Kybernetika

Nermin Kartli; Erkan Bostanci; Mehmet Serdar Guzel

A new algorithm for optimal solution of fixed charge transportation problem

Kybernetika, Vol. 59 (2023), No. 1, 45–63

Persistent URL: http://dml.cz/dmlcz/151583

KYBERNETIKA — VOLUME *59* (2023), NUMBER 1, PAGES 45–63

A NEW ALGORITHM FOR OPTIMAL SOLUTION OF FIXED CHARGE TRANSPORTATION PROBLEM

Nermin Kartli, Erkan Bostanci and Mehmet Serdar Guzel

Fixed charge transportation problem (FCTP) is a supply chain problem. In this problem, in addition to the cost per unit for each transported product, a fixed cost is also required. The aim is to carry out the transportation process at the lowest possible cost. As with all supply chain problems, this problem may have one, two, or three stages. An algorithm that can find the optimal solution for the problem in polynomial time is not known, even if it is a single-stage problem. For this reason, new algorithms have been proposed in recent years to provide an approximate solution for the problem. The vast majority of these algorithms are meta-heuristic algorithms. In this study, we propose a new heuristic algorithm to find an optimal solution for the 1-stage FCTP. We compare the results of our algorithm with the results of other existing algorithms.

Keywords: supply chain, transportation problem, fixed charge transportation problem, feasible solution, optimal solution

Classification: 90C08, 90B06, 90C59, 90C10

1. INTRODUCTION

The supply chain planning problem is one of the important research problems. The purpose of the mathematical models proposed for supply chain planning is to set decision-support systems that distribute strategic resources efficiently in the long term. There are three types of models in supply chain planning: the production-distribution model [7], the location-allocation and routing model [8], and the inventory-transport model [6]. In the production-distribution model, production and scheduling decisions are taken and distribution is planned. In the location-allocation and routing model, the convenient location of the facilities and warehouses is investigated, and the most suitable transportation route is planned for the customers. In the inventory-transport model, the inventory control of the warehouses is carried out, and the transportation of the products from the production facilities to the wholesalers, retailers, and customers is planned. Supply chain models can be constructed as one stage, two stages, or three stages. One-stage model is the Plant-Distributor (PD) model. Two-stage models are Plant-Distributor-Retailer (PDR) model and Plant-Distributor-Consumer (PDC) model.

DOI: 10.14736/kyb-2023-1-0045

Three-stage supply chains are Plant-Distributor-Retailer-Consumer (PDRC), Supplier-Plant-Distributor-Retailer (SPDR) and Supplier-Plant-Distributor-Consumer (SPDC) models [25, 26].

Economic models involving an optimization problem can be divided into 2 groups: models with and without uncertainty. Models including uncertainty can be solved through probability theory, interval analysis, and fuzzy logic. If there are random variables in the problem formulation, it is appropriate to model and solve the problem through probability theory. If there are no random variables in the problem formulation, but there exists uncertainty, it is more reasonable to solve the problem using the interval analysis method or to model the problem with fuzzy sets. If the values of certain variables in the problem formulation are not known, but it is known that the variables take the values in an interval with the same possibility, then the interval analysis method is used. If certain values of the problem are unknown and the variables take the values in an interval with different possibilities, then fuzzy logic and fuzzy sets are used. A problem can include all three kinds of variables and in that case, hybrid approaches are used.

Different problems related to supply chain problem models are investigated in the academic literature. Many of these problems are listed below.

- TP (Transportation Problem)
- FCTP (Fixed Charge Transportation Problem)
- CFCTP (Capacitated Fixed Charge Transportation Problem)
- SFCTP (Step Fixed Charge Transportation Problem)
- NLFCTP (Non-linear Fixed Charge Transportation Problem)
- FCSTP (Fixed Charge Solid Transportation Problem)
- FFCSTP (Fuzzy Fixed Charge Transportation Problem)
- SSFCTP (Single Sink Fixed Charge Transportation Problem)

Classical transportation problem defined by Hitchcock [15] for the first time is actually a one-stage supply chain problem. Hitchcock [15] defines the problem as follows: a product that is produced in more than one factory will be sent to different cities. The cost of transporting products from different suppliers to a city and from a supplier to different cities is different. The purpose is to form a distribution plan in which the products are distributed at the lowest cost. Each supplier has a production capacity, and each distributor has a storage capacity, in classical transportation problems. There is a transportation cost for shipping a product from each supplier to each distributor and this cost is directly proportional to the quantity of the transported product. In real life, this issue may be different. Most transportation companies require a fixed cost regardless of the number of products transported. Fixed Charge Transportation Problem (FCTP) is defined by Hirsch and Dantzig [16] for the first time and under certain circumstances it is transformed into a classical linear programming problem. [9] is the first study in which the FCTP is examined as a linear integer programming problem. In this study,

Balinski studied two classical transportation problems instead of FCTP and found upper and lower limits for the optimal solution of the main problem. He proposed a matrix called the Balinski matrix in order to find the lower limit of the problem and solved the TP with real-valued conditions. To find an upper limit, he formed a feasible solution based on the optimal solution obtained for the lower limit. He defined a new classical TP using this feasible solution and solved the problem. We explain this problem in detail in Section 3. Actually, in order to find an approximate solution for both one-stage and two-stages FCTP, most of the algorithms proposed in the literature are either based on modifying the Balinski matrix to increase the lower value founded or modifying various meta-heuristic algorithms or directly using them in order to decrease the upper value.

The main idea behind the algorithms proposed to decrease the upper value will also be explained in Section 3. After the work of Balinski [9], this problem is studied by Gottlieb and Paulmann [14], Sun et al. [31], Adlakha and Kowalski [1], Adlakha and Kowalski [2], Adlakha et al. ([3]-[5]), Jo et al [20], Raj and Rajendran [28, 29], Panicker et al. [25], Lotfi and Tavakkoli-Moghaddam [24], El-Sherbiny and Alhamali [12], Eskandarpour et al. [13], Pop et al. [27], Tari and Hashemi [32], Calvete et al. [10], Hong et al. [17], Cosma et al. [11], Singh and Singh [30]. Sun et al. [31] solve the FCTP for the plant-distributor (PD) model with Tabu Search (TS) algorithm. Gottlieb and Paulmann [14] use a genetic algorithm for the same problem, while Adlakha and Kowalski [2] propose a new heuristic algorithm. Jo et al. [20] solve the NLFCTP (Non-linear Fixed Charge Transportation Problem) for the PD model with genetic algorithms. Jawahar and Balaji [18] investigate FCTP for the plant-distributor-retailer (PDR) model by using a genetic algorithm. Jawahar et al. [19] propose a Simulated Annealing algorithm to solve FCTP for the PD model. Panicker et al. [25] solve the FCTP for the PDR model using the ant colony optimization algorithm. Hong et al. [17] also solve the same problem with the ant colony optimization method. This study differs from [25] as the model enables a new distributor to be added to the examined problem. Panicker et al. [26] use a genetic algorithm to solve FCTP for the plant-distributor-retailer-customer (PDRC) model. Eskandarpour et al. [13] propose a heuristic algorithm Large Neighborhood Search to solve FCTP for the TTDM model. Pop et al. [27] solve FCTP (Fixed Charge Transportation Problem) for the plant-distributor-customer (PDC) model using a genetic algorithm. Cosma et al. [11] propose a new algorithm based on a genetic algorithm to solve FCTP for the PDR model.

In all the studies, described above, researchers either proposed heuristic algorithms for obtaining better results following the method of Balinski or applied various meta-heuristic algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO) algorithm, or modified versions of them. Generally, the way followed is to transform the problem into a classical transportation problem.

In this study, we also examine FCTP. However, we follow a different way such that we try to minimize the value of objective function starting from an initial solution without transforming FCTP to a classical TP. We use a large number of initial solutions (i. e. use a large search space) to avoid getting stuck with the local minimum. As a result, in this study, we propose a new heuristic algorithm to approximate the optimal solution of FCTP. The results of the experiments show that the proposed algorithm is efficient and successful. The proposed algorithm can also be applied to find an approximate solution

for large-scale FCTP.

2. ONE-STAGE FCTP

There are m suppliers and n distributors. Shipping cost per unit product from ith supplier to jth distributor is c_{ij}. Also if at least one product is transferred to jth distributor from the ith supplier, then a constant cost a_{ij} is required. Capacity of ith supplier is s_i and capacity of jth distributor is d_j. All given numbers are positive integers and the balance condition defined below holds.

We can write the problem as below:

$$F(X,Y) = \sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij}x_{ij} + a_{ij}y_{ij}) \to \min \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = s_i \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} = d_j \tag{3}$$

$$\sum_{j=1}^{n} d_j = \sum_{i=1}^{m} s_i. \tag{4}$$

For all i, j

$$x_{ij} \geq 0, \tag{5}$$

$$y_{ij} = \begin{cases} 1, & \text{if} \quad x_{ij} > 0 \\ 0, & \text{if} \quad x_{ij} = 0. \end{cases} \tag{6}$$

The condition (4) is called a balance condition and for all i, j the given values c_{ij}, a_{ij}, s_i, d_j are positive integers. The aim is to find non-negative integers x_{ij} and y_{ij} which minimize the objective function (1) under the conditions (2–6). The $m \times n$ sized matrices $X = (x_{ij})$ and $Y = (y_{ij})$ are called a feasible solution if they satisfy the conditions (2–6). A pair (X^0, Y^0) of $m \times n$ matrices which minimizes the objective function (1) among all feasible solutions is called an optimal solution.

3. PRELIMINARIES

Even though FCTP is first defined by Hirsch and Dantzig [16], the problem is first investigated by Balinski [9] as an integer programming problem. In general, integer programming problems are known to be more difficult than problems with real numbers. These problems can arise not only in transportation but also in other areas [22, 23]. Balinski proved many important theorems in his study. Let us remember some of them:

Theorem 1. (Balinski [9]) Problem (1–6) has the optimal solution (X^0, Y^0) in the set of integer numbers.

Let the pair (X^0, Y^0) be the optimal solution in the set of integer numbers for the problem (1–6).

For all i, j, define $\tilde{c}_{ij}$ as:

$$\tilde{c}_{ij} = \begin{cases} c_{ij}, & \text{if} \quad y_{ij}^0 = 1 \\ c_{ij} + a_{ij}, & \text{if} \quad y_{ij}^0 = 0\,. \end{cases} \tag{7}$$

Then, the objective function is defined as:

$$\tilde{F}(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} (\tilde{c}_{ij} x_{ij}) \to \min\,. \tag{1*}$$

Theorem 2. (Balinski [9]) Let (X^0, Y^0) be the optimal solution of problem (1–6) in the set of integer numbers. Then, X^0 is the optimal solution of the problem (1*)-(2–5).

Note that problem (1*)-(2–5) is a classical transportation problem.

Now, let us define $m_{ij} = \min\{s_i, d_j\}$ for all i, j. Then the condition

$$0 \le x_{ij} \le m_{ij} y_{ij} \tag{5*}$$

will hold for all feasible solutions $X = (x_{ij})$ of the problem (1–6).

Theorem 3. (Balinski [9]) If $(\bar{X} = (\bar{x}_{ij}),\ \bar{Y} = (\bar{y}_{ij}))$ is the optimal solution of the problem (1–4) – (5*) in the real numbers set, then for all i, j the equality $\bar{x}_{ij} = m_{ij}\bar{y}_{ij}$ is satisfied.

Theorem 4. (Balinski [9]) Let (X^0, Y^0) be the optimal solution of problem (1–6) in the set of integer numbers. Then, at most, $m + n - 1$ number of elements of the matrix X^0 are positive integers.

In order to set the lowest bound for the optimal value of the objective function of the problem (1–6), Balinski puts $y_{ij} = \frac{x_{ij}}{m_{ij}}$ instead of y_{ij} in the objective function using Theorem 3. Then, the objective function becomes:

$$F(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(c_{ij} + \frac{a_{ij}}{m_{ij}} \right) x_{ij} \to \min\,. \tag{1**}$$

Afterward, he searches for the minimum value of the function (1**) holding conditions (2-5) in the set of integers. This value will not exceed the optimal value of the problem (1–6). In the literature, matrix $B = (b_{ij}) = (c_{ij} + \frac{a_{ij}}{m_{ij}})$ is called Balinski matrix.

Let $\bar{X} = (\bar{x}_{ij})$ be the optimal solution of problem (1**)-(2–5) in the set of the integer numbers. Balinski forms the feasible solution $(X^* = (x_{ij}^*), Y = (y_{ij}^*))$ of the problem (1–6) based on this optimal solution as: For all i, j; if $\bar{x}_{ij} = 0$, then $x_{ij}^* = y_{ij}^* = 0$; else if $\bar{x}_{ij} > 0$, then $x_{ij}^* = \bar{x}_{ij},\ \ y_{ij}^* = 1$.

After that, he finds the optimal value of the problem (1*)-(2–5) by setting values of $\tilde{c}_{ij}$ based on values of y^*_{ij} in the formula (7). This value will not be lesser than the optimal value of the problem (1–6).

As we stated before, most of the algorithms proposed for finding an approximate solution of the FCTP, are either based on modifying the Balinski matrix for increasing the lowest value or modifying the meta-heuristic algorithms or directly using them to decrease the highest value Balinski finds.

The main idea behind the algorithms to decrease the highest value Balinski finds is simple: randomly choose a matrix Y, then form the values of formula (7) based on this matrix and find the optimal solution of the problem (1*)-(2–5). If this solution holds the condition (6), then stop; otherwise, update the matrix Y and continue. It is natural that there is no guarantee for the condition (6) to be held, even if it holds, it is possible that the solution obtained will not be the optimal solution to the problem (1–6). Moreover, based on just a matrix Y, the probability of getting a good result is low. Therefore, a specific number of (population) matrices Y are taken in the proposed algorithms, and some meta-heuristic algorithms like genetic algorithms are applied to crossover matrices Y.

4. THE PROPOSED ALGORITHM TO FIND AN APPROXIMATE OPTIMAL SOLUTION

In this section, we propose a new algorithm to find an approximate optimal solution for the FCTP. Let us explain the main idea behind the algorithm.

First of all, assume that we have an initial solution X. Pick up any 2×2 sub-matrix SQ of this solution. In the general case, this sub-matrix can be defined as follows:

$$SQ = \begin{bmatrix} x_{i1,j1} & x_{i1,j2} \\ x_{i2,j1} & x_{i2,j2} \end{bmatrix}.$$

Let us take $i_1 = 1, i_2 = 2, j_1 = 1, j_2 = 2$ for the simplicity of the explanation.

That is, let SQ be:

$$SQ = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix}.$$

Let us define:

$$\tilde{s}_1 = x_{11} + x_{12}$$
$$\tilde{s}_2 = x_{21} + x_{22}$$
$$\tilde{d}_1 = x_{11} + x_{21}$$
$$\tilde{d}_2 = x_{12} + x_{22}.$$

Consider the FCTP defined below:

$$c_{11}\tilde{x}_{11} + c_{12}\tilde{x}_{12} + c_{21}\tilde{x}_{21} + c_{22}\tilde{x}_{22} + a_{11}\tilde{y}_{11} + a_{12}\tilde{y}_{12} + a_{21}\tilde{y}_{21} + a_{22}\tilde{y}_{22} \to \min$$

$$\tilde{x}_{11} + \tilde{x}_{12} = \tilde{s}_1$$

$$\tilde{x}_{21} + \tilde{x}_{22} = \tilde{s}_2$$

$$\tilde{x}_{11} + \tilde{x}_{21} = \tilde{d}_1$$

$$\tilde{x}_{12} + \tilde{x}_{22} = \tilde{d}_2,$$

where for all $i = 1, 2$ and $j = 1, 2$ the conditions $\tilde{x}_{ij} \geq 0$ are satisfied. Moreover, if $\tilde{x}_{ij} > 0$, then $\tilde{y}_{ij} = 1$; otherwise, $\tilde{y}_{ij} = 0$. Since $\tilde{s}_1 + \tilde{s}_2 = \tilde{d}_1 + \tilde{d}_2$, this problem is a balanced problem. The matrix SQ is a feasible solution to this problem. If all the elements of the matrix SQ are positive, the matrix SQ cannot be the optimal solution to the problem according to Theorem 4. Assume that at least one element of matrix SQ is zero. For example, let $x_{11} = 0$. If $x_{12} = 0$ or $x_{21} = 0$, then the matrix SQ is the unique feasible solution, and therefore it is the optimal solution to the problem.

Now, let $x_{12}x_{21} > 0$. If $x_{22} > 0$, then, no other solution can be obtained by decreasing this value, because when this value is decreased, x_{12} must be increased, but this is not possible according to the condition $x_{11} = 0$. So it can be possible to obtain a better solution by decreasing x_{21} or x_{12}. For example, assume that the minimum of these two values is x_{12}. Now, search for the new solution by:

$$\tilde{X} = \begin{bmatrix} \tilde{s}_1 - x & x \\ \tilde{d}_1 + x - \tilde{s}_1 & \tilde{d}_2 - x \end{bmatrix}.$$

Here, $\tilde{s}_1 = x_{12}$ and $\tilde{d}_1 = x_{21}$. Condition $0 \leq x \leq \tilde{s}_1 = x_{12}$ holds for x. If $x \neq 0$ or $x \neq x_{12}$, then all elements of the matrix $\tilde{X}$ will be positive, consequently this solution cannot be the optimal solution by the Theorem 4. So in order to be an optimal solution, either $x = 0$ or $x = x_{12}$ should hold. The case of $x = x_{12}$ is the same as the initial case. The case of $x = 0$ means that the minimum element of the diagonal (both elements of which are positive) is subtracted from the elements of this diagonal and added to the elements of the other diagonal. That is, in order to find the optimal solution, it is sufficient to calculate the value of the objective function for these two cases and choose the smallest one.

The proposed algorithm has 2 parameters h and k. The parameter h shows the maximum number of times the objective function is allowed not to decrease at each minimization stage. The parameter k is equal to the number of the initial feasible solutions. In the algorithm, we define a variable kf with an initial value equal to 0. We also define the variables $fbest$ and $fprebest$ with initial values $+\infty$ (In fact, the largest number that can be used on a computer) and repeat the steps below as long as $kf < h$. In these steps for each $1 \leq i \leq k$, we find an initial feasible solution $X(i)$ and calculate the value of the objective function $f(i)$ at $X(i)$. Then, for all $X(i)$, we provide the minimization stage in which the minimization procedure returns the new values $f(i)$ and $X(i)$. After this step, we find the minimum value of $f(i)$ and the index $i\_\min$ for which this minimum value is attained. We assign the value $f(i\_\min)$ to the variable $fbest$. If this value is lower than the value of $fprebest$, then we assign $fprebest = fbest$ and store the solution $X(i\_\min)$ as the best solution XC; otherwise, we increase the value of the variable kf by one. The steps of the algorithm are explained below. These steps are shown in Figure 1 as a flowchart.

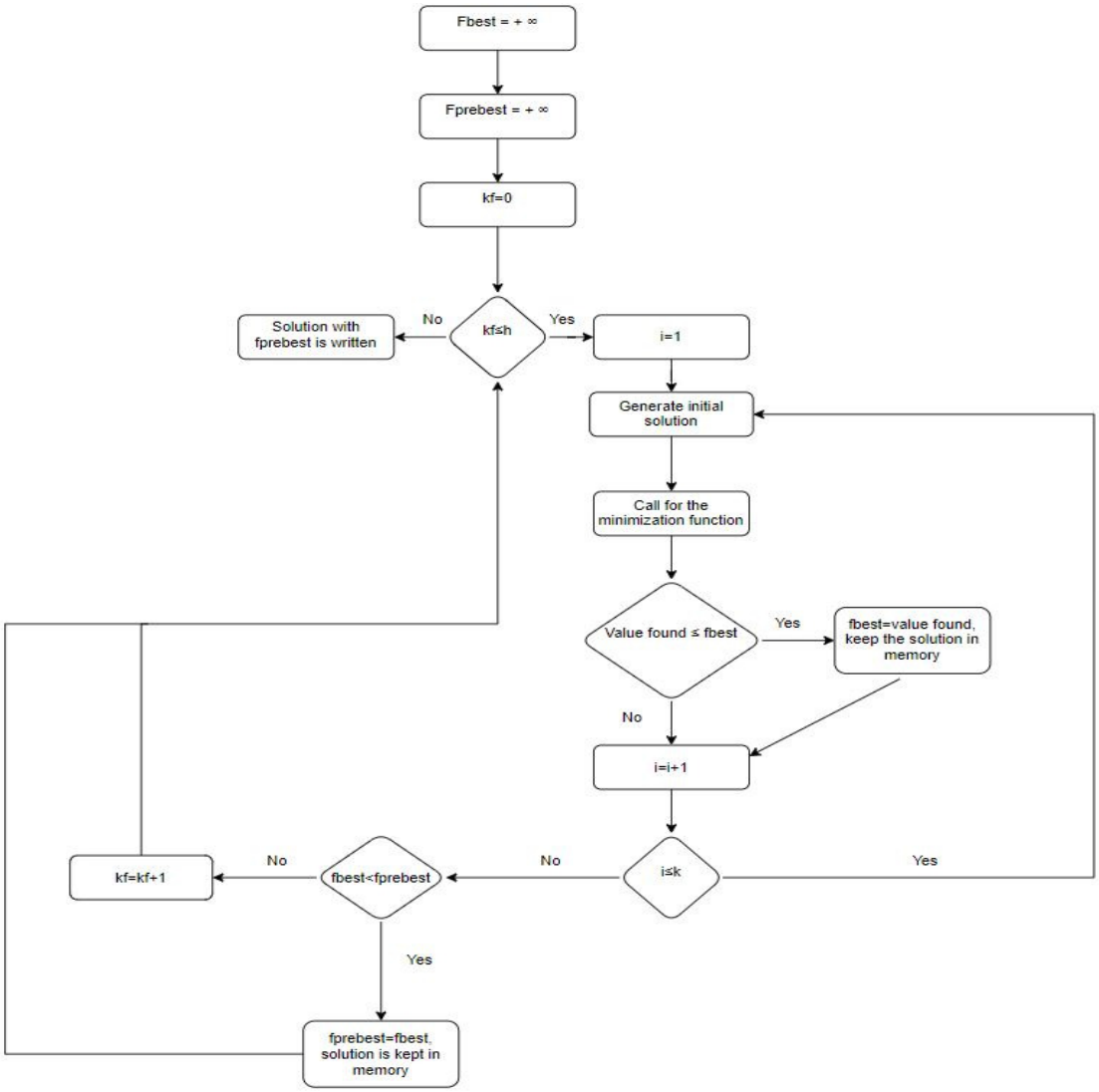

Fig. 1. Flowchart of the algorithm.

Step 1. Define the number k. (It shows the number of initial solutions to be used in each stage.)

Step 2. Define the number h. (This number shows the maximum number of times the objective function is allowed not to decrease.)

Step 3. Define $kf = 0$, $fbest = \infty$, $fprebest = \infty$.

Step 4. While $kf < h$, repeat Steps 5–10.

Step 5. For each $1 \leq i \leq k$, find the initial solution $X(i)$.

Step 6. For each $X(i)$, calculate $f(i) = F(X(i))$

Step 7. Call for minimization procedure for all $X(i)$. (The minimization procedure finds a new solution $X(i)$ and $f(i)$ for each i.)

Step 8. Find the index $i_{\min}$ for which $f(i)$ takes the minimum value.

Step 9. $fbest = f(i_{\min})$

Step 10. If $fbest < fprebest$, then $fprebest = fbest$, $XC = X(i_{\min})$, otherwise $kf = kf + 1$

Let us now explain the minimization stage. In this stage, we define a variable df with an initial value equal to -1. We assign the value of the objective function $f(X)$ for the initial feasible solution X to the variable $f1$. In order to find the initial feasible solution, we apply the algorithm proposed by us recently [21]. Then, we begin a loop while $df < 0$. At the beginning of this loop, we assign $df = 0$. Then, we search for each element of the X and find nonzero elements. We keep row and column numbers of the nonzero elements in the vectors $Irow$ and $Jcol$, respectively. According to Theorem 4, dimensions of these vectors cannot exceed $m + n - 1$. Then we look at all 2×2 sub-matrices where the product of the elements of a diagonal is nonzero. We find the minimum of the elements of this diagonal, subtract this minimum from the elements of this diagonal, add it to the elements of the other diagonal, and write the result to a 2×2 dimensional matrix SQ. We calculate the difference $df1$ of the objective function for SQ and for the considered 2×2 sub-matrix of X. If $df1 < df$ then we put $df = df1$. In this case, we store the matrix SQ as SQM, the row and column numbers of the considered 2×2 sub-matrix of X. After looking at all possible 2×2 sub-matrices of X, if $df < 0$, the elements of the stored 2×2 sub-matrix are replaced with the appropriate elements of the matrix SQM. We also change the value of $f1$ with $f1 + df$ and repeat the iterations for the new matrix X while $df < 0$. After finishing the while loop, we return the value $f1$ and X.

Pseudo-code for the minimization procedure is given below in Algorithm 1.

Remark. It is possible that the optimal solution cannot be obtained for an initial solution after the minimization process. We can show this case in Balinski's example.

$$X = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 0 & 30 \\ 10 & 30 & 0 \\ 0 & 20 & 0 \end{bmatrix}.$$

The value of the objective function for this initial solution is 360. According to the proposed algorithm, we keep indices of nonzero elements of X in vectors $Irow$ and $Jcol$ in order to find 2×2 sub-matrices.

Algorithm 1: Minimization procedure

Input: $m \times n$ matrix X, which is initial feasible solution.
Output: $m \times n$ matrix X, which is approximate optimal solution and $f1$, the value of the objective function.

procedure Min(X)
$f1 \leftarrow F(X)$; $df \leftarrow -1$
while $df < 0$ **do**
  $df \leftarrow 0$; $kk \leftarrow 0$
  for $i \leftarrow 1$ *to* m **do**
    for $j \leftarrow 1$ *to* n **do**
      if $X(i,j) > 0$ **then**
        $kk \leftarrow kk + 1$; $Irow(kk) \leftarrow i$; $Jcol(kk) \leftarrow j$
      end
    end
  end
  for $ik \leftarrow 1$ *to* $kk - 1$ **do**
    $i1 \leftarrow Irow(ik)$; $j1 \leftarrow Jcol(ik)$
    for $jk \leftarrow ik + 1$ *to* kk **do**
      $i2 \leftarrow Irow(jk)$; $j2 \leftarrow Jcol(jk)$
      if $i1 \neq i2$ *and* $j1 \neq j2$ **then**
        $min \leftarrow X(i1, j1)$
        if $min > X(i2, j2)$ **then**
          $min \leftarrow X(i2, j2)$
        end
        $SQ(1,1) \leftarrow X(i1, j1) - min$; $SQ(1,2) \leftarrow X(i1, j2) + min$
        $SQ(2,1) \leftarrow X(i2, j1) + min$; $SQ(2,2) \leftarrow X(i2, j2) - min$
        $f2 \leftarrow C(i1, j1) * X(i1, j1) + C(i1, j2) * X(i1, j2) + C(i2, j1) * X(i2, j1) +$
        $C(i2, j2) * X(i2, j2) + A(i1, j1) + A(i2, j2)$
        if $X(i1, j2) > 0$ **then**
          $f2 \leftarrow f2 + A(i1, j2)$
        end
        if $X(i2, j1) > 0$ **then**
          $f2 \leftarrow f2 + A(i2, j1)$
        end
        $f3 \leftarrow C(i1, j1) * SQ(1,1) + C(i1, j2) * SQ(1,2) + C(i2, j1) * SQ(2,1) +$
        $C(i2, j2) * SQ(2,2) + A(i1, j2) + A(i2, j1)$
        if $SQ(1,1) > 0$ **then**
          $f3 \leftarrow f3 + A(i1, j1)$
        end
        if $SQ(2,2) > 0$ **then**
          $f3 \leftarrow f3 + A(i2, j2)$
        end
        $df1 \leftarrow f3 - f2$
        if $df1 < df$ **then**
          $df \leftarrow df1$; $f \leftarrow f1 + df$
          $i1m \leftarrow i1$; $j1m \leftarrow j1$
          $i2m \leftarrow i2$; $j2m \leftarrow j2$
          $SQM(1,1) \leftarrow SQ(1,1)$; $SQM(1,2) \leftarrow SQ(1,2)$
          $SQM(2,1) \leftarrow SQ(2,1)$; $SQM(2,2) \leftarrow SQ(2,2)$
        end
      end
    end
  end
  if $df < 0$ **then**
    $X(i1m, j1m) \leftarrow SQM(1,1)$; $X(i1m, j2m) \leftarrow SQM(1,2)$
    $X(i2m, j1m) \leftarrow SQM(2,1)$; $X(i2m, j2m) \leftarrow SQM(2,2)$
    $f1 \leftarrow f$
  end
end
return X *and* $f1$

	$Irow$	$Jcol$
$kk = 1$	1	1
$kk = 2$	2	3
$kk = 3$	3	1
$kk = 4$	3	2
$kk = 5$	4	2

Tab. 1. Indices of nonzero elements of X.

For, $i_1 = 1$ and $j_1 = 1$; positive cells are: (2,3), (3,1), (3,2), and (4,2). Here, (i, j) indicates the intersection of $i-$th row and $j-$th column. Then we have to consider 3 sub-matrices (Tables 2-4). The sub-matrix corresponding to the (3,1) is not considered since it is in the same column as (1,1).

	$j_1 = 1$	$j_2 = 3$
$i_1 = 1$	10	0
$i_2 = 2$	0	30

Tab. 2. The first sub-matrix considered for the cell (1,1).

	$j_1 = 1$	$j_2 = 2$
$i_1 = 1$	10	0
$i_2 = 3$	0	30

Tab. 3. The second sub-matrix considered for the cell (1,1).

	$j_1 = 1$	$j_2 = 2$
$i_1 = 1$	10	0
$i_2 = 4$	0	20

Tab. 4. The third sub-matrix considered for the cell (1,1).

For example, for the first case (Table 2) we have the sub-matrix $\begin{bmatrix} 10 & 0 \\ 0 & 30 \end{bmatrix}$

In order to create sub-matrix SQ, we subtract 10 from positive cells and add 10 to the other cells, since the minimum number in the diagonal whose elements are positive is 10.

Sub matrix SQ obtained is:

$$SQ = \begin{bmatrix} 0 & 10 \\ 10 & 20 \end{bmatrix}.$$

Now, let us calculate the value of the objective function for the examined 2×2 sub-matrix and sub-matrix SQ.

The first value is $c_{11}x_{11} + a_{11} + c_{23}x_{23} + a_{23} = 2 \times 10 + 10 + 1 \times 30 + 20 = 80$.

The value for the matrix SQ is:

$c_{13}x_{12}+c_{21}sq_{21}+c_{23}sq_{22}+a_{13}+a_{21}+a_{23} = 4\times10+3\times10+1\times20+20+10+20 = 140$.

Since there is no decreasing, we do not store this 2×2 sub-matrix and the matrix SQ in memory.

Similarly, we can check that the value of the objective function does not decrease for other cases too.

Therefore, the result of the minimization process will remain 360 for this initial solution X.

However, the value of the objective function will be 350 for

$$X = \begin{bmatrix} 0 & 0 & 10 \\ 0 & 30 & 0 \\ 20 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix},$$

which means that we do not get the optimal value from this initial solution. It is clear that it may be many such initial solutions. Therefore, in our proposed algorithm, we use a large number of initial solutions in order to increase the possibility of reaching optimal value.

5. EXPERIMENTAL RESULTS

Experiments were conducted by MATLAB 7.0 using a computer with the following technical specifications: Intel®Core^{TM}2 DUO CPU E8500@3.16GHZ, 3.17GHZ, 4GB RAM.

We apply the proposed algorithm to the problems in recently published studies of [20, 24, 30] in which algorithms are proposed for the one-stage FCTP.

Singh and Singh [30] propose PSO based algorithm, Jo et al. [20] and Lotfi ve Tavakkoli-Moghaddam [24] propose a genetic algorithm.

Inputs of the problems P1–P5 taken from the study of [30] are given below in Tables 5–9. The numbers in the cells of the tables indicate unit cost and fixed charge, respectively. For example, the numbers 34 and 91 in the cell (S_1, D_1) of Table 5 means that the unit cost is 34 and the fixed charge is 91 for shipping a product from supplier S_1 to the distributor D_1.

	D_1	D_2	D_3	D_4	
S_1	34;91	97;47	57;44	37;68	76
S_2	99;26	49;62	8;60	70;45	83
S_3	50;57	78;57	47;32	63;96	63
	73	33	66	52	

Tab. 5. Inputs of the Problem 1.

	D_1	D_2	D_3	D_4	D_5	
S_1	6;46	65;57	55;55	23;45	21;58	49
S_2	25;95	82;91	57;60	46;59	71;92	87
S_3	34;31	46;90	40;39	68;56	16;45	97
	40	54	43	42	54	

Tab. 6. Inputs of the Problem 2.

	D_1	D_2	D_3	D_4	D_5	
S_1	45;58	30;33	85;69	61;61	44;72	86
S_2	98;63	62;27	85;37	39;21	78;64	50
S_3	31;68	61;57	100;99	69;46	1;100	99
S_4	90;88	83;94	29;56	43;22	57;47	75
	58	75	31	77	69	

Tab. 7. Inputs of the Problem 3.

	D_1	D_2	D_3	D_4	D_5	D_6	
S_1	77;45	22;88	69;89	8;64	93;42	41;62	50
S_2	19;90	31;56	33;75	16;32	90;80	6;86	98
S_3	53;57	13;81	40;24	69;72	74;98	61;82	95
S_4	36;77	80;92	7;23	63;75	31;52	10;52	87
	54	29	67	70	36	74	

Tab. 8. Inputs of the Problem 4.

	D_1	D_2	D_3	D_4	D_5	D_6	
S_1	73;55	56;30	64;32	50;21	75;52	62;64	76
S_2	89;47	96;45	46;37	97;23	82;90	60;73	78
S_3	95;31	87;61	78;91	20;32	3;34	40;60	40
S_4	51;24	32;54	71;78	33;74	72;31	86;21	42
S_5	58;54	26;37	49;100	59;36	26;84	59;20	51
	57	53	34	59	36	48	

Tab. 9. Inputs of the Problem 5.

All of these problems are balanced problems. Singh and Singh [30] solved these 5 problems with known algorithms and 5 different PSO-based algorithms. They obtained their best result with the least number of iterations using the PSO-5 algorithm. According to the results of this study, the best value for P1, P2, P3, P4, and P5 are 8021, 8364, 9516, 6889, and 12468, respectively. They obtained these values at 49, 31, 236, 274, and

637 iterations, respectively. Comparative results of the algorithms for problems 1-5 are given in Table 10.

Problem	**VAM**	**SSM**	**MODI**	**LINGO**	**PSO-5**	**Proposed algorithm**
P1 (3x4)	8021	8021	8021	8021	8021	**8021**
P2 (3x5)	8428	8364	8364	8364	8364	**8364**
P3 (4x5)	9909	9516	9516	9516	9516	**9516**
P4 (4x6)	6889	6889	6889	6889	6889	**6889**
P5 (5x6)	12492	12492	12492	12468	12468	**12468**

Tab. 10. The results of the algorithms for Problems 1–5.

Since these problems are small in size, the values found by LINGO are optimal. We take population number k as 5 and h as 3 for all these problems.

Matrix X is found as the best solution for problem 1:

$$X = \begin{bmatrix} 24 & 0 & 0 & 52 \\ 0 & 17 & 66 & 0 \\ 49 & 14 & 0 & 0 \end{bmatrix}.$$

The running time of the proposed algorithm is 2.14 s for problem 1.

The best solution obtained for problem 2 is:

$$X = \begin{bmatrix} 0 & 0 & 0 & 38 & 11 \\ 40 & 0 & 43 & 4 & 0 \\ 0 & 54 & 0 & 0 & 43 \end{bmatrix}.$$

The running time of our algorithm is 1.83 s for problem 2.

The best solution obtained for problem 3 is:

$$X = \begin{bmatrix} 28 & 58 & 0 & 0 & 0 \\ 0 & 17 & 0 & 33 & 0 \\ 30 & 0 & 0 & 0 & 69 \\ 0 & 0 & 31 & 44 & 0 \end{bmatrix}.$$

The proposed algorithm found the result at 1.67 s for problem 3.

Our algorithm found the best solution below at 2.22 s for problem 4:

$$X = \begin{bmatrix} 0 & 0 & 0 & 50 & 0 & 0 \\ 4 & 0 & 0 & 20 & 0 & 74 \\ 50 & 29 & 16 & 0 & 0 & 0 \\ 0 & 0 & 51 & 0 & 36 & 0 \end{bmatrix}.$$

Our algorithm found the best solution below at 3.8 s for problem 5:

$$X = \begin{bmatrix} 15 & 2 & 0 & 59 & 0 & 0 \\ 0 & 0 & 34 & 0 & 0 & 44 \\ 0 & 0 & 0 & 0 & 36 & 4 \\ 42 & 0 & 0 & 0 & 0 & 0 \\ 0 & 51 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Problems 6–8 are taken from the study [20]. Inputs of this algorithm are given in Tables 11–13 below.

	D_1	D_2	D_3	D_4	D_5	
S_1	10;100	8;150	12;120	9;80	15;90	30
S_2	11;130	13;80	8;110	10;90	9;140	40
S_3	13;150	11;120	15;90	8;130	10;100	20
S_4	13;110	10;140	12;90	9;80	14;100	10
	20	30	15	25	10	

Tab. 11. Inputs of the Problem 6.

	D_1	D_2	D_3	D_4	D_5	
S_1	8;60	4;88	3;95	5;76	8;97	57
S_2	3;51	6;72	4;65	8;87	5;76	93
S_3	8;67	4;89	5;99	3;89	4;100	50
S_4	4;86	6;84	8;70	3;92	3;88	75
	88	57	24	73	33	

Tab. 12. Inputs of the Problem 7.

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	
S_1	8;160	4;488	3;295	5;376	2;297	1;360	3;199	5;292	2;481	6;162	157
S_2	2;451	3;172	4;265	8;487	5;176	3;260	5;280	1;300	4;354	5;201	293
S_3	7;167	4;250	5;499	3;189	4;340	2;216	4;177	3;495	7;170	3;414	150
S_4	1;386	2;184	8;370	1;292	3;188	1;206	4;340	6;205	8;465	2;273	575
S_5	4;156	5;244	6;460	3;382	3;270	4;180	2;235	1;355	2;276	1;190	310
	225	150	90	215	130	88	57	124	273	133	

Tab. 13. Inputs of the Problem 8.

Problem	**St-GA (Genetic algorithm based algorithm of Jo et al. [20])**	**Pb-GA(Genetic algorithm based algorithm of Lotfi and Tavakkoli-Moghaddam [24])**	**LINGO**	**Proposed algorithm**
6	1610	1610	1610	**1610**
7	1642	1484	1484	**1484**
8	6696	6195	6195	**6195**

Tab. 14. Comparative results of algorithms for Problems 6–8.

Our algorithm solved problem 6 at 0.84 s, problem 7 at 1.02 s, and problem 8 at 1.15 s.

Lotfi and Tavakkoli–Moghaddam [24], produced 5 problems randomly (P9 – 14) and used the algorithm they developed for these problems and compared their results with the algorithm proposed by Jo et al. [20]. To compare, we applied our algorithm to these problems. The dimensions of the problems and the input data can be found in [24].

Comparative results obtained with the proposed algorithm are given in Table 15.

P	**Parameter**		**St-GA**		**Pb-GA**		**Proposed algorithm**	
	Pop /k	**max-gen /h**	**Mean**	**Time (s)**	**Mean**	**Time (s)**	**Mean**	**Time (s)**
10	10 /5	300 /3	9404	2.93	9295	1.96	**9168**	**0.77**
	10 /5	500 /5	9364	4.88	9295	3.25	**9168**	**1.03**
11	10 /5	300 /3	13866	6.95	12751	3.47	**12718**	**2.51**
	10 /5	500 /5	13481	11.54	12734	5.81	**12718**	**3.94**
12	20 /5	300 /5	16388	24.89	14047	9.33	**13934**	**4.48**
	20 /5	500 /10	15750	41.45	14139	15.6	**13930**	**7.06**
	30 /5	500 /5	15621	62.63	13987	23.5	**13934**	**9.45**
13	20 /5	500 /5	27844	85.15	22484	29.9	**22040**	**7.8**
	20 /5	700 /10	27333	119.1	22376	42.1	**22040**	**10.7**
	30 /10	700 /30	27620	180.8	22284	62.8	**22035**	**40.7**
14	30 /5	500 /5	45586	349.1	34119	97.4	**33622**	**12.7**
	30 /15	700 /10	45473	473	33796	136	**33498**	**79**
	40 /25	700 /15	45208	657	33912	182	**33420**	**112**
15	40 /5	700 /5	78145	1612	56399	403	**56508**	**29**
	40 /15	1000 /15	78252	2295	56007	575	**55677**	**212**
	50 /30	1000 /10	77777	2893	55912	722	**55310**	**330**

Tab. 15. Comparative results of the algorithms for Problems 9–14.

6. CONCLUSION

In this study, we examine fixed-charge transportation problems. As it is known, FCTP is an NP-hard problem, that is, there is no algorithm that can find the exact solution in polynomial time. Therefore, in most of the studies in the literature, meta-heuristic algorithms are applied to find an approximate optimal solution. Researchers usually transform the FCTP into a classical TP for this purpose. In this study, we propose a new heuristic algorithm for finding an approximate optimal solution for the FCTP.

There are 2 parameters of our algorithm. The first parameter is the number of initial solutions k and the second one is h indicating the maximum number that the objective function is allowed not to decrease. Here, k corresponds to the population number in the genetic algorithm and h corresponds to the max-gen parameter. The running time of the algorithm and how close it will find the approximate solution to the optimal solution depend on these two parameters.

We apply our algorithm to the problems defined in recently published papers [20], [24], and [30]. Results obtained from the experimental studies indicate that the proposed algorithm is efficient considering running time and being able to find the approximate optimal solution.

ACKNOWLEDGEMENT

We would like to thank Editor-in-Chief Prof. Sergej Čelikovský and the anonymous reviewers for taking the time and effort necessary to review the study. We sincerely appreciate all valuable comments and suggestions, which improved the quality of our paper.

(Received October 7, 2022)

REFERENCES

[1] V. Adlakha and K. Kowalski: On the fixed-charge transportation problem. Omega *27* (1999), 3, 381–388. DOI:10.1016/S0305-0483(98)00064-4

[2] V. Adlakha and K. Kowalski: A simple heuristic for solving small fixed-charge transportation problems. Omega *31* (2003), 3, 205–211. DOI:10.1016/S0305-0483(03)00025-2

[3] V. Adlakha, K. Kowalski, and R. R. Vemuganti: Heuristic algorithms for the fixed-charge transportation problem. Opsearch *43* (2006), 2, 132–151.

[4] V. Adlakha, K. Kowalski, and B. Lev: A branching method for the fixed charge transportation problem. Omega *38* (2010), 5, 393–397. DOI:10.1016/j.omega.2009.10.005

[5] V. Adlakha, K. Kowalski, R. R. Vemuganti, and B. Lev: More-for-less algorithm for fixed-charge transportation problems. Omega: Int. J. Manag. Sci. *35* (2007), 1, 116–127. DOI:10.1016/j.omega.2006.03.001

[6] W. B. Allen and D. Liu: An inventory-transport model with uncertain loss and damage. Logist.Transport. Rev. *29* (1993), 2, 101.

[7] P. Amorim, H. Meyr, C. Almeder, and B. Almada-Lobo: Managing perishability in production-distribution planning: a discussion and review. Flexible Services Manufactur. J. *25* (1993), 3, 389–413.

[8] S. H. Amin and F. Baki: A facility location model for global closed-loop supply chain network design. Appl. Math. Modell. *41* (2017), 316–330. DOI:10.1016/j.apm.2016.08.030

[9] M. L. Balinski: Fixed-cost transportation problems. Naval Res. Logistic Quarterly *8* (1961), 1, 41–54. DOI:10.1002/nav.3800080104

[10] H. I. Calvete, C. Gale, J. A. Iranzo, and P. Toth: A matheuristic for the two-stage fixed-charge transportation problem. Comput. Oper. Res. *95* (2018), 113–122. DOI:10.1016/j.cor.2018.03.007

[11] O. Cosma, P. C. Pop, and D. Dănciulescu: A novel matheuristic approach for a two-stage transportation problem with fixed costs associated to the routes. Comput. Oper. Res. *118* (2020), 104906. DOI:10.1016/j.cor.2020.104906

[12] M. M. El-Sherbiny and R. M. Alhamali: A hybrid particle swarm algorithm with artificial immune learning for solving the fixed charge transportation problem. Comput. Industr, Engrg. *64* (2013), 2, 610–620. DOI:10.1016/j.cie.2012.12.001

[13] M. Eskandarpour, P. Dejax, and O. Peton: A large neighbourhood search heuristic for supply chain network design. Comput. Oper. Res. *8* (2017), 4, 23–37. DOI:10.1016/j.cor.2016.11.012

[14] J. Gottlieb and L. Paulmann: Genetic algorithms for the fixed charge transportation problems. In: Proc. of the IEEE Conf. on Evolutionary Computation, ICEC 1998, pp. 330–335. DOI:10.1109/icec.1998.699754

[15] F. L. Hitchcock: Distribution of a product from several sources to numerous locations. J. Math. Physics *20* (1941), 224–230. DOI:10.1002/sapm1941201224

[16] W. M. Hirsch and G. B. Dantzig: The fixed Charge problem. Naval Res. Log. Quart. *15* (1968), 413–424. DOI:10.1002/nav.3800150306

[17] J. Hong, A. Diabat, V. V. Panicker, and S. Rajagopalan: A two-stage supply chain problem with fixed costs: An ant colony optimization approach. Int. J. Product. Econom. *204* (2018), 214–226. DOI:10.1016/j.ijpe.2018.07.019

[18] N. Jawahar and A. N. Balaji: A genetic algorithm for the two-stage supply chain distribution problem associated with a fixed charge. Europ. J. Oper. Res. *194* (2009), 2, 496–537. DOI:10.1016/j.ejor.2007.12.005

[19] N. Jawahar, A. Gunasekaran, and N. Balaji: A simulated annealing algorithm to the multi-period fixed charge distribution problem associated with backorder and inventory. Int. J. Prod. Res. *50* (2011), 9, 2533–2554. DOI:10.1080/00207543.2011.581013

[20] J. B. Jo, Y. Li, Y., and M. Gen: Nonlinear fixed charge transportation problem by spanning tree-based genetic algorithm. Comput. Industr. Engrg. *53* (2007), 2, 290–298. DOI:10.1016/j.cie.2007.06.022

[21] N. Kartlı, E. Bostancı, and M. S. Güzel: A new algorithm for the initial feasible solutions of fixed charge transportation problem. In: 7th International Conference on Computer Science and Engineering (UBMK), IEEE 2022, pp. 82–85. DOI:10.1109/ubmk55850.2022.9919524

[22] P. Li: Solving the sensor cover energy problem via integer linear programming. Kybernetika *57* (2021), 4, 568–593. DOI:10.14736/kyb-2021-4-0568

[23] V. Lin: Binary integer programming solution for troubleshooting with dependent actions. Kybernetika *53* (2017), 3, 493–512. DOI:10.14736/kyb-2017-3-0493

[24] M. M. Lotfi and R. Tavakkoli-Moghaddam: A genetic algorithm using priority-based encoding with new operators for fixed charge transportation problems. Appl. Soft Comput. *13* (2013), 5, 2711–2726. DOI:10.1016/j.asoc.2012.11.016

[25] V. V. Panicker, R. Vanga, and R. Sridharan: Ant colony Optimization algorithm for distribution-allocation problem in a two-stage supply chain with a fixed transportation charge. Int. J. Prod. Res. *51* (2012), 3, 698–717. DOI:10.1080/00207543.2012.658118

[26] V. V. Panicker, R. Sridharan, and B. Ebenezer: Three-stage supply chain allocation with fixed cost. J. Manuf. Technol. Manag. *23* (2012), 7, 853–868. DOI:10.1108/17410381211267691

[27] P. C. Pop, C. Sabo, B. Biesinger, B. Hu, and G. R. Raidl: Solving the two-stage fixed charge transportation problem with a hybrid genetic algorithm. Carpathian J. Math. *33* (2017), 3, 36–371.

[28] K. A. A. D. Raj and C. Rajendran: A hybrid genetic algorithm for solving single-stage fixed-charge transportation problems. Technol. Oper. Manag. *2* (2011), 1, 1–15. DOI:10.1016/j.proeng.2011.08.023

[29] K. A. A. D. Raj and C. Rajendran: A genetic algorithm for solving the fixed-charge transportation model: two-stage problem. Comput. Oper. Res. *39* (2012), 9, 2016–2032. DOI:10.1016/j.cor.2011.09.020

[30] G. Singh and A. Singh: Solving fixed-charge transportation problem using a modified particle swarm optimization algorithm. Int. J. System Assurance Engrg. Management *12* (2021), (6), 1073–1086. DOI:10.1007/s13198-021-01171-2

[31] M. Sun, J. E. Aronson, P. G. Mckeown, and D. Drinka: A tabu search heuristic procedure for the fixed charge transportation problem. Europ. J. Oper. Res. *106* (1999), 2–3, 411–456. DOI:10.1016/s0377-2217(97)00284-1

[32] F. G. Tari and I. Hashemi: Prioritized K-mean clustering hybrid GA for discounted fixed charge transportation problems. Comput. Industr. Engrg. *126* (2018), 63–74. DOI:/10.1016/j.cie.2018.09.019

Nermin Kartli, Computer Engineering Department, Ankara University, Ankara, 06830. Turkey.
e-mail: nkartli@ankara.edu.tr

Erkan Bostanci, Computer Engineering Department, Ankara University, Ankara, 06830. Turkey.
e-mail: ebostanci@ankara.edu.tr

Mehmet Serdar Guzel, Computer Engineering Department, Ankara University, Ankara, 06830. Turkey.
e-mail: mguzel@ankara.edu.tr