Scientia Iranica E (2016) 23(3), 1429–1440

Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
www.scientiairanica.com

Solving a multi-stage multi-product solid supply chain network design problem by meta-heuristics

A. Mahmoodirad* **and M. Sanei**

Department of Mathematics, Central Tehran Branch, Islamic Azad University, Tehran, Iran.

Received 6 October 2014; received in revised form 10 May 2015; accepted 16 June 2015

KEYWORDS
Supply chain network design;
Differential evolution;
Particle swarm optimization algorithm;
Gravitational search algorithm;
Taguchi experimental Design.

Abstract. This paper presents an effective optimization method based on meta-heuristics algorithms for the design of a multi-stage, multi-product solid supply chain network design problem. First, a mixed integer linear programming model is proposed. Second, because the problem is an NP-hard, three meta-heuristics algorithms, namely Differential Evolution (DE), Particle Swarm Optimization (PSO), and Gravitational Search Algorithm (GSA), are developed for the first time for this kind of problem. To the best of our knowledge, neither DE, nor PSO, nor GSA have been considered for the multi-stage solid supply chain network design problems. Furthermore, the Taguchi experimental design method is used to adjust the parameters and operators of the proposed algorithms. Finally, to evaluate the impact of increasing the problem size on the performance of our proposed algorithms, different problem sizes are applied and the associated results are compared with each other.

1. Introduction

The Supply Chain Network (SCN) design problem is an important strategic issue in supply chain management that has recently drawn the focus of many researchers [1-4].

SCN is widely used and includes all activities in the field of production and the final product provides the service of distribution of the most elementary stage, i.e. from the primary stage of raw materials, to the most final stage, i.e. delivery to the customer and even worn out product recycling. Supply chain management includes managing supply and demand, supply of components and raw materials, manufacturing and assembly, storage and shipping of inventory, order management, distribution channels, and supply and delivery to the customer. Nowadays, service providers and products, distribution channels (distributors and wholesalers), and customers as well as supply chain management consultants, system developers and suppliers of software products, and supply chain managers are all key elements. Achieving the effective supply chain management is dependent on the cooperation of supply chain members.

A Multi-stage Supply Chain Network (MSCN) can be modeled by means of a sequence of multiple SCN stages for production of multi-product so that the flow would be transferred only between two successive stages. Since the MSCN is difficult to solve optimally [5], many researchers have developed heuristic and meta-heuristic approaches to solve it. The work done in this regard is as follows.

Jayaraman and Pirkul [6] have presented an efficient heuristic approach based on the Lagrangean relaxation for the single-source, multi-product, multi-stage SCN design problem. They use this heuristic method to evaluate the performance of the model with respect to solution quality and algorithm performance. Syam [7] focused on a heuristic method proposed based

*. *Corresponding author. Tel.: +98 21 66434096; Fax: +98 21 66434099*
E-mail addresses: alimahmoodirad@yahoo.com (A. Mahmoodirad); masoudsanei49@yahoo.com (M. Sanei)

on Lagrangean relaxation and simulated annealing for a multi-source, multi-product, multi-location framework. Another heuristic approach based on steady-state genetic algorithm has been developed by Altiparmak et al. [5] for a single-source, multi-product, multi-stage SCN design problem. They propose two different encoding approaches to represent a solution to the problems: priority-based encoding and integer encoding. The priority-based encoding is used for the first two stages of SCN and integer encoding is used in the last stage. Moreover, the efficiency and effectiveness of the algorithm have been investigated by comparing its results with those of other methods such as CPLEX, Lagrangean heuristic, hybrid genetic algorithm, and simulated annealing on a set of SCN design problems with different sizes.

Mehdizadeh and Afrabandpei [8] have proposed a mixed integer nonlinear programming model for the multi-stage, multi-product network design problem to minimize the total cost of supply chain. They have developed a hybrid priority-based Genetic Algorithm (GA) and simulated annealing algorithm to find optimal solution in two phases. In the first phase, the optimal routes are determined by the use of GA. In the second phase, they use the SA algorithm for convergence speed. They use a matrix and vector to represent the solution. The obtained results have shown that the proposed algorithms can find near optimal solutions in reasonable time spans.

Kadadevaramath et al. [9] have presented an integer linear programming model for the constrained three echelons SCN problem to minimize the total supply chain operating cost. They have used four algorithms based on a Particle Swarm Optimization (PSO) algorithm and Genetic Algorithm (GA) for solving the problem and the obtained results of PSO algorithms have been compared with those of GA.

Olivares-Benitez et al. [10] addressed a supply chain design problem based on a two-echelon single-product system. The meta-heuristic algorithm was proposed to solve the problem, which combined principles of greedy functions, Scatter search, Path relinking, and Mathematical programming.

Mehdizadeh et al. [3] considered an integrated multi-stage, multi-product logistic network design problem which included forward and reverse logistics and proposed a mixed integer nonlinear programming model. To find the proper solutions, they developed two meta-heuristic algorithms, namely hybrid priority-based genetic algorithm and simulated annealing algorithm. In order to tune the significant parameters of the algorithms, they used the response surface methodology.

Crdenas-Barron and Trevino-Garza [11] developed a more general mathematical model proposed by Kadadevaramath et al. [9] when the number of periods and products was one. They solved all instances in Kadadevaramath et al. [9] by CPLEX and showed that all instances could easily be solved optimally by any integer linear programming solver.

Kristianto et al. [12] developed a supply chain network by optimizing inventory allocation and transportation routing. They proposed a fuzzy shortest path into two-stage programming in order to find the global optimum solution.

Khalifehzadeh et al. [13] considered a four-echelon supply chain network design with shortage. They presented a multi-objective mathematical model to minimize the total operating costs of all the supply chain elements and to maximize reliability of the system. They solved this problem by a comparative particle swarm optimization algorithm.

The multi-product, multi-stage solid SCN design problem considered in this paper consists of three stages: supplier, plant, DC and customer. The problem is to determine the optimal transportation network in order to satisfy the customer demands of products by using several kinds of conveyance with minimum costs. To this end, firstly, we propose a mixed integer programming model for the multi-product, multi-stage solid SCN design problem, in which the objective is minimization of the total costs of supply chain. Secondly, due to complexity of the problem, we develop three meta-heuristic algorithms, namely Differential Evolution (DE), Particle Swarm Optimization (PSO), and Gravitational Search Algorithm (GSA), for this problem. Furthermore, the Taguchi experimental design method is used to adjust the parameters and operators of the proposed algorithms. Finally, to evaluate the impact of increasing the problem size on the performance of our proposed algorithms, different problem sizes are applied and the associated results are compared with each other.

The rest of this paper is arranged as follows: In Section 2, we describe the mathematical model and descriptions. Section 3 explains the proposed solution approaches. Section 4 describes the Taguchi experimental design and compares the computational results. Finally, in Section 5, conclusions are made and provided.

2. Problem description and mathematical model

The considered problem can formally be described as follows:

The multi-stage, multi-product solid SCN design problem can consist of suppliers, plants, DCs, and customers. In the first stage, the suppliers provide the raw materials for the plants to produce multi-product. In the second stage, the plants produce and send the products to DCs. Finally, the DCs transport the products to the customers. Also, conveyances can be considered as transportation types so that each conveyance would be related to the cost, and one must be selected to transport the products to each stage. The objective is minimization of the total costs of supply chain that will satisfy all capacities and demand requirement for each product imposed by customers. We formulated this problem as a mixed-integer nonlinear programming model. The assumption used in this problem is as follows:

- The number of suppliers, the maximum number of plants, the maximum number of DCs, and the number of conveyances are known;
- The capacities of suppliers, plants, and DCs are known;
- The number of customers and their demands are known.

The following notations are used to define the mathematical model:

Set of indices:

R	Set of raw materials ($r = 1, 2, \cdots, R$);
P	Set of products ($p = 1, 2, \cdots, P$);
S	Set of suppliers ($s = 1, 2, \cdots, S$);
I	Set of plants ($i = 1, 2, \cdots, I$);
J	Set of DCs ($j = 1, 2, \cdots, J$);
K	Set of customers ($k = 1, 2, \cdots, K$);
M	Set of conveyances in the first stage ($m = 1, 2, \cdots, M$);
N	Set of conveyances in the second stage ($n = 1, 2, \cdots, N$);
L	Set of conveyances in the third stage ($l = 1, 2, \cdots, L$).

Parameters:

E_{sr}	Capacity of supplier s for raw material r;
D_i	Capacity of plant i;
W_j	Capacity of DCj;
C_{pk}	Demand for product p at customer k;
u_{rp}	Utilization rate of raw material r per unit of product p;
F_i	Fixed cost for operating a plant i;
G_j	Fixed cost for operating a DCj;
a_{rsim}	Cost of transporting and purchasing for raw material r from supplier s to plant i by conveyance m;
b_{pijn}	Cost of transporting one unit of product p from plant i to DCj by conveyance n;
c_{pjkl}	Cost of sending one unit of product p from DCj to customer k by conveyance l;
f_{sim}	Fixed cost of transporting for raw materials from supplier s to plant i by conveyance m;
g_{ijn}	Fixed cost of transporting products from plant i to DCj by conveyance n;
h_{jkl}	Fixed cost of sending products from DCj to customer k by conveyance l;
v_i	Unit production cost of product at plant i;
v'_j	Unit storing cost of product at DCj;
$E^{(1)}_m$	Maximum capacity of conveyance m in the first stage;
$E^{(2)}_n$	Maximum capacity of conveyance n in the second stage;
$E^{(3)}_l$	Maximum capacity of conveyance l in the third stage.

Decision variables:

α_i	Binary variable equal to 1 if plant i is opened and equal to 0 otherwise;
β_j	Binary variable equal to 1 if DCj is opened and equal to 0 otherwise;
x_{rsim}	Quantity of raw material r shipped from supplier s to plant i by conveyance m;
y_{pijn}	Quantity of product p shipped from plant i to DCj by conveyance n;
z_{pjkl}	Quantity of product p shipped from DCj to customer k by conveyance l;
$t^{(1)}_{sim}$	Binary variable equal to 1 if $\sum_{r=1}^{R} x_{rsim} > 0$ and equal to 0 otherwise;
$t^{(2)}_{ijn}$	Binary variable equal to 1 if $\sum_{p=1}^{P} y_{pijn} > 0$ and equal to 0 otherwise;
$t^{(3)}_{jkl}$	Binary variable equal to 1 if $\sum_{p=1}^{P} z_{pjkl} > 0$ and equal to 0 otherwise.

The mathematical model of the multi-stage, multi-product solid SCN design problem is as follows:

$$\text{Min} Z = \sum_{r=1}^{R}\sum_{s=1}^{S}\sum_{i=1}^{I}\sum_{m=1}^{M} a_{rsim}x_{rsim}$$

$$+ \sum_{p=1}^{P}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{n=1}^{N} b_{pijn}y_{pijn}$$

$$+\sum_{p=1}^{P}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{l=1}^{L} c_{pjkl} z_{pjkl}$$

$$+\sum_{s=1}^{S}\sum_{i=1}^{I}\sum_{m=1}^{M} f_{sim} t_{sim}^{(1)} + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{n=1}^{N} g_{ijn} t_{ijn}^{(2)}$$

$$+\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{l=1}^{L} h_{jkl} t_{jkl}^{(3)} + \sum_{i=1}^{I} F_i \alpha_i + \sum_{j=1}^{J} G_j \beta_j$$

$$+\sum_{p=1}^{P}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{n=1}^{N} v_i y_{pijn}$$

$$+\sum_{p=1}^{P}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{n=1}^{N} v'_j y_{pijn}, \qquad (1)$$

subject to:

$$\sum_{i=1}^{I}\sum_{m=1}^{M} x_{rsim} \le E_{sr} \qquad \forall s, r, \qquad (2)$$

$$\sum_{p=1}^{P}\sum_{j=1}^{J}\sum_{n=1}^{N} y_{pijn} \le D_i \alpha_i \qquad \forall i, \qquad (3)$$

$$\sum_{p=1}^{P}\sum_{j=1}^{J}\sum_{n=1}^{N} u_{rp} y_{pijn} \le \sum_{s=1}^{S}\sum_{m=1}^{M} x_{rsim} \qquad \forall r, i, \qquad (4)$$

$$\sum_{p=1}^{P}\sum_{i=1}^{I}\sum_{n=1}^{N} y_{pijn} \le W_j \beta_j \qquad \forall j, \qquad (5)$$

$$\sum_{k=1}^{K}\sum_{l=1}^{L} z_{pjkl} \le \sum_{i=1}^{I}\sum_{n=1}^{N} y_{pijn} \qquad \forall j, p, \qquad (6)$$

$$\sum_{j=1}^{J}\sum_{l=1}^{L} z_{pjkl} \ge C_{pk} \qquad \forall p, k, \qquad (7)$$

$$\sum_{r=1}^{R}\sum_{s=1}^{S}\sum_{i=1}^{I} x_{rsim} \le E_m^{(1)} \qquad \forall m, \qquad (8)$$

$$\sum_{p=1}^{P}\sum_{i=1}^{I}\sum_{j=1}^{J} y_{pijn} \le E_n^{(2)} \qquad \forall n, \qquad (9)$$

$$\sum_{p=1}^{P}\sum_{j=1}^{J}\sum_{k=1}^{K} z_{pjkl} \le E_l^{(3)} \qquad \forall l, \qquad (10)$$

$$\alpha_i \in \{0,1\} \qquad \forall i, \qquad (11)$$

$$\beta_j \in \{0,1\} \qquad \forall j, \qquad (12)$$

$$t_{sim}^{(1)} \in \{0,1\} \qquad \forall s, i, m, \qquad (13)$$

$$t_{ijn}^{(2)} \in \{0,1\} \qquad \forall i, j, n, \qquad (14)$$

$$t_{jkl}^{(3)} \in \{0,1\} \qquad \forall j, k, l, \qquad (15)$$

$$x_{rsim}, y_{pijn}, z_{pjkl} \ge 0 \qquad \forall r, p, s, i, j, k, m, n, l. \qquad (16)$$

In this model, objective function (1) minimizes the total cost of supply chain network. Constraint (2) is the capacity constraint for the suppliers. Constraint (3) gives the plant capacity constraint. Constraint (4) gives the raw material requirement. Constraint (5) is the capacity constraint for DCs. Constraint (6) limits the total quantity of products shipped from a DC to customers and cannot exceed the amount of shipped products in that DC. Constraint (7) represents demand satisfaction for each customer. Constraints (8)-(10) give capacity constraint for conveyance in the first, second, and third stages, respectively. Ultimately, Constraint sets (11)-(16) define the decision variables.

Since the problem is minimization of the objective function, in the optimal solution, no extra products or raw materials are transported at various stages of the supply chain network. Thus, in the optimal solution of the equality, Constraints (4), (6), and (7) are established.

3. Solution approach

Although the exact algorithms such as Dynamic Programming, local search techniques, Branch-and-Cut, Branch-and-Bound, Branch-and-Price, and Lagrangean relaxation guarantee the optimal solution or prove that no feasible solution exists, the real-world problems are time consuming. Therefore, the meta-heuristic algorithms to find the near optimal solution in a reasonable time have been proposed by researchers. The meta-heuristics are simple, easy to implement, robust, and have proven to be highly effective to solve many optimization problems [14].

Since the single-stage fixed cost transportation problem can be categorized as NP-hard [15,16], the multi-stage, multi-product solid SCN design problem is NP-hard, too. In this section, first, the solution representation is described and then three meta-heuristic algorithms are developed to find the near optimal solutions.

3.1. Encoding scheme and initialization

The encoding scheme plays a very important role in the effectiveness of the meta-heuristic algorithms. In fact, it is the approach of making a solution recognizable for the meta-heuristic algorithms. Among different methods of encoding, the priority-based encoding has successfully been applied for many optimization problems [3,5,17-19]. It needs no repairing process and it belongs to the permutation encoding category [19].

In the single-stage fixed cost solid transportation problem, we have two three-dimensional cost matrices, namely the three-dimensional variable cost matrix and the three-dimensional fixed cost matrix. Therefore, selecting a route with minimum variable cost will not give good solutions.

When the priority-based encoding is utilized for the single-stage, single-product fixed cost solid transportation problem, a solution consists of priorities of sources (M), depots (N), and conveyances (K); in this case, the solution length is equal to $|Q| = |M| + |N| + |K|$. In the single-stage, multi-product fixed cost solid transportation problem, consider P to be the set of products. In this case, the solution based on the priority-based encoding consists of $|P|$ parts and the length of each part is equal to $|P| \times |Q|$, and the digit values of the solution are between 1 and $|P| \times |Q|$. To obtain the priority-based encoding, in the single-stage, multi-product fixed cost solid transportation problem, a priority assignment to nodes is started from the highest value ($|P| \times |Q|$) and it is reduced by one until assigning a priority to all nodes.

In this paper, we develop the priority-based decoding procedure developed by Gen et al. [17,20] and Altiparmak et al. [5] to adapt to the single-stage, multi-product fixed cost solid transportation problem. The procedure to decode the solution of the single-stage, multi-product fixed cost solid transportation problem is shown in Figure 1.

In the multi-stage, multi-product solid SCN prob-

```
Procedure 1: Decoding of solution
Input:
$M$: Set of sources;
$N$: Set of depots;
$K$: Set of conveyances;
$P$: Set of products;
$L$: Set of customers;
$a_i$: Supply of source $i$, $\forall i \in M$;
$b_j$: Demand of depot $j$, $\forall j \in N$;
$e_k$: Capacity of conveyance $k$, $\forall k \in K$;
$n_p$: Requirement rate of product $p$ on a source, $\forall p \in P$;
$C_{lp}$: Demand of customer $l$ for product $p$, $\forall l \in L$, $\forall p \in P$;
$TD$: Total remained demand of costumers that is equal to $\sum_{p=1}^{P}\sum_{l=1}^{L} C_{lp}$;
$c_{pijk}$: Variable transportation cost of product $p$ from source $i$ to depot $j$ by conveyance $k$;
$g_{ijk}$: Fixed cost of transporting products associated with route $(i,j,k)$;
$v(p(i+j+k))$: Solution, for each $p$, $i$, $j$, $k$,
Output: $x_{pijk}$: The amount of transported product $p$ from source $i$ to depot $j$ by conveyance $k$,
  Step 1:  $x_{pijk} \leftarrow 0$, for each $p$, $i$, $j$, $k$;
  Step 2:  $q \leftarrow \arg\max\{v(t) : t = 1, \cdots, |P| \times (|M|+|N|+|K|)\}$ determine a product;
  Step 3:  $p^* = \left\lceil \frac{q}{|M|+|N|+|K|} \right\rceil$: select the product type;
  Step 4:  If $q \leq |M|$ for product $p^*$, then $i^* \leftarrow q$: select a source,
           $(j^*, k^*) \leftarrow \arg\min\{C_{i^*jk} = c_{i^*jk} + \frac{g_{i^*jk}}{\min\{a_{i^*}, b_j, c_k\}} | v((p^*-1)$
           $\times(|M|+|N|+|K|) + |M| + j) \neq 0, j \in N$,
           $v((p^*-1) \times (|M|+|N|+|K|) + |M| + |N| + k) \neq 0, k \in K\}$:
           Select a depot and a conveyance with lowest cost, else if
           $|M| < q \leq |M| + |N|$, then $j^* \leftarrow q - |M|$: select a depot,
           $(i^*, k^*) \leftarrow \arg\min\{C_{ij^*k} = c_{ij^*k} + \frac{g_{ij^*k}}{\min\{a_i, b_{j^*}, c_k\}} | v((p^*-1)$
           $\times(|M|+|N|+|K|) + i) \neq 0, \quad i \in M$,
           $v((p^*-1) \times (|M|+|N|+|K|) + |M| + |N| + k) \neq 0, k \in K\}$:
           Select a source and a conveyance with lowest cost, else
           $k^* \leftarrow q - |M| - |N|$: select a conveyance,
           $(i^*, j^*) \leftarrow \arg\min\{C_{ijk^*} = c_{ijk^*} + \frac{g_{ijk^*}}{\min\{a_i, b_j, c_{k^*}\}} | v((p^*-1)$
           $\times(|M|+|N|+|K|) + i) \neq 0, i \in M$,
           $v((p^*-1) \times (|M|+|N|+|K|) + |M| + j) \neq 0, j \in N\}$:
           Select a source and a depot with lowest cost;
  Step 5:  $x_{i^*j^*k^*} \leftarrow \min\left\{ \left\lceil \frac{a_{i^*}}{n_{p^*}} \right\rceil, b_{j^*}, c_{k^*}, TD \right\}$: Assign available amount of units;
  Step 6:  Update availabilities of source $i^*$, depot $j^*$ and conveyance $k^*$;
           $a_{i^*} \leftarrow a_{i^*} - x_{i^*j^*k^*} \times n_{p^*}; b_{j^*} \leftarrow b_{j^*} - x_{i^*j^*k^*}; c_{k^*} \leftarrow c_{k^*} - x_{i^*j^*k^*}; TD = TD - x_{i^*j^*k^*}$:
  Step 7:  Remove priority of selected source, depot and conveyance,
           If $a_{i^*} = 0$ then $v((p^*-1) \times (|M|+|N|+|K|) + i^*) \leftarrow 0$,
           If $b_{i^*} = 0$ then $v((p^*-1) \times (|M|+|N|+|K|) + |M| + j^*) \leftarrow 0$,
           If $c_{k^*} = 0$ then $v((p^*-1) \times (|M|+|N|+|K|) + |M| + |N| + k^*) \leftarrow 0$;
  Step 8:  While $TD > 0$, go to Step 2, else calculate the total transportation cost.
```

Figure 1. Priority-based decoding procedure for the SFCSTP.

lem, a solution consists of three segments. The solution length is equal to the total length of segment one $|Q_1| = |R| \times (|S| + |I| + |M|)$, segment two $|Q_2| = |P| \times (|I| + |J| + |N|)$, and segment three $|Q_3| = |P| \times (|J| + |K| + |L|)$. Therefore, total length of the solution is equal to $|Q_1| + |Q_2| + |Q_3|$. We developed the above procedure to the multi-stage, multi-product solid SCN problem. The overall decoding procedure of the three-stage multi-product solid SCN problem and decoding procedures for the 3rd, 2nd, and 1st stages, respectively, are presented in Figures 2-5.

3.2. Differential evolution algorithm

In this section, we develop the DE algorithm to use for the problem. DE is a very simple and powerful stochastic population-based search method, proposed by Storn and Price [21], whose main objective is functions optimization. Due to simplicity of implementation, reliability, and robustness, DE is considered as an effective global optimization algorithm [22-25].

The key idea behind DE is a scheme for generating trial vectors. Like other evolutionary-type algorithms, it contains three basic operators: mutation, crossover, and selection.

DE starts with an initial population of individuals (candidate solutions) randomly selected from the search space. It is usual to denote each individual by a D-dimensional vector called a target vector.

$$X_{i,G} = (X^1_{i,G}, \cdots, X^D_{i,G}), \qquad i = 1, 2, \cdots, NP,$$

where NP denotes the population size and D denotes the number of variables. Then, mutation and crossover operators are applied to generate new candidate vectors and a selection operator is employed to determine

```
Procedure 2: Decoding of solution for the multi-stage,
multi-product solid SCN
  Step 1:   Find $z_{pjkl}$ by Procedure 3 (3th stage decoding);
  Step 2:   Find $y_{pijn}$ by Procedure 4 (2th stage decoding);
  Step 3:   Find $x_{rsim}$ by Procedure 5 (1th stage decoding);
  Step 4:   Calculate the value of objective function ($Z$)
  and stop.
```

Figure 2. Decoding procedure for the priority based encoding.

```
Procedure 3: 3rd stage decoding
input:
$J$: Set of DCs;
$K$: Set of customers;
$L$: Set of conveyances;
$P$: Set of products;
$W_j$: Capacity of DC $j$, $\forall j \in J$;
$C_{pk}$: Demand for product $p$ at costumer $k$, $\forall k \in K$;
$E^{(3)}_l$: Maximum capacity of conveyance $l$, $\forall l \in L$;
$c_{pjkl}$: Shipping cost of one unit product $p$ from DC $j$
      to customer $k$ by conveyance $l$;
$v_3(p \times (j + k + l))$: Solution, $\forall j \in J, \forall k \in K, \forall l \in L, \forall p \in P$.
Output:
$z_{pjkl}$: The amount of shipment from DC $j$ to customer $k$
      by conveyance $l$;
$W'_{pj}$: Total customer demand for product $p$ on DC $j$,
      $\forall j \in J, \forall p \in P$.
  Step 1:   Set DCs as sources, the customers as depots and
            the conveyances as conveyances; call Procedure 1
            to obtain $z_{pjkl}$; calculate $W'_{pj}$
            considering $z_{pjkl}$ and return.
```

Figure 3. Decoding procedure for third segment of the solution.

```
Procedure 4: 2nd stage decoding
Input:
$I$: Set of plants;
$J$: Set of DCs;
$N$: Set of conveyances;
$P$: Set of products;
$D_i$: Capacity of plant $i$, $\forall i \in I$;
$W'_{pj}$: Total customer demand for product $p$ on DC $j$,
      $\forall j \in J, \forall p \in P$;
$E^{(2)}_n$: Capacity of conveyance $n$, $\forall n \in N$;
$r_p$: Capacity utilization rate per unit of product $p$;
$b_{pijn}$: Shipping cost of one unit product from plant $i$
      to DC $j$ by conveyance $n$, $\forall p \in P, \forall i \in I$,
      $\forall j \in J, \forall n \in N$;
$v_2(p \times (i + j + n))$: Solution $\forall p \in P, \forall i \in I, \forall j \in J, \forall n \in N$.
Output:
$y_{pijn}$: The amount of shipment from plant $i$ to DC $j$
      by conveyance $n$;
$y_{pijn}$: Total customer demand for product $p$ on plant $i$,
      $\forall i \in I, \forall p \in P$.
  Step 1:   Set plants as sources, the DC as depots and
            the conveyances as conveyances; call Procedure 1
            to obtain $y_{ijn}$; Calculate $D'_{pi}$
            considering $y_{pijn}$ and return.
```

Figure 4. Decoding procedure for second segment of the solution.

```
Procedure 5: 1st stage decoding
Input:
$S$: Set of suppliers;
$I$: Set of plants;
$M$: Set of conveyances;
$R$: Set of raw materials;
$E_{sr}$: Capacity of supplier $s$ for raw material $r$;
$D'_{pi}$: Total customer demand for product $p$ on plant $i$,
      $\forall \in I, \forall p \in P$;
$E^{(1)}_m$: Capacity of conveyance $m$, $\forall m \in M$,
$a_{rsim}$: Cost of transporting and purchasing for raw material
      $r$ from supplier $s$ to plant $i$ by conveyance $m$;
$u_{rp}$: Utilization rate of raw material $r$ per unit of product $p$;
      $\forall r \in R, \forall p \in P$;
$a_{ri}$: The amount of raw material $r$ to produce all products
      on plant $i$;
$v_1(r \times (s + i + m))$: Solution, $\forall r \in R$, $\forall s \in S$, $\forall i \in I$, $\forall m \in M$;
Output:
$x_{rsim}$: The amount of raw material $r$ shipped from supplier $s$ to
      plant $i$ by conveyance $m$, $\forall r \in R$, $\forall s \in S$, $\forall i \in I, \forall m \in M$,
Step 1:   $x_{rsim} \leftarrow 0$, $\forall r \in R$, $\forall s \in S$, $\forall i \in I$, $\forall m \in M$;
          calculate the amount of raw material $r$ to produce all
          products on plant $i$, $a_{ri} = \sum_{p=1}^{P} D'_{pi} u_{rp}$, $\forall i \in I$, $\forall r \in R$,
Step 2:   Set supplier as sources, the plants as depots
          and the conveyances as conveyances; call Procedure 1
          to obtain $x_{rsim}$ and return.
```

Figure 5. Decoding procedure for first segment of the solution.

whether the offspring or the parent survives in the next generation. The above process is repeated until a termination criterion is reached. According to Storn and Price [21], the strategy of DE is described below.

3.2.1. Mutation operator

For each target vector $X_{i,G}$, $i = 1, 2, \cdots, NP$, a mutant vector $V_{i,G}$ is generated according to the following scheme:

$$V_{i,G} = X_{r_1,G} + F(X_{r_2,G} - X_{r_3,G}),$$

$$r_1 \neq r_2 \neq r_3 \neq i,$$

with randomly chosen indices and $r_1, r_2, r_3 \in \{1, 2, \cdots, NP\}$. Note that these indices have to be different from each other and from the running index i so that NP must be at least 4. According to Storn and Price [21], $F \in [0, 2]$ is to control the amplification of the difference vector.

3.2.2. Crossover operator

In order to increase diversity of the perturbed parameter vectors, crossover is introduced after the mutation operation. The target vector is mixed with the mutated vector to get the trial vector $U_{i,G+1}$ according to the following [21]:

$$U_{i,j,G+1} = \begin{cases} V_{i,j,G+1} & \text{if } rand(j) \leq CR \text{ or} \\ & j = rand(i) \\ X_{i,j,G+1} & \text{if } rand(j) > CR \text{ or} \\ & j \neq rand(i) \end{cases}$$

where $rand(j)$ is the jth evaluation of a random number uniformly distributed in the range $[0, 1]$ and rand$n(i)$ is a randomly chosen index from the set $\{1, 2, \cdots, N\}$. $CR \in [0, 1]$ is a crossover constant rate that controls the diversity of the population. The more the value of CR, the less the influence of the parent will be.

3.2.3. Selection operator

To generate better offspring for the next generation, selection operation is performed between each individual and its corresponding trial vector by the following greedy selection criterion:

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & \text{if } f(U_{i,G+1}) < f(X_{i,G}), \\ X_{i,G} & \text{otherwise,} \end{cases}$$

where f is the objective function and $X_{i,G+1}$ is the individual of the new population.

The steps of DE are shown in Figure 6.

Steps of DE	
Step 1	Initialization of individuals;
Step 2	Evaluation;
Step 3	Differential mutation;
Step 4	Differential crossover;
Step 5	Evaluation;
Step 6	Selection;
Step 7	Repeat Steps 3 to 6 until a stopping criterion is met.

Figure 6. Steps of the DE algorithm.

3.3. Particle swarm optimization algorithm

The PSO was first introduced by Kennedy and Eberhart [26] to simulate the social behavior of animals as a population based meta-heuristic. The PSO has rapid convergence speed and provides appropriate way for performing global search [27]. In PSO, the social interaction of a population is imitated in the sense that the individuals, the so called particles, are encouraged to move toward the best individual for finding the best position. Hence, the behavior of each individual is formed by the personal and social information.

Each particle has its own position vector $x_i(t)$, velocity vector $v_i(t)$, and best positions $p_i(t)$. First random positions in range (0, positive number] and with velocities between $v_{\min}$ and $v_{\max}$ are generated. In each generation, the velocities and positions are updated to their best encountered position and the global best position encountered, $p_g(t)$, according to the following equations:

$$v_i(t) = w \times v_i(t-1) + c_1 \times r_1(t) \times (p_i(t) - x_i(t))$$
$$+ c_2 \times r_2(t) \times (p_g(t) - x_i(t)),$$

$$x_i(t) = v_i(t-1) + v_i(t),$$

where w is representative of the inertia weight, c_1 is cognition learning factor, c_2 is social learning factor, and $r_1(t)$ and $r_2(t)$ are two random deviates within (0,1). The inertia weight is determined based on the following equation:

$$w = w_{\max} - \frac{(w_{\max} - w_{\min})}{st} \times ct,$$

where $w_{\max}$ and $w_{\min}$ are higher and lower inertia weight values, st and ct show the solving time and the current time, respectively. The steps of PSO are shown in Figure 7.

3.4. Gravitational search algorithm

GSA has first been introduced by Rashedi et al. [28] and is a novel optimization algorithm based on Newton's laws of gravity and the law of motion. This

Steps of PSO	
Step 1	Initialize parameters;
Step 2	Initialize population;
Step 3	Evaluate fitness value;
Step 4	Find particle best;
Step 5	Find global best;
Step 6	Update velocity;
Step 7	Update position;
Step 8	Evaluate;
Step 9	Repeat Steps 4 to 8 until a stopping criterion is met.

Figure 7. Steps of the PSO algorithm.

algorithm has recently been used for many optimization problems [23,29,30]. In GSA, each mass has four attributes: position, inertial mass, active gravitational mass, and passive gravitational mass. The position of the mass is consistent with a solution of the optimization problem and moving the position of agent results in an improvement of the solution's quality [31]. All masses are determined by using the evaluation function.

To describe GSA in depth, consider a system with N masses in which position of the ith mass is defined as Eq. (17):

$$X_i = (x_i^1, \cdots, x_i^d, \cdots, x_i^n) \qquad i = 1, 2, \cdots, N, \tag{17}$$

where x_i^d denotes position of the ith mass in the dth dimension. Then, the force of gravity on the mass i from the mass j at a specific time t is defined as Eq. (18):

$$F_{ij}^d(t) = G(t) \times \frac{M_i(t) \times M_j(t)}{R_{ij}(t) + \mathcal{E}} \times \left(x_j^d(t) - x_i^d(t)\right), \tag{18}$$

where $M_i(t)$ and $M_j(t)$ are the masses of agent i and agent j, respectively, $\mathcal{E}$ is a small constant, $R_{ij}(t)$ is the Euclidean distance between two agents i and j at time t equal to $R_{ij}(t) = ||x_i(t), x_j(t)||_2$, and $G(t)$ is a function of the initial value (G_0) and time t; it will be reduced with time given in Eq. (19):

$$G(t) = G_0 \exp\left(-\alpha \frac{t}{T}\right). \tag{19}$$

In Eq. (3), α is a user-specified constant, t is the current iteration, and T is the total number of iterations [32]. The total force used for agent i at time t in the dth dimension is presented in Eq. (20):

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i}^{N} rand_j \times F_{ij}^d(t), \tag{20}$$

where $rand_j$ is a uniformly distributed random variable in the interval [0,1] and $Kbest$ is the set of the first K agents with the best fitness value and the biggest mass. At the starting, $Kbest$ is initialized at K_0 and linearly reduced step-by-step as time lapses.

Regarding the law of motion, the force that accelerates the agent i is given as Eq. (21):

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}, \tag{21}$$

where $M_i(t)$ is the inertial mass of the agent i. The next velocity and the next position of agent i in dimension d are computed in Eqs. (22) and (23):

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t), \tag{22}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \tag{23}$$

where in Eq. (6), $rand_i$ is a uniformly distributed random variable in the interval [0,1]. This random number is used to give a randomized characteristic to the search and $v_i^d(t)$ and $x_i^d(t)$ are its current velocity and position, respectively. The masses of agents are evaluated by the fitness function. Assuming the equality of the gravitational and inertia mass, the mass $M_i(t)$ is updated by Eqs. (24), (25), and (26):

$$M_i = M_{ii} \qquad i = 1, 2, \cdots, N, \tag{24}$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}, \tag{25}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \tag{26}$$

where $fit_i(t)$ represent the fitness value of the agent i at time t, $best(t)$ and $worst(t)$ are the best and the worst values of the fitness function at time t and for a minimization problem are defined in Eqs. (27) and (28).

$$best(t) = \min_{j \in 1,2,\cdots,N} fit_j(t), \tag{27}$$

$$worst(t) = \max_{j \in 1,2,\cdots,N} fit_j(t). \tag{28}$$

The steps of GSA are shown in Figure 8.

4. Experimental design

4.1. Instances

To evaluate the effectiveness of the proposed algorithms, we were required to generate some test problems. The data needed for this problem include the number of products, the number of raw materials, suppliers, plants, DCs, customers, conveyances, total capacity of suppliers, plants and conveyance, total demand of customers, range of variables, and fixed costs. The experimental design is shown in Table 1. Problem sizes are determined by the number of products, raw materials, suppliers, plants, DCs, customers, and conveyances in each stage. Also, the utilization rate of raw material r per unit of product p, namely u_{rp}, is selected from a uniform distribution of $U(0.5, 1.5)$.

Steps of GSA	
Step 1	Population-based initialization;
Step 2	Fitness evaluation of the agents;
Step 3	Updating $M_i(t)$, $best(t)$, $worst(t)$ and $G(t)$ for $i = 1, 2, \cdots, N$;
Step 4	Calculation of total forces in different directions;
Step 5	Calculation of acceleration and velocity
Step 6	Updating agents' positions
Step 7	Checking for the constraints of the problem
Step 8	Repeat Steps 2 to 7 until a stopping criterion is met

Figure 8. Steps of the GSA.

Table 1. Test problems characteristics.

Problem size $R\times P\times S\times M\times I\times N\times J\times L\times K$	Total E_{sr}	Total D_i	Total W_j	Total demand	Total $E_m^{(1)}$	Total $E_n^{(2)}$	Total $E_l^{(3)}$	Prob.* type	Range of variable costs	Range of fixed costs
$1\times1\times5\times2\times3\times2\times5\times2\times10$	3000	2000	3000	1000	1500	1500	1500	A	U(10,30)	U(100,500)
$1\times1\times10\times2\times5\times2\times10\times2\times20$	6000	4000	6000	2000	3000	3000	3000	B	U(10,30)	U(100,500)
$1\times1\times15\times2\times8\times2\times15\times2\times30$	8000	6000	8000	3000	4500	4500	4500	C	U(20,50)	U(100,500)
$2\times2\times20\times2\times10\times2\times20\times3\times40$	11000	9000	11000	4500	7000	7000	7000	D	U(20,50)	U(100,500)
$2\times2\times25\times2\times15\times3\times25\times3\times45$	14000	12000	14000	5000	8000	8000	8000		U(20,50)	U(100,500)
$2\times2\times30\times2\times50\times3\times30\times3\times50$	15000	13000	15000	7000	9500	9500	9500		U(30,60)	U(100,500)
$3\times2\times35\times3\times60\times3\times35\times4\times60$	18000	15000	18000	9000	11000	11000	11000		U(30,60)	U(100,500)
$3\times2\times40\times3\times70\times3\times45\times4\times75$	20000	18000	20000	11000	13000	13000	13000		U(30,80)	U(100,500)
$3\times2\times45\times3\times80\times4\times45\times4\times80$	22000	20000	22000	13000	15000	15000	15000		U(40,100)	U(100,500)
$3\times3\times50\times3\times100\times5\times50\times4\times100$	25000	23000	25000	15000	17500	17500	17500		U(40,100)	U(100,500)

*Prob: Problem

4.2. Parameter setting

In this subsection, the effects of different operators and parameters of the proposed algorithms are presented. To tune the operators and parameters of algorithms, there are different ways, of which one is the full factorial design approach. It will test all possible combinations of factors, but due to high cost and time is neither cost-effective nor applicable. As we will see later, for DE, there are 40 test problems, four 3-level factors, and one 3-level factor in our case, of which each should be run for ten times. Therefore, the total number of runs for the problem in DE is $40 \times 3^3 \times 10$, which is equal to 10,800. In the same manner, the total numbers of runs for the problems in PSO and GSA are equal to 874,800 and 32,400, respectively.

To decrease the number of experiments, several experimental design methods have been proposed. Among them, the Taguchi experimental design method has successfully been employed for the analysis of many different operators and parameters without all the combinations of the factors [33]. In Taguchi approach, the orthogonal arrays are used for classifying the results and analyzing a large number of variables with a small number of experiments [16].

Taguchi [33] has employed a transformation of the repetition data to another value, which is the measure of variation. The transformation is the Signal-to-Noise (S/N) ratio. The S/N ratio denotes the amount of variations present in the response variable. The aim is to maximize the signal-to-noise ratio. In the Taguchi method, the S/N ratio of the maximization objectives is as follows:

$$S/N\text{ratio} = -10\log_{10}(\text{objective function})^2.$$

Here, the Taguchi approach is used for fine-tuning the parameters to get better robustness of the proposed algorithms. The control factors of DE are as follows: *Popsize*, mutation constant (F), crossover constant (CR). The control factors of PSO are *Popsize*, minimum velocity ($v_{\min}$), maximum velocity ($v_{\max}$), maximum value of inertia weight ($w_{\max}$), minimum value of inertia weight ($w_{\min}$), and parameters (c_1) and (c_2). The control factors of the GSA are as follows: number of masses (N), initial Gravitational constant (G_0), constant ($\mathcal{E}$), and user-specified constant (α). Table 2 shows the operators and parameters of the proposed algorithms and their levels.

In order to solve this problem, the Relative Percentage Deviation (RPD) is applied for each instance. The RPD is obtained by the following formula:

$$RPD = \frac{\text{Alog}_{\text{sol}} - Min_{sol}}{Min_{\text{sol}}} \times 100,$$

where Alg_{sol} and Min_{sol} are the obtained objective value and minimum objective value found from the proposed algorithms for each instance, respectively. Thus, the RPD measure in the proposed algorithms is applied.

After obtaining the results of the test problems, they are transformed into RPD measures. The RPD measures are averaged and their values are shown in Figures 9-11.

4.3. Experimental results

We applied searching time as stopping criterion to be equal for all the algorithms which are equal to

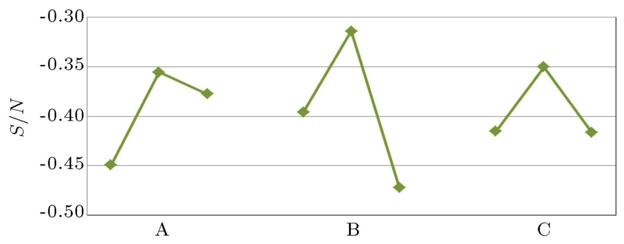

Figure 9. Mean S/N ratio plot for each level of the factors in DE.

Table 2. Factors and their levels.

DE factors	DE levels	PSO factors	PSO levels	GSA factors	GSA levels
Popsize	A(1)-80	*Popsize*	A(1)-50	N	A(1)-50
	A(2)-90		A(2)-55		A(2)-60
	A(3)-95		A(3)-60		A(3)-70
F	B(1)-0.45	$v_{\min}$	B(1)- (-2)	G0	B(1)-40
	B(2)-1		B(2)- (-1.5)		B(2)-60
	B(3)-1.5		B(3)- (-1)		B(3)-80
CR	C(1)-0.3	v_{max}	C(1)-2.5	$\mathcal{E}$	C(1)-0.005
	C(2)-0.4		C(2)-3		C(2)-0.007
	C(3)-0.5		C(3)-3.5		C(3)-0.009
		c_1	D(1)-0.3	α	D(1)-7
			D(2)-0.4		D(2)-12
			D(3)-0.5		D(3)-17
		c_2	E(1)-0.4		
			E(2)-0.5		
			E(3)-0.6		
		$w_{\max}$	F(1)-0.9		
			F(2)-0.85		
			F(3)-0.8		
		$w_{\min}$	G(1)-0.55		
			G(2)-0.5		
			G(3)-0.45		

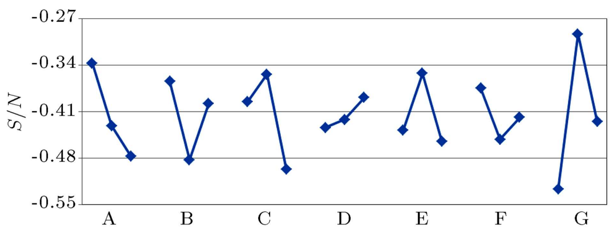

Figure 10. Mean S/N ratio plot for each level of the factors in PSO.

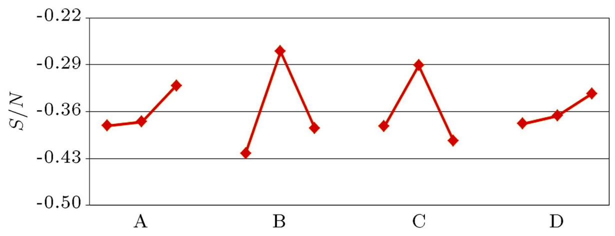

Figure 11. Mean RPD ratio plot for each level of the factors in GSA.

$0.6 \times (|Q_1| + |Q_2| + |Q_3|)$ milliseconds. Therefore, this criterion is affected by all problem characteristics.

In other words, any rise in the number of problem size directly increases the searching time. Forty instances for each of the ten problem sizes, i.e. totally 400 instances, are generated and are different from the ones used for calibration to avoid bias in the results. Each instance is run ten times. In each algorithm, there are forty instances considered for each of the ten problem sizes and the instances are run ten times. Therefore, we deal with 400 sets data for each algorithm by utilizing RPD. Since we had to appraise the robustness of the algorithms in different situations, the effects of the problem sizes on the performance of algorithms have been analyzed. The reciprocal relationship between the capability of the algorithms and the size of problems is illustrated in Figure 12. It shows the averages of the mentioned 400 sets data for each algorithm and each instance.

Based on the obtained results, we can conclude that the proposed GSA is effective to solve the problems. In order to verify the statistical validity of the results, we have performed an analysis of variance to accurately analyze the results. The point concluded

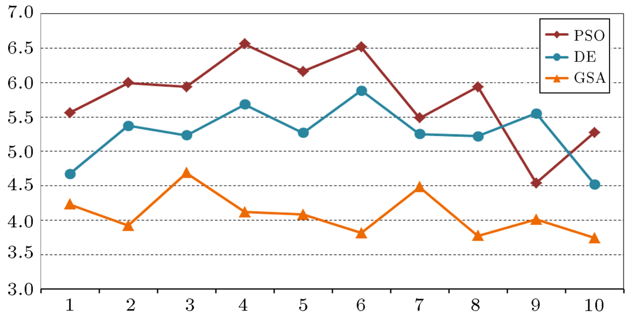

Figure 12. Means plot for the interaction between each algorithm and problem size.

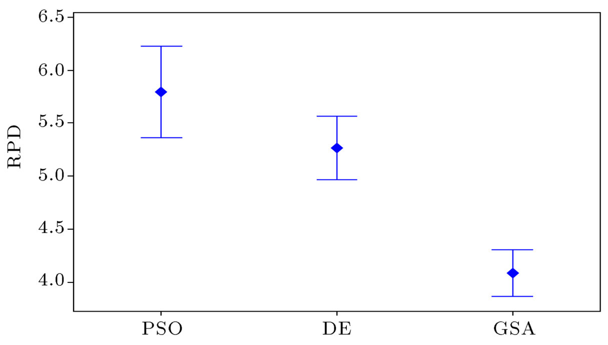

Figure 13. Means plot and LSD intervals for the algorithms.

from the results shows a clear statistically meaningful difference between the performances of GSA, DE, and PSO. The means plot and LSD intervals for all the algorithms are shown in Figure 13.

5. Conclusions and future research directions

In this paper, we proposed a mixed-integer programming model for the multi-stage, multi-product solid supply chain network design problem. The objective was minimization of the total cost of supply chain network. To solve this NP-hard problem, three meta-heuristic algorithms, namely differential evolution, particle swarm optimization, and gravitational search algorithm, were developed. Because of the dependency of the meta-heuristic algorithms on the proper selection of parameters, the experimental design approach was applied. The computational results demonstrate the convergence of GSA to solve the generated instances and its higher performance compared with differential evolution and particle swarm optimization algorithms in all problem sizes. For future research directions, new algorithms based on other meta-heuristic algorithms can be developed and compared with the proposed algorithms in this paper. Uncertainties in the model parameters and variables can be extended. Furthermore, for tuning the parameters of these algorithms, we can use the response surface methodology.

Acknowledgements

We are grateful to the editor and anonymous reviewers of the journal for their helpful suggestions which helped us to improve the quality of this paper.

References

1. Ko, M., Tiwari A. and Mehnen, J. "A review of soft computing applications in supply chain management", *Applied Soft Computing*, **10**, pp. 661-674 (2010).
2. Chen, C., Shih, H., Shyur H. and Wu, K. "A business strategy selection of green supply chain management via an analytic network process", *Computers & Mathematics with Applications*, **64**, pp. 2544-2557 (2012).
3. Mehdizadeha, E., Afrabandpeia, F., Mohaselafshar, S. and Afshar-Nadja, B. "Design of a multi-stage transportation network in a supply chain system: Formulation and efficient solution procedure", *Scientia Iranica*, **20**(6), pp. 2188-2200 (2013).
4. Wu, T. and Zhang, K. "A computational study for common network design in multi-commodity supply chains", *Computers & Operations Research*, **44**, pp. 206-213 (2014).
5. Altiparmak, F., Gen, M., Lin, L. and Karaoglan, I. "A steady-state genetic algorithm for multi-product supply chain network design", *Computers & Industrial Engineering*, **56**(2), pp. 521-537 (2009).
6. Jayaraman, V. and Pirkul, H. "Planning and coordination of production and distribution facilities for multiple commodities", *European Journal of Operational Research*, **133**, pp. 394-408 (2001).
7. Syam, S.S. "A model and methodologies for the location problem with logistical components", *Computers & Operations Research*, **29**, pp. 1173-1193 (2002).
8. Mehdizadeh, E. and Afrabandpei, F. "Design of a mathematical model for logistic network in a multi-stage multi-product supply chain network and developing a metaheuristic algorithm", *Journal of Optimization in Industrial Engineering*, **10**, pp. 35-43 (2012).
9. Kadadevaramath, R.S., Chen, J.C., Shankar, B.L. and Rameshkumar, K. "Application of particle swarm intelligence algorithms in supply chain network architecture optimization", *Expert Systems with Applications*, **39**, pp. 10160-10176 (2012).
10. Olivares-Benitez, E., Rıos-Mercado, R.Z. and Gonzaĺez -Velarde, J.L. "A metaheuristic algorithm to solve the selection of transportation channels in supply chain design", *International Journal of Production Economics*, **145**, pp. 161-172 (2013).
11. Cardenas-Barron, L.E. and Trevino-Garza, G. "An optimal solution to a three echelon supply chain network with multi-product and multi-period", *Applied Mathematical Modeling*, **38**, pp. 1911-1918 (2014).
12. Kristianto, Y., Gunasekaran, A., Helo, P. and Hao, Y. "A model of resilient supply chain network design: A two-stage programming with fuzzy shortest path", *Expert Systems with Applications*, **41**, pp. 39-49 (2014).

13. Khalifehzadeh, S., Seifbarghy, M. and Naderi, B.A. "four-echelon supply chain network design with shortage: Mathematical modeling and solution methods", *Journal of Manufacturing Systems*, **35**, pp. 164-175 (2015).

14. Cardona-Valdés, Y., Alvarez, A. and Pacheco, J. "Metaheuristic procedure for a bi-objective supply chain design problem with uncertainty", *Transportation Research Part B*, **60**, pp. 66-84 (2014).

15. Lotfi, M.M. and Tavakkoli-Moghaddam, R.A. "Genetic algorithm using priority-based encoding with new operators for fixed charge transportation problems", *Applied Soft Computing*, **13**, pp. 2711-2726 (2013).

16. Molla-Alizadeh-Zavardehi, S., Sadi Nezhad, S., Tavakkoli-Moghaddam, R. and Yazdani, M. "Solving a fuzzy fixed charge solid transportation problem by metaheuristics", *Mathematical and Computer Modelling*, **57**, pp. 1543-1558 (2013).

17. Gen, M. and Cheng, R., *Genetic Algorithms and Engineering Optimization*, Second Edn., John Wiley & Sons, New York (2000).

18. Lee, J.E., Gen, M. and Rhee, K.G. "Network model and optimization of reverse logistics by hybrid genetic algorithm", *Computers & Industrial Engineering*, **56**, pp. 951-964 (2009).

19. Lin, L., Gen, M. and Wang, X. "Integrated multi-stage logistics network design by using hybrid evolutionary algorithm", *Computers & Industrial Engineering*, **56**, pp. 854-873 (2009).

20. Gen, M., Altiparmak F. and Lin, L. "A genetic algorithm for two-stage transportation problem using priority-based encoding", *OR Spectrum*, **28**, pp. 337-354 (2006).

21. Storn R. and Price, K. "Differential evolution- a simple and efficient adaptive scheme for global optimization over continuous spaces", *Journal of Global Optimization*, **11**, pp. 341-359 (1997).

22. Mahmoodi-Rad, A., Molla-Alizadeh-Zavardehi, S., Dehghan, R., Sanei, M. and Niroomand, S. "Genetic and differential evolution algorithms for the allocation of customers to potential distribution centers in a fuzzy environment", *International Journal of Advanced Manufacturing Technology*, **70**, pp. 1939-1954 (2014).

23. Mozdgir, A., Fatemi Ghomi, S.M.T., Jolai, F. and Navaei, J. "Three meta-heuristics to solve the no-wait two-stage assembly flow-shop scheduling problem", *Scientia Iranica*, **20**(6), pp. 2275-2283 (2013).

24. Shaw, B., Mukherjee, V. and Ghoshal, S.P. "Solution of reactive power dispatch of power systems by an opposition-based gravitational search algorithm", *Electrical Power and Energy Systems*, **55**, pp. 29-40 (2014).

25. Tsai, J.T. "Improved differential evolution algorithm for nonlinear programming and engineering design problems", *Neurocomputing*, **148**, pp. 628-640 (2015).

26. Kennedy, J. and Eberhart, R. "Particle swarm optimization", In: *IEEE International Conference on Neural Networks*, pp. 1942-1948 (1995).

27. Zeighami, V. Akbari, R. and Ziarati, K. "Development of a method based on particle swarm optimization to solve resource constrained project scheduling problem", *Scientia Iranica*, **20**(6), pp. 2123-2137 (2013).

28. Rashedi, E., Nezamabadi-pour H. and Saryazdi, S. "GSA: a gravitational search algorithm", *Information Sciences*, **179**(13), pp. 2232-2248 (2009).

29. David, R.C., Precup, R.E., Petriu, E.M., Radac, M.B. and Preitl, S. "Gravitational search algorithm-based design of fuzzy control systems with a reduced parametric sensitivity", *Information Sciences*, **247**, pp. 154-173 (2013).

30. Davarynejad, M., Berg J. and Rezaei, J. "Evaluating center-seeking and initialization bias: The case of particle swarm and gravitational search algorithms", *Information Sciences*, **278**(10), pp. 802-821 (2014).

31. Gao, S., Vairappan, C., Wang, Y., Cao, G. and Tang, Z. "Gravitational search algorithm combined with chaos for unconstrained numerical optimization", *Applied Mathematics and Computation*, **231**, pp. 48-62 (2014).

32. Han, X.H., Quan, L., Xiong, X.Y. and Wu, B. "Diversity enhanced and local search accelerated gravitational search algorithm for data fitting with B-splines", *Engineering with Computers*, **31**, pp. 215-236 (2015).

33. Taguchi, G., *Introduction to Quality Engineering*, Asian Productivity Organization/UNIPUB, White Plains (1986).

Biographies

Ali Mahmoodirad is a PhD candidate of Applied Mathematics (Operation Research) at Islamic Azad University, Central Tehran Branch, Tehran, Iran. His research interests include meta-heuristics and hybrid methods, supply chain management, and fuzzy mathematical programming. He has published research articles in international journals of mathematics and industrial engineering.

Masoud Sanei is an Associate Professor in the Department of Applied Mathematics, Islamic Azad University, Central Tehran Branch, Tehran, Iran. He received his PhD degree in Applied Mathematics (Operations Research) from Islamic Azad University, Science and Research Branch, Tehran, Iran, in 2004. His research interests are in the areas of operation research such as data envelopment analysis and meta-heuristic algorithms. He has several papers in journals and conference proceedings.