i^{gen+1}
\]  

(26)

Besides, Eqs. (27)-(30) are used to set the acceleration coefficients [7].

\[
\phi = \phi_1 + \phi_2 \geq 4
\]  

(27)

\[
\chi = 2 \sqrt{2 - \phi - \sqrt{\phi^2 - 4\phi}}
\]  

(28)

\[
c_1 = \chi \phi_1
\]  

(29)

\[
c_2 = \chi \phi_2
\]  

(30)

5.3.2. PSO implementation

To start, PSO produces an initial population containing \(N_{pop}\) particles. Then, the articles are appraised based on their fitness function values. Since annual demands are TrFN, there are four fitness function values in calculations of each particle. Therefore, the centroid
method defuzzifies them for evaluation. Finally, the PSO obtains new particles and evaluates them to present the best solution. The algorithm iterates in a hope the swarm (large numbers of particles) moves to a near-optimal solution. Moreover, in order to make sure all particles are feasibly produced, the death penalty approach is employed, in which the centroid method is used for defuzzification. Note that a fixed number of iterations \( I_t \), which is also tuned by the Taguchi method of Section 6, is used to stop PSO. Note also that the parameters of the PSO (i.e. \( I_t, \phi_1, \phi_2, \text{ and } N_{\text{pop}} \)) all are calibrated by the Taguchi method.

6. Tuning the parameters

The parameters of a meta-heuristic that are needed to be tuned act like controllable factors in the design of experiments (DOE). The aim is to find an optimal combination of the parameters such that the response (the fitness function) is optimized. Consequently, a set of experiments is performed in this section and the results are statistically analyzed.

There are three main methods to calibrate parameters of meta-heuristics. They are (1) fractional factorial design used for example by Shadrokh and Kianfar [40], Costa et al. [8], and Costa et al. [9], (2) response surface methodology (RSM) employed for instance by Najafi et al. [34], Pasandideh et al. [35], and Zhang et al. [56]), and (3) the Taguchi method utilized for example by Naderi et al. [33], Mousavi et al. [31], Mousavi et al. [30], Rahmati et al. [37], and Fazel Zarandi et al. [13]). As an alternative, this paper employs the Taguchi method to tune the parameters of the algorithms.

6.1. Taguchi method

The Taguchi method is one of the commonly employed approach used to optimize a response using some designed experiments that are performed based on different combinations of some controllable factors. It is the premier work in robust optimization that
requires less experiments compared to the response surface methodology (RSM). While RSM works directly with the objective function and requires more experiments using full factorial designs, the Taguchi method is a special case of the fractional factorial design in which some special orthogonal arrays are used.

In order to analyze the results obtained by the Taguchi method, there are two suggestions in the literature. First, to apply the analysis of variance for the experiments that repeat once. The second suggestion, which is used for the experiments that have multiple runs, is to use the signal to noise ratio \(S/N\) as the response variable, where \(S\) represents controllable factors and \(N\) shows noise factors affecting the response [38]. Since the meta-heuristics run several times to obtain a better solution, in this research we use \(S/N\) to analyze the results using Taguchi et al. [47].

6.1.1. Taguchi method implementation

Five steps have to be taken in the Taguchi method [38]. First, parameters with significant effects on the response are determined. Second, a trial and error procedure determines the values of the parameters that provide good fitness function to implement the experiments (mainly because Taguchi emphasized that quality should be designed at the beginning of production and not during it). Third, a suitable orthogonal array is selected with respect to available degrees of freedom (DOF). In this regard, the smallest orthogonal array corresponding to DOF should be chosen to minimize experimentation time. Fourth, the obtained design is used to implement the experiments. Finally, the results are analyzed based on the S/N approach.

The parameters of GA that have potential effects on a solution are \(N_{\text{pop}}, \text{It}, P_c, \text{and} \ P_m\). Similarly the affecting parameters in PSO are \(N_{\text{pop}}, \text{It}, \varphi_1, \text{and} \ \varphi_2\). Since the values of the factors (the third column in Tables 2 and 3) are determined by a trial and error procedure,
three levels of each factor are assumed. The total DOF of four factors each with three levels is eight; thus, the L9 orthogonal array of the Taguchi method is used in this research.

Table 2: GA parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_{\text{pop}} )</td>
<td>50 75 100</td>
</tr>
<tr>
<td>( I_t )</td>
<td>1000 1250 1500</td>
</tr>
<tr>
<td>( P_c )</td>
<td>0.75 0.8 0.85</td>
</tr>
<tr>
<td>( P_m )</td>
<td>0.05 0.10 0.15</td>
</tr>
</tbody>
</table>

Table 3: PSO parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_{\text{pop}} )</td>
<td>75 100 125</td>
</tr>
<tr>
<td>( I_t )</td>
<td>1000 1250 1500</td>
</tr>
<tr>
<td>( \phi_1 )</td>
<td>1.90 2 2.10</td>
</tr>
<tr>
<td>( \phi_2 )</td>
<td>2.10 2.20 2.30</td>
</tr>
</tbody>
</table>

Based on a supply chain consisting of one vendor and five retailers with input data given in Table 4, the results of five runs of the meta-heuristics using the L9 array are shown in Tables 5 and 6 for GA and PSO, respectively. Note that the problem is solved ten times and the five best solutions are shown in these tables. Also note that \( \tilde{d}_i \) is the trapezoidal fuzzy number of \( i^{th} \) retailer's demand for example and that "1", "2", and "3" in the first four columns of Tables 5 and 6 correspond to the first, second, and third levels of the parameters, respectively.

Table 4

<table>
<thead>
<tr>
<th>( A_i )</th>
<th>( h_i )</th>
<th>( \tilde{d}_1 )</th>
<th>( \tilde{d}_2 )</th>
<th>( \tilde{d}_3 )</th>
<th>( \tilde{d}_4 )</th>
<th>( \tilde{d}_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>6</td>
<td>495 331 319 627 407</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>7</td>
<td>969 955 977 962 1078</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>3</td>
<td>1302 1233 1211 1218 1230</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>7</td>
<td>1538 1425 1500 1590 1495</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>8</td>
<td>1538 1425 1500 1590 1495</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Transportation costs between retailers based on the distances in between \( (S_{ik}) \)

<table>
<thead>
<tr>
<th></th>
<th>( R_1 )</th>
<th>( R_2 )</th>
<th>( R_3 )</th>
<th>( R_4 )</th>
<th>( R_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>0</td>
<td>822</td>
<td>586</td>
<td>623</td>
<td>635</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>822</td>
<td>0</td>
<td>430</td>
<td>929</td>
<td>483</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>586</td>
<td>430</td>
<td>0</td>
<td>798</td>
<td>482</td>
</tr>
<tr>
<td>( R_4 )</td>
<td>623</td>
<td>929</td>
<td>798</td>
<td>0</td>
<td>599</td>
</tr>
<tr>
<td>( R_5 )</td>
<td>635</td>
<td>483</td>
<td>482</td>
<td>599</td>
<td>0</td>
</tr>
</tbody>
</table>

\( H=20; P=13297; A=208, Z=1000, f=1, F=3000, N=4r. \)
For a minimization problem, the Taguchi method aims to maximize $S/N$ using Eq. (31).

$$S/N = -10 \log_{10} \left( \frac{1}{n} \sum_{i=1}^{n} Y_i^2 \right) \quad (31)$$

where $n$ is the number of the replications (5 here) and $Y_i$ is the response variable ($TC$ here). While the $S/N$s are shown in the last column of Tables 5 and 6, Figs. 6 and 7 show them graphically. Based on the results in Tables 5 and 6, and also Figs. 6 and 7, the optimal values of the parameters that maximize $S/N$ are given in Table 7.
Note that $N_{pop}$ and $It$ are problem dependent as their optimum values must be obtained based on the size of the problem at hand. Note also that Minitab 15.1.30.0. is used to apply the Taguchi method.
7. Analysis of the solutions

In order to demonstrate the application of the proposed methodology and to analyze and compare the results obtained employing the two meta-heuristics, four measures, namely, relative deviation index (RDI), relative percentage deviation (RPD), best solution, and average solution are used in this section. RDI and the RPD are defined as [33]:

\[
RDI = \left( \frac{\text{Alg}_{sol} - \text{Min}_{sol}}{\text{Max}_{sol} - \text{Min}_{sol}} \right) \times 100
\]  
(32)

\[
RPD = \left( \frac{\text{Alg}_{sol} - \text{Min}_{sol}}{\text{Min}_{sol}} \right) \times 100
\]  
(33)

Here \( \text{Alg}_{sol} \) is the value of the fitness function that the meta-heuristics present for each problem and the worst and the best solutions are denoted by \( \text{Max}_{sol} \) and \( \text{Min}_{sol} \), respectively. Note that lower values of both RDI and RPD are indicatives of better efficiency of the solution algorithms.

Table 8 presents the values of the above measures obtained based on solving 10 problems of different sizes using the two developed meta-heuristics. In this table, \( \bar{RPD} \) and \( \bar{RDI} \) are the mean values of the first two measures and Time corresponds to the computer execution time of reaching a solution in seconds. Note that the meta-heuristics were executed on a PC with 2.2 GHz Intel Core 2 Duo CPU, and 4 GB of RAM memory with MATLAB 2010b software. Moreover, "2 Re" in the first column for example shows a supply chain problem with one vendor and 2 retailers.

**Table 7**

<table>
<thead>
<tr>
<th>Parameters GA Value</th>
<th>Parameters PSO Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_{pop} )</td>
<td>100</td>
</tr>
<tr>
<td>( I_{t} )</td>
<td>1500</td>
</tr>
<tr>
<td>( P_c )</td>
<td>0.8</td>
</tr>
<tr>
<td>( P_m )</td>
<td>0.15</td>
</tr>
</tbody>
</table>
Table 8
Performance comparison

<table>
<thead>
<tr>
<th>Prob.</th>
<th>GA</th>
<th>PSO</th>
<th>GA</th>
<th>PSO</th>
<th>GA</th>
<th>PSO</th>
<th>GA</th>
<th>PSO</th>
<th>GA</th>
<th>PSO</th>
<th>GA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av:</td>
<td>60.52</td>
<td>73.65</td>
<td>42382.32</td>
<td>40307.95</td>
<td>40445.37</td>
<td>40016.59</td>
<td>3.73</td>
<td>0.49</td>
<td>53.98</td>
<td>41.32</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

While the results in Table 8 show better performances of PSO in all the performance measures except Time, Figs. 8 and 9 provide better means to show the differences.

In order to compare the performances of the two algorithms statistically, normal probability plots are first drawn based on the results provided in Table 8, where no significant departure from normality is observed. Fig. 10 displays one of them. Then, the mean CPU times of the two algorithms are compared using the T-Test. The t-statistic that tests the alternative $\text{Time}_{GA} < \text{Time}_{PSO}$ is significant at the commonly admitted 10% level. In other words, the $p$-value of 0.091 indicates that the null hypothesis is rejected so that GA is a faster algorithm at almost 90% confidence. These strengthening evidences are illustrated in Table 9 and Figs. 11, 12, and 13. In Table 9, StDev and SE Mean indicate standard deviation and the standard error of the mean, respectively.
Fig. 8: Comparison of the means of the solutions obtained

![Comparison of the means of the solutions obtained](image)

**Table 9**

<table>
<thead>
<tr>
<th></th>
<th>Number</th>
<th>Mean</th>
<th>StDev</th>
<th>SE Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>10</td>
<td>60.5</td>
<td>19.9</td>
<td>6.3</td>
</tr>
<tr>
<td>PSO</td>
<td>10</td>
<td>73.6</td>
<td>22.3</td>
<td>7.1</td>
</tr>
</tbody>
</table>

T-Test of difference = 0 (vs<): P-Value = 0.091.

Fig. 9: Comparison of the best solutions obtained

![Comparison of the best solutions obtained](image)
Fig. 10: Normal probability plot of the average solutions obtained by PSO

Fig. 11: Box plot of Time
In order to compare the mean costs obtained by the two algorithms, namely *Average solution* and *Best solution*, two other t-Tests are employed. The t-statistics with *p*-values of
0.998 and 0.975, respectively, indicate that there is no significant differences between the means. The box plot shown in Fig. 14 can better describe this equality. Moreover, Table 10 shows the ANOVA result of comparing average solutions.

![Box plot of the cost functions obtained by GA and PSO](image)

**Fig 14:** Box plot of the cost functions obtained by GA and PSO

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor</td>
<td>3</td>
<td>34847849</td>
<td>11615950</td>
<td>0.01</td>
<td>0.998</td>
</tr>
<tr>
<td>Error</td>
<td>36</td>
<td>35464308762</td>
<td>985119688</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>39</td>
<td>35499156612</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S=31387; R-Sq = 0.10%; R-Sq(adj)=0.00%.

However, based on the results shown in Fig. 11 and 12, PSO is slower than GA in almost all cases. That is why it is improved in the next subsection in order to make it faster.

### 7.1. The improvement of the PSO: Hybrid PSO

While there are several ways such as using operators of other algorithms and local
search to improve the performances of meta-heuristics, this paper takes advantage of a GA and uses it as a local search for the PSO, called thereafter hybrid PSO. The local searcher is employed in the generating particle stage of PSO such that GA first presents feasible solutions, and then hybrid PSO uses them to obtain better results.

Table 11 and Fig. 15 present the improvement in CPU Time of employing the hybrid PSO. It can be easily seen that while there is almost no difference between the quality of the solutions obtained in terms of the best fitness (a 0.002 difference between the best solutions in Table 11 that is logical due to random nature), the hybrid PSO requires 41 percent less CPU time compared to the original PSO. Note that all the figures in Table 11 are the best among 20 runs of the algorithms. In other words, the cost functions provided by PSO and hybrid PSO are statistically equal based on the $p$-value of 0.993 in Table 12. However, the difference between the two mean CPU times is significant at the 0.10 level ($p$-value=$0.000<0.10$) based on the results shown in Table 13. Therefore, there is a significant advantage of employing the hybrid PSO. Fig. 16 shows this graphically.

### Table 11
Comparison of the improved PSO with the PSO

<table>
<thead>
<tr>
<th>Problems</th>
<th>PSO CPU Time</th>
<th>PSO Best Fitness</th>
<th>Hybrid CPU Time</th>
<th>Hybrid Best Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Re</td>
<td>42.48</td>
<td>7409.45</td>
<td>41.5</td>
<td>7409.45</td>
</tr>
<tr>
<td>3 Re</td>
<td>44.60</td>
<td>12152.66</td>
<td>42</td>
<td>12152.66</td>
</tr>
<tr>
<td>5 Re</td>
<td>45.13</td>
<td>21279.52</td>
<td>43</td>
<td>21279.52</td>
</tr>
<tr>
<td>7 Re</td>
<td>83.51</td>
<td>21778.19</td>
<td>43.5</td>
<td>21778.19</td>
</tr>
<tr>
<td>9 Re</td>
<td>94.98</td>
<td>25866.04</td>
<td>43.9</td>
<td>25992.01</td>
</tr>
<tr>
<td>11 Re</td>
<td>78.75</td>
<td>31141.51</td>
<td>43.7</td>
<td>31241.12</td>
</tr>
<tr>
<td>13 Re</td>
<td>107.58</td>
<td>54097.76</td>
<td>43.9</td>
<td>54658.87</td>
</tr>
<tr>
<td>15 Re</td>
<td>80.96</td>
<td>54033.36</td>
<td>43.9</td>
<td>54208.36</td>
</tr>
<tr>
<td>17 Re</td>
<td>76.79</td>
<td>64596.66</td>
<td>44</td>
<td>64825.66</td>
</tr>
<tr>
<td>20 Re</td>
<td>81.71</td>
<td>107810.78</td>
<td>44.14</td>
<td>107810.70</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>73.64853</strong></td>
<td><strong>40016.59206</strong></td>
<td><strong>43.354</strong></td>
<td><strong>40135.65</strong></td>
</tr>
</tbody>
</table>
Fig. 15: Comparison of the CPU Time obtained

Table 12
T-Test for CPU Time

<table>
<thead>
<tr>
<th></th>
<th>Number</th>
<th>Mean</th>
<th>StDev</th>
<th>SE Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSO</td>
<td>10</td>
<td>40017</td>
<td>30546</td>
<td>9659</td>
</tr>
<tr>
<td>Hybrid PSO</td>
<td>10</td>
<td>40136</td>
<td>30595</td>
<td>9675</td>
</tr>
</tbody>
</table>

T-Test of difference = 0 (vs not equal): P-Value = 0.993.

Table 13
Analysis of variance for CPU Time

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor</td>
<td>1</td>
<td>4589</td>
<td>4589</td>
<td>18.38</td>
<td>0.000</td>
</tr>
<tr>
<td>Error</td>
<td>18</td>
<td>4494</td>
<td>250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>9083</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S = 15.80; R-Sq = 50.52%; R-Sq(adj) = 47.77%.
Moreover, the setting of GA is $N_{pop}=10$; $It=400$; $P_e=0.8$; $P_m=0.15$, and for the PSO is $N_{pop}=30$; $It=1500$; $\phi_1=2$; $\phi_2=2.1$. They are obtained using a trial and error producer.

8. Conclusion and recommendation for future research

In this research, a combinatorial optimization model was developed for a VMI problem of a single-vendor multi-retailer supply chain with fuzzy demand. In the proposed model, the vendor managed inventories of the retailers and replenished them regarding a replenishment constraint. Not only the retailers had limited warehouse spaces, but also the vendor could not hold a large number of stocks on the average. Moreover, to reduce the transportation cost of the goods to the retailers, the proposed model finds a near-optimal route for the vendor's vehicle. The aim of this paper was to minimize the total supply chain cost including inventory and the transportation costs with respect to the VMI approach. In short, the VMI model of Zavanella and Zanoni [55] was extended to take into account the transportation cost along with several constraints.
Considering the computational complexity of the problem at hand that showed to be NP-hard, two meta-heuristics, namely particle swarm optimization (PSO) and genetic algorithm (GA), were developed to solve the problem. Furthermore, since fuzzy customer demands were modeled by trapezoidal fuzzy numbers, the centroid defuzzification method was used to defuzzify the membership functions. After tuning the parameters of the meta-heuristics by the Taguchi method, the performances of the meta-heuristics in terms of four performance measures were compared. We showed that while PSO was the better algorithm based on three measures, GA only had a better performance in terms of computer execution time. Keeping the solution quality, we made PSO significantly faster by using a local searcher called GA at the end.

Some recommendations for future research follow

- The model can be extended to consider shortages and discounts as well.
- A multi-vendor multi-retailer multi-warehouse VMI problem can be investigated.
- Instead of Taguchi, response surface methodology can be used to tune the parameters.

9. Acknowledgment

The authors are thankful for constructive comments of the anonymous reviewers. Taking care of the comments certainly improved the presentation of the manuscript.

References


