Scientia Iranica E (2021) 28(1), 532–546

Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
http://scientiairanica.sharif.edu

Cross-dock scheduling considering time windows and deadline for truck departures

A. Golshahi-Roudbaneh[a], M. Hajiaghaei-Keshteli[a,*], and M.M. Paydar[b]

a. *Department of Industrial Engineering, University of Science and Technology of Mazandaran, Behshahr, Iran.*
b. *Department of Industrial Engineering, Babol Noshirvani University of Technology, Babol, Iran.*

Received 10 January 2019; received in revised form 15 March 2019; accepted 20 May 2019

KEYWORDS
Cross-docking;
Truck scheduling;
Trucks departure;
Keshtel algorithm;
Hybrid metaheuristic.

Abstract. Recent years have seen a great deal of interest in optimizing logistics and transforming systems. An important challenge in this regard is cross dock scheduling with several real-life limitations such as the deadline for both perishable and imperishable products. This study is a new cross-dock scheduling problem considering not only a time window but, also, for all shipping trucks, in this research area, the deadline is assumed by the presence of perishable products for the first time. Based on these suppositions, a new mathematical model is developed. Last but not least a new hybrid metaheuristic is proposed by combining a recent nature-inspired metaheuristic called the Keshtel Algorithm (KA) and a well-known algorithm named Simulated Annealing (SA). The proposed hybrid algorithm is not only compared with its individual parts but some other well-known metaheuristic algorithms are also used. Finally, the performance of the proposed algorithm is validated by several experiments with different complexities and statistical analyses.

1. Introduction

Nowadays, according to industrial development and rapid changes in today's competitive market, the issue of customer satisfaction has become more than ever crucial for companies [1]. These days, if supply chain performance does not meet customer satisfaction, it will not be considered efficient. Therefore, taking into account effective criteria in accomplishing customer satisfaction could improve company performance [2]. Generally, customers mainly want to receive high-quality products at the proper location, at the proper time, and with the lowest cost. However, each customer may have their own personal definition of quality and correct location, proper time and cost bear invariable definitions among the majority [3]. Hence, taking them into account could create satisfaction among a wide range of customers.

Paying attention to distribution centers and optimizing them throughout the supply chain could aid considerably in achieving the above goals [4]. A favorable procedure for increasing the efficiency of the distribution centers is a cross docking system which has recently been highly regarded. Cross docking is a distribution concept in which products are entered into the inbound dock by receiving trucks, and after being sorted in accordance with customer demand, they are directly transferred to the outbound dock in order to be loaded into shipping trucks. Long-term storage is not allowed in this system. Therefore, the cross-docking system helps to improve the physical flow of products in the supply chain. It also eliminates both long-run storage and product retrieval costs. In fact, the better the performance and capability of a cross

*. *Corresponding author.*
E-mail address: mostafahaji@mazust.ac.ir (M. Hajiaghaei-Keshteli)

doi: 10.24200/sci.2019.52662.2824

docking system, the better the efficiency in increasing distribution performance and also customer satisfaction. Apte and Viswanathan [5] investigated several techniques to improve the efficiency of a cross docking center. Among the important ones, one can refer to the automated material handling system, efficient use of Information Technology (IT), taking advantage of the whole capacity of trucks in terms of product transfer, and the effective utilization of design and management tools. They also mention that applying a cross docking system will be appropriate when demand rate is stable and stock-out cost is low. But, when demand rate is unstable and stock-out cost is high, utilizing a traditional warehousing system is more suitable. In some cases implementing both systems simultaneously will be more effective (see, [6]).

In literature, various divisions, including location of cross-docks, layout design, vehicle routing, truck Scheduling, dock door assignment, networks, etc. are presented to classify crossdocking problems (see, [7,8]). Ladier and Alpan [9] reviewed the cross docking operation. They categorized problems into five groups including truck to door assignment, truck to door sequencing, truck to door scheduling, truck sequencing and truck scheduling. They believed that in most performed studies, minimization of the makespan and travel distance was considered a performance measure. They also surveyed the gap between existent studies and industry needs, and then claimed that considering a deadline for shipping trucks would be necessary.

Some studies are also highly significant in the field of scheduling. One of them is the work carried out by Yu [10]. He studied the truck scheduling problem with the aim of determining the optimized sequence for receiving and shipping trucks and minimizing makespan. The study presented by Yu and Egbelu [11] has been considered by many researchers. They presented a mathematical model for the truck scheduling problem in which a receiving door and a shipping door and also a temporary storage in front of the shipping door are considered. They suggested nine heuristic methods as solutions for the model. Then, the efficiency of the suggested heuristic methods was investigated through comparing them with the exact results obtained from the complete enumeration method.

Chen and Lee [12] studied the truck scheduling problem as a flow-shop machine scheduling problem. They solved the problem by the use of the branch-and-bound algorithm. According to the authors, this algorithm can find the optimal solution of problems up to 60 jobs in an acceptable period of time. Boysen [13] studied a cross dock scheduling problem in storage ban mode. The purpose of the problem included minimizing flow time, processing time and tardiness of outbound trucks, respectively. They used dynamic programming and Simulated Annealing (SA) methods to solve the mathematical model.

Li et al. [14] considered the cross docking scheduling problem as a two-phase parallel machine problem with earliness and tardiness. Amini and Tavakkoli-Moghaddam [15] discussed a problem in which trucks might face breakdown during the service times. They also considered a due date for each shipping truck. They used three multi-objective meta-heuristic algorithms to solve the problem. In 2017, Golshahi-Roudbaneh et al. [16] proposed heuristics and meta-heuristics to find the optimal for a receiving and shipping trucks sequence, based on Yu [10]. In another similar research, Serrano et al. [17] proposed a mixed integer linear programming model to schedule inbound truck arrival time (considering given soft time windows), shop-floor repackaging operations and outbound truck departure times. In 2018, Motaghedi-Larijani and Aminnayeri [18] proposed a queuing model in order to optimize the number of outbound doors based on minimizing the total costs, including the costs of adding a new outbound door and the expected waiting time of customers. Similarly, Mohammadzadeh et al. [19] proposed truck scheduling based on benchmarks generated by Golshahi-Roudbaneh et al. [16], and solved it by three recent nature-inspired metaheuristics, including Virus Colony Search (VCS), Water Wave Optimization (WWO) and Red Deer Algorithm (RDA). More recently, Baniamerian et al. [20] considered a profitable heterogeneous vehicle routing problem with cross-docking. They formulated a mixed integer linear programming model. A new hybrid meta-heuristic algorithm based on a Modified Variable Neighborhood Search (MVNS) with four shaking and two neighborhood structures and a Genetic Algorithm (GA) is presented to solve large-sized problems. The results are compared with those obtained with an Artificial Bee Colony (ABC) and a SA algorithm.

A main case in relation to scheduling problems is selecting the solution method. Due to intricate problems, in most studies, different heuristic and meta-heuristic methods are applied in order to find answers. Table 1 illustrates the approach of recent papers with regard to scheduling problems.

This paper investigates a new truck scheduling problem in a cross docking system. A mathematical model is developed on the basis of the rest of the models that exist in this area. In this paper, a time window is regarded for every shipping truck, and products are divided into two groups; namely perishable and imperishable. Due to the presence of perishable products, a deadline is considered for the shipping trucks. To the best of the authors knowledge, there is no similar paper that considers all these suppositions simultaneously in this research area. In order to solve the model in large scale, a strong hybrid algorithm is suggested.

Table 1. Solution method of the related studies to this paper.

Paper(s)	Method		
	Exact	Heuristic	Metaheuristic
Yu and Egbelu [11]	√	√	–
Chen and Song [34]	√	√	–
Boysen [13]	√	–	√
Soltani and Sadjadi [35]	√	–	√
Boysen et al. [36]	–	√	–
Forouharfard and Zandieh [37]	–	–	√
Larbi et al. [38]	√	√	–
Arabani et al. [39]	√	–	√
Shakeri et al. [40]	–	√	–
Berghman et al. [41]	√	–	–
Davoudpour et al. [42]	–	–	√
Sadykov [43]	√	–	–
Boysen et al. [44]	√	√	√
Van Belle et al. [45]	√	–	√
Bjelić et al. [46]	–	–	√
Joo and Kim [47]	–	–	√
Konur and Golias [48]	–	√	√
Ladier and Alpan [49]	√	√	–
Ladier and Alpan [50]	√	√	√
Madani-Isfahani et al. [51]	–	–	√
Amini et al. [52]	–	√	√
Mohtashami et al. [53]	–	–	√
Golshahi-Roudbaneh et al. [16]	–	√	√
Khalili-Damghani et al. [54]	√	–	√
Wisittipanich and Hengmeechai [55]	–	–	√
Mohammadzadeh et al. [19]	–	–	√
Moteghedi-Larijani and Aminnayeri [18]	√	√	–
Beniamerian et al. [20]	–	–	√

This paper is organized as follows. The proposed model is formulated in Section 2. Section 3 explains metaheuristic algorithms. The parameter settings of the algorithms are described in Section 4. Section 5 depicts the computational results, and finally, Section 6 presents conclusions.

2. Problem description

Here, initially, the basic mathematical model is illustrated, and subsequently, the new suppositions are considered, including the time window and the deadline for truck departures using different types of product simultaneously for the first time in order to develop the proposed model.

2.1. Basic mathematical model

The mathematical model shown below is the same as that developed by Yu and Egbelu [11]. The following notations are used to define the mathematical model.

Parameters

R Number of receiving trucks

S Number of shipping trucks

N Number of product types

r_{ik} Number of units of product type k that was initially loaded in receiving truck i

s_{jk} Number of units of product type k that was initially needed for shipping truck j

D	Truck changeover time
V	Moving time of products from receiving dock to shipping dock
M	Big number

Continuous variables

T	Makespan
c_i	Time at which receiving truck i enters the receiving dock
F_i	Time at which receiving truck i leaves the receiving dock
d_j	Time at which shipping truck j enters the shipping dock
L_j	Time at which shipping truck j leaves the shipping dock

Integer variables

X_{ijk}	Number of units of product type k that transfer from receiving truck i to shipping truck j

Binary variables

$$v_{ij} = \begin{cases} 1, & \text{If any products transfer from receiving truck } i \text{ to shipping truck } j; \\ 0, & \text{Otherwise.} \end{cases}$$

$$p_{ij} = \begin{cases} 1, & \text{If receving truck } i \text{ preceeds receiving truck } j \text{ in the receiving truck sequence;} \\ 0, & \text{Otherwise.} \end{cases}$$

$$q_{ij} = \begin{cases} 1, & \text{If shipping truck } i \text{ preceeds shipping truck } j \text{ in the shipping truck sequence;} \\ 0, & \text{Otherwise.} \end{cases}$$

The mathematical model is formulated as explained below:

$$\min T$$

s.t. :

$$T \geq L_j, \qquad \text{for all } j, \tag{1}$$

$$\sum_{j=1}^{S} x_{ijk} = r_{ik}, \qquad \text{for all } i, k, \tag{2}$$

$$\sum_{i=1}^{R} x_{ijk} = s_{ik}, \qquad \text{for all } j, k, \tag{3}$$

$$x_{ijk} \leq M v_{ij}, \qquad \text{for all } i, j, k, \tag{4}$$

$$F_i \geq c_i + \sum_{k=1}^{N} r_{ik}, \qquad \text{for all } i, \tag{5}$$

$$c_i \geq F_i + D - M(1 - p_{ij}), \quad \text{for all } i, j \text{ and where } i \neq j, \tag{6}$$

$$c_i \geq F_j + D - M p_{ij}, \quad \text{for all } i, j \text{ and where } i \neq j, \tag{7}$$

$$p_{ii} = 0, \qquad \text{for all } i, \tag{8}$$

$$L_j \geq d_j + \sum_{k=1}^{N} s_{jk}, \qquad \text{for all } j, \tag{9}$$

$$d_j \geq L_i + D - M(1 - q_{ij}), \quad \text{for all } i, j \text{ and where } i \neq j, \tag{10}$$

$$d_j \geq L_j + D - M q_{ij}, \quad \text{for all } i, j \text{ and where } i \neq j, \tag{11}$$

$$q_{ii} = 0, \qquad \text{for all } i, \tag{12}$$

$$L_j \geq c_i + V + \sum_{k=1}^{N} x_{ijk} - M(1 - v_{ij}), \quad \text{for all } i, j, \tag{13}$$

all variables ≥ 0.

2.2. Development mathematical model

In this model, a time window and a deadline are considered for each shipping truck. There are several types of product which are divided into two groups namely; perishable and imperishable. The model assumptions are touched on as follows:

- If the shipping truck carries perishable products, its departure time can never exceed the determined deadline;
- If the shipping truck carries imperishable products, it is possible that its departure time could exceed the determined deadline;
- There are a time window and a deadline which are both unique for each shipping truck.

In this model, if the departure time of truck j is more than its deadline, a tardiness penalty cost will be allocated to the time difference between tardiness and deadline, and a deadline penalty cost will be assigned to the time difference between departure time and deadline of shipping truck j.

Before model development, essential notations are defined as follows:

Parameters

DD_j	Due date of shipping truck j
l_j	Upper bound of time window for shipping truck j
e_j	Lower bound of time window for shipping truck j
dl_j	Deadline of shipping truck j
α_{1j}	Earliness penalty cost of shipping truck j carrying imperishable products
α_{2j}	Earliness penalty cost of shipping truck j carrying perishable products
β_{1j}	Tardiness penalty cost of shipping truck j carrying imperishable products
β_{2j}	Tardiness penalty cost of shipping truck j carrying perishable products
β_{3j}	Deadline penalty cost of shipping truck j

Continuous variables

T_j	Tardiness of shipping truck j
E_j	Earliness of shipping truck j

2.2.1. Objective function

The objective function is minimizing total costs resulting from the tardiness and earliness of shipping trucks.

$$\min \sum \alpha_{1j} \times \max(0, e_j - L_j) \times (1 - W_j) + \beta_{1j} \times \max(0, L_j - l_j) \times (1 - W_j) \times (1 - Y_j) + \alpha_{2j} \times \max(0, e_j - L_j) \times W_j + \beta_{2j} \times \max(0, L_j - l_j) \times W_j + \left(\beta_{1j} \times (dl_j - l_j) + \beta_{3j} \times (L_j - dl_j)\right) \times Y_j, \tag{14}$$

in which the first and second terms, respectively, calculate the earliness and tardiness penalty of shipping trucks carrying imperishable products. The third and fourth terms similarly calculate the earliness and tardiness penalty of shipping trucks carrying perishable products. The fifth term computes the penalty amount when the departure time of shipping trucks is greater than the predetermined deadline. It deserves a mention that y can accept 1 only for trucks not carrying perishable products. In other words, shipping trucks carrying perishable products are not allowed to have a departure time greater than the determined deadline. Figure 1 demonstrates the method of computing the objective function in different moods for a shipping truck lacking perishable products. The horizontal axis shows departure time.

The objective function can be revised as follows:

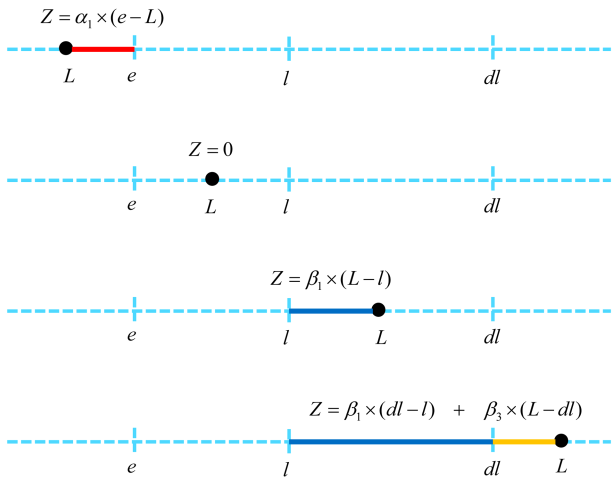

Figure 1. An example of computing the objective function.

$$\min \sum_{j=1}^{S} \Big[(\alpha_{1j} \times E_j) \times (1 - W_j) + (\beta_{1j} \times T_j) \times (1 - W_j) \times (1 - Y_j) + (\alpha_{2j} \times E_j) \times W_j + (\beta_{2j} \times T_j) \times W_j + (\beta_{1j} \times T_j + \beta_{3j} \times (L_j - dl_j)) \times Y_j \Big]. \tag{15}$$

In this case, Constraints (17), (18) and (19) are added to the model.

2.2.2. Constraints

As mentioned in the model assumption, the departure time of perishable products should not exceed the determined deadline. The following constraint presents this guarantee:

$$L_j \le dl_j + M(1 - W_j) \qquad \text{for all } j. \tag{16}$$

In fact, constraint ensures that if jth shipping truck is carrying perishable products, its departure time should not be greater than the deadline. If it carries imperishable products, it is possible that its departure time will be greater than the deadline, which faces a heavy penalty.

Simplifying the objective function in the previous section, the following constraints are added to the model:

$$E_j \ge e_j - L_j \qquad \text{for all } j, \tag{17}$$

$$T_j \ge L_j - l_j \qquad \text{for all } j, \tag{18}$$

$$T_j \ge (L_j - dl_j) \times Y_j \qquad \text{for all } j. \tag{19}$$

Constraints (17) and (18), respectively, compute earliness and tardiness values, if any, for jth shipping truck.

Constraints (19) compute the tardiness value if the jth shipping truck does not carry perishable products and its departure time is greater than the deadline.

2.2.3. Corollary

Finally, the whole model can be written as follows:

$$\min z = \sum_{j=1}^{S} \Big[(\alpha_{1j} \times E_j) \times (1 - W_j) + (\beta_{1j} \times T_j) \times (1 - W_j) \times (1 - Y_j) + (\alpha_{2j} \times E_j) \times W_j + (\beta_{2j} \times T_j) \times W_j + (\beta_{1j} \times T_j + \beta_{3j} \times (L_j - dl_j)) \times Y_j \Big], \quad (20)$$

such that:

$$\sum_{j=1}^{S} x_{ijk} = r_{ik}, \qquad \text{for all } i, k, \quad (21)$$

$$\sum_{i=1}^{R} x_{ijk} = s_{ik}, \qquad \text{for all } j, k, \quad (22)$$

$$x_{ijk} \leq M v_{ij}, \qquad \text{for all } i, j, k, \quad (23)$$

$$F_i \geq c_i + \sum_{k=1}^{N} r_{ik}, \qquad \text{for all } i, \quad (24)$$

$$c_i \geq F_i + D - M(1 - p_{ij}), \quad \text{for all } i, j \text{ and where } i \neq j, \quad (25)$$

$$c_i \geq F_j + D - M p_{ij}, \quad \text{for all } i, j \text{ and where } i \neq j, \quad (26)$$

$$p_{ii} = 0, \qquad \text{for all } i, \quad (27)$$

$$L_j \geq d_j + \sum_{k=1}^{N} s_{jk}, \qquad \text{for all } j, \quad (28)$$

$$d_j \geq L_i + D - M(1 - q_{ij}), \quad \text{for all } i, j \text{ and where } i \neq j, \quad (29)$$

$$d_j \geq L_j + D - M q_{ij}, \quad \text{for all } i, j \text{ and where } i \neq j, \quad (30)$$

$$q_{ii} = 0, \qquad \text{for all } i, \quad (31)$$

$$L_j \geq c_i + V + \sum_{k=1}^{N} x_{ijk} - M(1 - v_{ij}), \quad \text{for all } i, j, \quad (32)$$

$$L_j \leq dl_j + M(1 - W_j) \qquad \text{for all } j, \quad (33)$$

$$E_j \geq e_j - L_j \qquad \text{for all } j, \quad (34)$$

$$T_j \geq L_j - l_j \qquad \text{for all } j, \quad (35)$$

$$T_j \geq (L_j - dl_j) \times Y_j \qquad \text{for all } j, \quad (36)$$

all variables ≥ 0.

3. Metaheuristics

To solve the developed model based on the encoding plan of [16], not only has the SA and Differential Evolution (DE) been utilized as successful traditional metaheuristics from the literature, but also the Keshtel Algorithm (KA), a recent nature-inspired algorithm, is applied to solve the proposed problem. Last but not least, the main innovation for solving the proposed problem is to introduce a new hybrid metaheuristic formulated by KA and SA to better solve the proposed problem. The main motivation of this study is to propose a new hybrid algorithm that refers to a no free lunch theorem [21,22]. Based on this theory, there is no algorithm to solve all optimization problems properly. This means that the chance for a new metaheuristic always exits to show a better result in comparison with other existing metaheuristics to solve NP-hard problems such as the proposed truck scheduling considered by this study [23].

3.1. Simulated annealing

The SA algorithm was presented by Kirkpatrick et al. [24] for the first time. An optimization problem will be solved through this algorithm if a primary solution is firstly generated randomly and then is evaluated by the fitness function [25]. Then, a neighborhood solution is generated and evaluated. Hence, one of the following thee condition occurs:

1. The neighbor solution is better than the current solution. In this case, it is substituted by the neighbor solution;
2. The neighbor solution is worse than the current solution. In this state, the system is allowed to accept the neighbor solution with a probability as follows:

$$p = e^{\frac{\Delta f}{T}}, \quad (37)$$

   in which Δf is the difference between the fitness function value of the current solution and that of the neighbor solution. T is a parameter called temperature.
3. If the neighbor solution is not accepted regarding conditions 1 or 2, it will be omitted and a new neighborhood generated on the current solution.

At the first iterations of the algorithm, the temperature value is considered at a high level in order to raise the chance of the accepted worse solution. Then, at each iteration, the temperature decreases gradually. Ultimately, the algorithm converges toward a fine solution.

3.2. Differential Evolution (DE)

The DE algorithm was presented by Storn and Price [26], to solve optimization problems. In this algorithm, random vectors called target vectors are generated in the same number as population size (N_{pop}).

$$V_i(t), \qquad i = 1, 2, ..., N_{pop}.$$

Then changes from one generation to another are implemented by such operators as mutation, crossover, and selection. In this algorithm, unlike the GA, the first mutation operator and then a crossover operator are exerted [27].

Mutation

In order to exert a mutation operator, the following actions should be taken:

- Let $V_i(t)$ be the target vector in generation t. The first three vectors are randomly opted through target vectors. It deserves to be mentioned that the three opted vectors must be different from vector $V_i(t)$;
- Then, the difference between the two vectors is computed, which is named the difference vector;
- Ultimately, a weight from the difference vector is added to the third vector. The acquired vector is called the mutant vector.

Hence, the mutant vector is obtained by the following equation:

$$M_i(t) = V_a(t) + F\left[V_b(t) - V_c(t)\right], \tag{38}$$

where $V_a(t)$, $V_b(t)$ and $V_c(t)$ are three vectors selected randomly from the vectors' population in generation t and which differ from vector $V_i(t)$. F is a constant real value $\in [0, 1]$.

Crossover

In order to apply the crossover operator, the mutant vector (M_i) and the current target vector ($V_i(t)$) are mixed with one another. The acquired vector is called a trial vector. Hence, the jth dimension of trial vector i can be generated through an equation as follows:

$$T_{ij}(t) = \begin{cases} M_{ij}(t) & \text{if } rand\ (0,1) \leq \text{CR or} \quad j = j^* \\ V_{ij}(t) & \text{otherwise} \end{cases}$$

$$j = 1, 2, ..., D, \tag{39}$$

in which CR is the crossover constant selected from the uniform distribution in [0,1]. D is the dimension of vectors and j^* is a random integer number $\in (1, 2, ..., D)$, which guarantees that $T_{ij}(t)$ gets at least one value from $M_{ij}(t)$.

Selection

In order to employ the selection operator, the acquired trial vector is compared with the target vector in terms of fitness function value. If the trial vector bears a better value, it will be replaced; otherwise, the target vector will be kept for the next generation. With continuing this process, the population vectors of generation $t + 1$ are identified.

The pseudo code of the *DE* algorithm is as follows:

```
V_i(i = 1,2,...,N_pop) ← generate initial random target vectors
f(V_i) ← evaluate target vector V_i based on a fitness function
While t ≤ max number of iteration Do
      For i = 1: population size (N_pop) Do
         M_i(t) ← generate mutant vector
         according to Eq. (38)
         T_i(t) ← generate trial vector according to
         Eq. (39)
         f(T_i) ← evaluate trial vector T_i based on
         a fitness function
        If f(T_i) ≤ f(V_i)
            V_i(t+1) ← T_i
        Else;
            V_i(t+1) ← V_i;
      End if
    End for
    t = t + 1
End while
```

3.3. Keshtel algorithm (KA)

The KA was proposed by Hajiaghaei-Keshteli and Aminnayeri [28,29] in order to solve continuous optimization problems. Keshtel is the name of a bird. This bird shows very interesting behavior after finding food. The lucky Keshtels find better food in the lake. Then, the other Keshtels in their neighborhood are attracted towards them and all at once start to swirl around the food source. During swirling, if a Keshtel finds a better food source, it will be identified as a lucky Keshtel. Moreover, several Keshtels move toward intact spots of the lake in order to find other food. During this movement, they consider the position of two other Keshtels. In the lake, there are also some other Keshtels that do not manage to find any food. They leave the lake and are replaced with newcomer Keshtels.

3.3.1. Primary procedure
Population members, like the KA, are divided into three sections. Let N be the set of population members, then:

$$N = N_1 \cup N_2 \cup N_3. \quad (40)$$

In the above equation, N_1 includes a number of population members bearing a better value of fitness function compared to the rest of the members (lucky Keshtels). N_2 includes a number of population members bearing the worst value of fitness function compared to the rest of the members. N_3 includes population members which do not exist in N_1 and N_2 sets.

3.3.2. Attraction and swirling
As mentioned, if a lucky Keshtel finds a food source in the lake, another Keshtel in its neighborhood will be attracted towards it and swirl around the food source with a specific radius. Meanwhile, if it discovers a better food source, it is identified as a lucky Keshtel. Otherwise, after each swirl, it reduces the radius. This swirl continues until the food source is finished. This procedure is presented in Figure 2.

Let a maximum number of swirling ($S_{\max}$) be equaled to 3. According to Figure 2, one can create maximum $(2 \times S_{\max} - 1) = 5$ new solutions. The neighbor solutions can be generated by the following equations:

$$Position_1 = (a + (b - a)),$$

$$Position_2 = (a + (b - a)/3),$$

$$Position_3 = (Position_1 - (b - a)/3),$$

$$Position_4 = (b - (b - a)/3),$$

$$Position_5 = (b + (b - a)/3).$$

3.3.3. Replace the members of N_2 set with new ones
Not having been able to find any food, Keshtels leave the lake and new Keshtels hoping to find food come to the lake. Therefore, the members of set N_2 bearing a worse value of objective function than that of other members are omitted and new members are produced randomly and replaced.

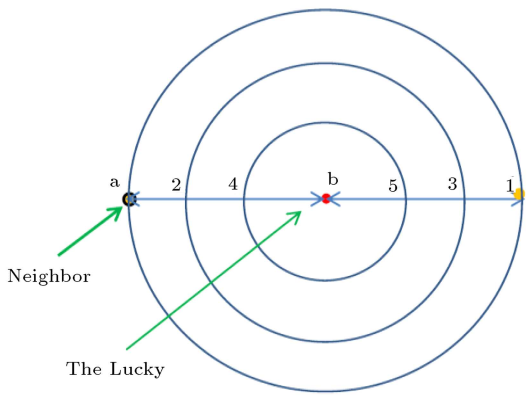

Figure 2. Attraction and swirling process.

3.3.4. Move the member of N_3 set
Each member of the N_3 set changes its position towards virgin spots in terms of the other two members' position. Let Y_i be a member of the N_3 set. It changes its position as follows:

$$v_i = \lambda_1 \times Y_j + (1 - \lambda_1) \times Y_t, \quad (41)$$

$$Y_i = \lambda_2 \times Y_i + (1 - \lambda_2) \times v_i, \quad (42)$$

in which Y_j and Y_t are two members selected randomly from the N_3 set and different from Y_i. λ_1 and λ_2 are random numbers selected from the uniform distribution in [0,1].

3.4. Hybrid KA-SA

The KA is an extraordinary operation in solving continuous problems. SA has also been designed in such a way to show a fine operation in detecting a near-optimal solution for both discrete and continuous problems [30]. Taking advantage of the KA's power to solve discrete problems led the authors to make alterations to the local search process. Striking a better balance between both diversification and intensification phases resulted in motivation to design the proposed algorithm. The KA bears a very high intensification. The property of the SA algorithm in forgetting some answers for initial iterations can be greatly helpful so that the algorithm does not undergo premature convergence.

The following steps are considered for the intended algorithm:

Step 1. The initial population is generated randomly and then evaluated according to the fitness function.

Step 2. Population members, like the KA, are divided into three sections according to Eq. (39).

Step 3. This step is applied to each member of the N_1 set and with a specific iteration number. Let X_i be a member of the N_1 set.

3-1 A random solution is generated in the neighborhood of X_i. Two methods are applied to create this neighborhood solution (i.e. swap and inversion). In the swap method, two genes are randomly chosen and their positions exchanged. In the inversion method, two points along the chromosome are randomly selected. Then, the sub-section between these two points is rotated 180 degrees;

3-2 The neighbor solution is evaluated according to the fitness function;

3-3 If the neighbor solution is better than X_i, then X_i will be replaced with it;

3-4 If the neighbor solution is not better than X_i, it will be accepted with a probability, as mentioned in Eq. (37);

Step 4. Members of the N_2 set, including solutions with the worst value of fitness function, are omitted, and instead new solutions are generated randomly;

Step 5. Each member of the N_3 set updates its position according to Eqs (41) and (42).

The procedure of the proposed KA-SA algorithm is briefly shown in Figure 3.

The pseudo code of the DE algorithm is as follows:

1. $X_i(i = 1, 2, ..., N_{pop})$ = generate initial random population
2. $f(Xi)$ = evaluate each member X_i based on fitness function
3. Allocate population members to N_1, N_2 and N_3 sets
4. **While** stopping condition not confirmed **Do**
5. **For** each member of the N_1 set **Do**
6. **For 1**: maximum considered number of Sub Iteration **Do**
7. Generate and evaluate a neighborhood solution X_i' of $X_i \in N_1$
8. **If** $f(X_i') \leq f(X_i)$
9. $X_i = X_i'$
10. **Else**
11. $\Delta f = f(X_i') - f(X_i)$
12. **If** *uniform* $(0, 1) \leq \exp(-\Delta f / T)$
13. $X_i = X_i'$
14. **End if**
15. **End if**
16. **End for**
17. **End for**
18. Reduce temperature
19. **For** each member of the N_2 set **Do**
20. Replace members of the N_2 set with the randomly generated new ones
21. **End for**
22. **For** each member of the N_3 set **Do**
23. Move solution according to Eqs. (41) and (42)
24. **End for**
25. **End while**.

4. Parameters setting

Parameter settings are a main issue in the use of metaheuristics algorithms, since the quality of solutions depends on the algorithm parameters to a large extent [30]. Tables 2–5 represent each algorithm parameters and their respective levels.

Testing all possible states may be very time consuming. For example, according to Table 2, there are 81 different trials for one problem in the SA algorithm. The design of experiments is a technique

Table 2. Parameters and their level in Simulated Annealing (SA).

Parameters	Levels		
	1	2	3
Maximum iteration (*Iter*)	1000	800	600
Initial temperature (T_0)	300	200	100
Temperature reduction rate (r)	0.99	0.9	0.8
Number of sub iteration (*Subiter*)	75	50	25

Table 3. Parameters and their level in Differential Evaluation (DE).

Parameters	Levels		
	1	2	3
Maximum iteration (*Iter*)	1000	1500	2000
Number of population (*NP*)	50	100	150
Crossover constant (cr)	0.7	0.5	0.3

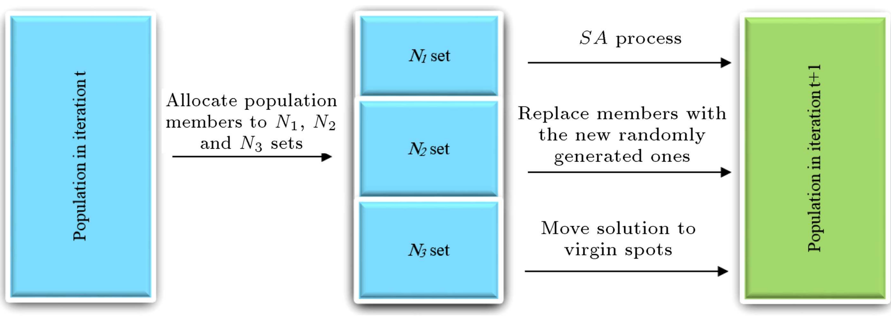

Figure 3. KA-SA procedure.

Table 4. Parameters and their level in Keshtel Algorithm (KA).

Parameters	Levels		
	1	2	3
Maximum iteration ($Maxit$)	450	650	–
Population size (n_{pop})	100	200	300
Percentage of $N1$ Keshtel (PN_1)	0.03	0.06	0.09
Percentage of N_2 Keshtel (PN_2)	0.2	0.3	0.4
Maximum Swirling ($S_{\max}$)	2	3	5

Table 5. Parameters and their level in KA-SA.

Parameters	Levels		
	1	2	3
Maximum iteration ($Maxit$)	450	650	–
Population size (n_{pop})	100	200	300
Percentage of N_1 Keshtel (PN_1)	0.05	0.1	0.15
Percentage of N_2 Keshtel (PN_2)	0.2	0.3	0.4
Maximum Swirling ($S_{\max}$)	10	15	20
Initial temperature (T_0)	200	400	600
Temperature reduction rate (r)	0.99	0.95	0.9

creating the highest payoff with minimal cost and time. The Taguchi method [31] is one of the most well-known and powerful ones. As a result, the current paper uses the Taguchi method in order to examine the impact of the value of the parameter on algorithm performance, as well as to obtain higher quality answers. The method reduces the tests and uses the S/N ratio in order to determine the parameters optimum levels.

Having determined the number of parameters and their levels, the number of required tests is specified through the proposed Taguchi table, known as Orthogonal arrays. Standard orthogonal arrays fix most experimental design needs, but, sometimes, the adjustments are unavoidable. Each experiment is repeated several times because of the random nature of metaheuristics algorithms. Here, each experiment has been repeated ten times. The results were analyzed using the S/N ratio, which will be acquired through the following equation:

$$S/N\ ratio = -10 \log \sum_i \sum_j f_{ij}^2, \qquad (43)$$

where, f_{ij} is the objective function value acquired in the jth replication of the ith experiment for each problem. Each level of the parameters that have the highest amount of S/N ratio is chosen as the optimal level [32,33]. The S/N ratio at each level of the parameters is shown in Figures 4–7. The results are shown in Table 6.

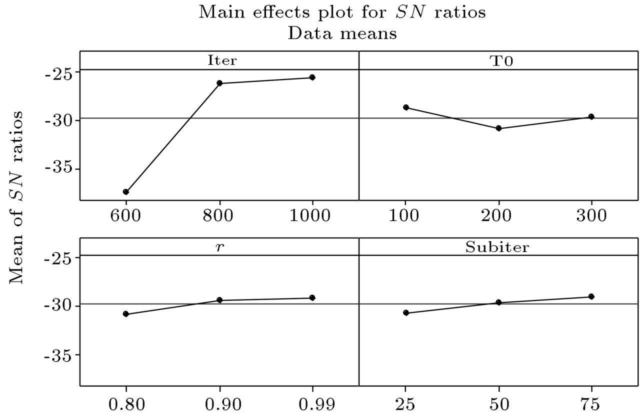

Figure 4. Average S/N ratio for Simulated Annealing (SA) parameters.

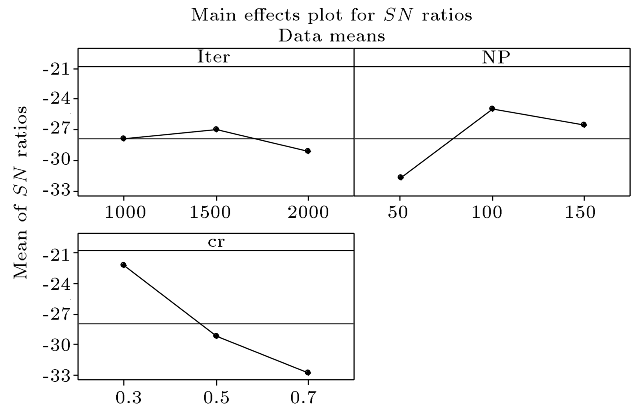

Figure 5. Average S/N ratio for Different Evolution (DE) parameters.

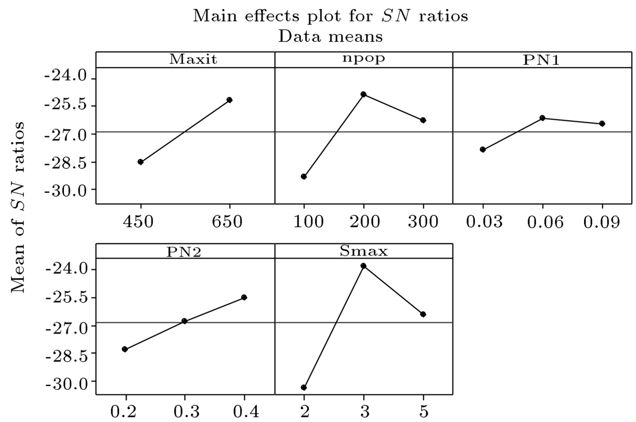

Figure 6. Average S/N ratio for Keshtel Algorithm (KA) parameters.

5. Numerical results

To implement the employed metaheuristics, a laptop using a system of Core 2 Dou-2.26 GHz processor is applied. All codes were written in C^{++} built in Microsoft Visual Studio. Based on this computer, first, 10 test problems are generated randomly in different

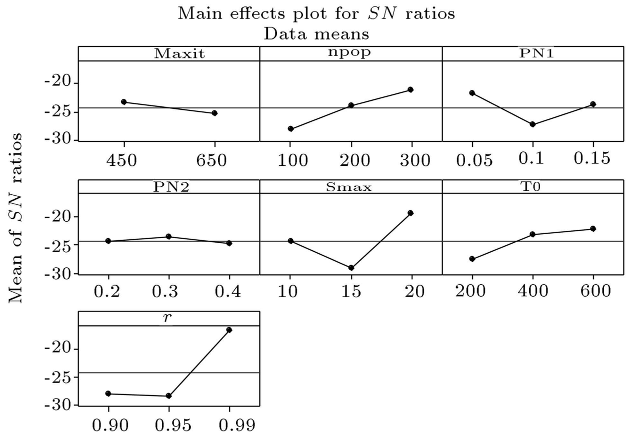

Figure 7. Average S/N ratio for KA-SA parameters.

Table 6. Parameters and their best level for each algorithm.

Algorithm	Parameter	Best level
SA	Maximum iteration ($Iter$)	1000
	Initial temperature (T_0)	100
	Temperature reduction rate (r)	0.99
	Number of sub iteration ($Subiter$)	75
DE	Maximum iteration ($Iter$)	1500
	Number of population (NP)	100
	Crossover constant (cr)	0.3
KA	Maximum iteration ($Maxit$)	650
	Population size (n_{pop})	200
	Percentage of N1 Keshtel (PN_1)	0.06
	Percentage of N2 Keshtel (PN_2)	0.4
	Maximum Swirling ($S_{\max}$)	3
KA-SA	Maximum iteration ($Maxit$)	450
	Population size (n_{pop})	300
	Percentage of N_1 Keshtel (PN_1)	0.05
	Percentage of N_2 Keshtel (PN_2)	0.3
	Maximum Swirling ($S_{\max}$)	20
	Initial temperature (T_0)	600
	Temperature reduction rate (r)	0.99

scales. The required time for truck changeover equals 75 per time and the needed time to transfer products from the receiving dock to the shipping dock equals 100 per time. Both loading and unloading time for all products are the same and equal 1 per time. Information related to these 10 problems is shown in Table 7.

The due date for each shipping truck is obtained through a uniform distribution, according to the following equation:

$$DD_j = uniform\left[\sum_{k=1}^{N}(s_{jk}) + V,\ \sum_{i=1}^{R}\sum_{j=1}^{S}\sum_{k=1}^{N} x_{ijk} + V + (S-1)D\right](1+\lambda). \qquad (44)$$

In the above equation, $\sum_{j=1}^{S}\sum_{k=1}^{N}(s_{jk}) + V$ is the required operation time for shipping truck j if all its needed products are ready in receiving dock, and:

$$\sum_{i=1}^{R}\sum_{j=1}^{S}\sum_{k=1}^{N} x_{ijk} + V + (D-1)S,$$

is the required operation time for all shipping trucks if their needed products are ready in the receiving dock. λ is a random number $\in$ uniform [0,0.5].

The lower bound and the upper bound of the time window for each shipping truck are acquired as follows:

$$e_j = uniform(0.8, 1) \times DD_j, \qquad (45)$$

$$l_j = uniform(1, 1.2) \times DD_j. \qquad (46)$$

The deadline for each shipping truck is obtained according to the following equation:

$$dl_j = uniform\left[l_j, 2\sum_{i=1}^{R}\sum_{j=1}^{S}\sum_{k=1}^{N} x_{ijk} + V + (D-1)S\right]. \qquad (47)$$

Algorithms were solved on a PC with an Intel core i5 processor. After parameters of the algorithms were tuned, each metaheuristic algorithm was run 30 times for each problem. In each run, the value of the objective function was recorded. The best and the mean of the acquired values through each algorithm are shown in Table 8. Having tuned the parameters of the algorithms, the diagram related to the average value obtained by each algorithm is shown in Figure 8. In order to show the results better and due to the difference between problem scales and the wide range of values in the objective function, all the results are converted to Robust Parameter Design (RPD). In this diagram, the vertical axis shows RPD, whose value can be obtained by the use of the following equation:

$$RPD = \frac{\text{Average solution} - \text{Best solution}}{\text{Best solution}}. \qquad (48)$$

The values obtained by the hybrid algorithm are

Table 7. Information of test problems.

Test problem	Number of receiving trucks	Number of shipping trucks	Number of product types	Number of perishable products	Total number of products
1	12	9	9	1	4040
2	12	11	12	1	6340
3	12	13	13	1	5440
4	14	11	13	1	5930
5	13	15	10	1	4627
6	15	16	9	2	3900
7	15	17	15	2	6281
8	14	18	14	2	6190
9	18	19	14	2	6981
10	20	19	16	2	8367

Table 8. Best and average value obtained by metaheuristic algorithms.

Set	KA		DE		SA		KA-SA	
	Best	Average	Best	Average	Best	Average	Best	Average
1	4522.5	5242.7	4522.5	4522.5	4522.5	5785.4	4522.5	4522.5
2	5243	6196.6	5243	5323.4	5612.4	6818.8	5243	5369.9
3	1333.3	1609.1	1333.3	1333.3	1333.3	1386.4	1333.3	1333.3
4	4019.4	4579.6	3898	3965.1	3898	4477	3898	3964.9
5	3746.6	5140.8	3552.3	3566.7	4135.2	4960.9	3552.3	3675.7
6	2022.1	2332	1903.2	2046.8	1903.2	1990. 8	1903	1938.1
7	2503.9	2829.4	2295	2493.4	2259.8	2660.6	2298	2376
8	2826.2	2912.9	1913.3	1993.2	1913.3	3229.2	1650.1	1882.2
9	2217	3056.9	1652.6	1924.6	1595	2325.3	1594.9	1838.4
10	3348.5	3620.9	1062.6	2035.2	2463	3360.2	1061.3	1872.3

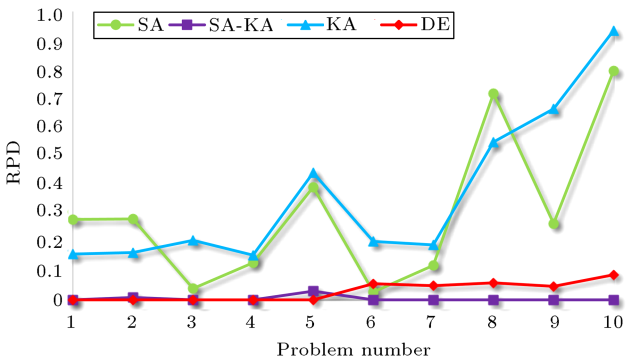

Figure 8. Average value obtained by the algorithm.

very desirable in most problems. In Figure 4, the value of PRD for the average results obtained by the hybrid algorithm is less than that obtained by other algorithms. This indicates that results obtained by the hybrid algorithm have good quality in all 30 trials for each problem. Therefore, the proposed hybrid algorithm is not only better than its original algorithm but also the DE as a well-known metaheuristic in the field.

6. Conclusion

The truck scheduling problem in a cross docking system is studied in this paper. A time window and a deadline are attributed to each shipping truck in this system. Products are also classified into two groups of perishable and imperishable products. Afterwards, a mathematical model is presented inspired by the available models. Three meta-heuristic algorithms and one proposed hybrid algorithm were used to solve the problem. The parameters of each algorithm were set using the Taguchi method. Ten test problems were generated to investigate the performance of the algorithms. Having set the parameters, each problem was performed thirty times by each algorithm. Consequences demonstrate that the suggested hybrid algorithm has a more desirable performance than the other algorithms.

To consider the main managerial implications of results, it can be concluded that considering time window limitations and different types of product as both perishable and imperishable, makes the model more practical. Solving this model is very important for managers to reach a robust answer in a logical time. Due to operational decisions of cross-docking systems, it is necessary to develop efficient solution algorithms to get a near-optimal solution in less time. Hence, the proposed hybrid metaheuristic algorithm gives this opportunity to a user to find a suitable and practical answer to the problem under study in large instances. For future studies, other metaheuristic and heuristic methods can be used to obtain better answers. Additionally, multiple receiving and shipping doors can be taken into account. Furthermore, a time window can be considered for arrival trucks. Based on the proposed hybrid metaheuristic algorithm, more in-depth analyses using standard benchmarks may be required to be explored. As such, other large-scale optimization problems can be employed to evaluate the proposed hybrid algorithm.

References

1. Fathollahi-Fard, A.M. and Hajiaghaei-Keshteli, M. "A stochastic multi-objective model for a closed-loop supply chain with environmental considerations", *Applied Soft Computing*, **69**, pp. 232–249 (2018).
2. Shafaei, R. and Mozdgir, A. "Master surgical scheduling problem with multiple criteria and robust estimation", *Scientia Iranica*, **26**(1), pp. 486–502 (2019).
3. Hajiaghaei-Keshteli, M. and Sajadifar, S.M. "Deriving the cost function for a class of three-echelon inventory system with N-retailers and one-for-one ordering policy", *The International Journal of Advanced Manufacturing Technology*, **50**(1–4), pp. 343–351 (2010).
4. Hajiaghaei-Keshteli, M., Sajadifar, S.M., and Haji, R. "Determination of the economical policy of a three-echelon inventory system with (R,Q) ordering policy and information sharing", *The International Journal of Advanced Manufacturing Technology*, **55**(5–8), pp. 831–841(2011).
5. Apte, U.M. and Viswanathan, S. "Effective cross docking for improving distribution efficiencies", *International Journal of Logistics*, **3**(3), pp. 291–302 (2000).
6. Zuluaga, J.P.S., Thiell, M., and Perales, R.C. "Reverse cross-docking", *Omega*, **66**, pp. 48–57 (2017).
7. Boysen, N. and Fliedner, M. "Cross dock scheduling: Classification, literature review and research agenda", *Omega*, **38**(6), pp. 413–422 (2010).
8. Van Belle, J., Valckenaers, P., and Cattrysse, D. "Cross-docking: State of the art", *Omega*, **40**(6), pp. 827–846 (2012).
9. Ladier, A.-L. and Alpan, G. "Cross-docking operations: Current research versus industry practice", *Omega*, **62**, pp. 145–162 (2016).
10. Yu, W., *Operational Strategies for Cross Docking Systems*, Digital Repository, Retrospective Theses and Dissertations, **413** (2002). https://doi.org/10.31274/rtd-180813-11026
11. Yu, W. and Egbelu, P.J. "Scheduling of inbound and outbound trucks in cross docking systems with temporary storage", *European Journal of Operational Research*, **184**(1), pp. 377–396 (2008).
12. Chen, F. and Lee, C.-Y. "Minimizing the makespan in a two-machine cross-docking flow shop problem", *European Journal of Operational Research*, **193**(1), pp. 59–72 (2009).
13. Boysen, N. "Truck scheduling at zero-inventory cross docking terminals", *Computers & Operations Research*, **37**(1), pp. 32–41 (2010).
14. Li, Y., Lim, A., and Rodrigues, B. "Crossdocking-JIT scheduling with time windows", *Journal of the Operational Research Society*, **55**(12), pp. 1342–1351 (2004).
15. Amini, A. and Tavakkoli-Moghaddam, R. "A bi-objective truck scheduling problem in a cross-docking center with probability of breakdown for trucks", *Computers and Industrial Engineering*, **96**, pp. 180–191 (2016).
16. Golshahi-Roudbaneh, A., Hajiaghaei-Keshteli, M., and Paydar, M.M. "Developing a lower bound and strong heuristics for a truck scheduling problem in a cross-docking center", *Knowledge-Based Systems*, **129**, pp. 17–38 (2017).
17. Serrano, C., Delorme, X., and Dolgui, A. "Scheduling of truck arrivals, truck departures and shop-floor operation in a cross-dock platform, based on trucks loading plans", *International Journal of Production Economics*, **194**, pp. 102–111 (2017).
18. Motaghedi-Larijani, A. and Aminnayeri, M. "Optimizing the number of outbound doors in the crossdock based on a new queuing system with the assumption of beta arrival time", *Scientia Iranica*, **25**(4), pp. 2282–2296 (2018).
19. Mohammadzadeh, M., Sahebjamnia, S., Fathollahi-Fard, A.M., and Hajiaghaei-Keshteli, M., "New approaches in metaheuristics to solve the truck scheduling problem in a cross-docking center", *International Journal of Engineering, Transaction B: Applications*, **31**(8), pp. 1258–1266 (2018).
20. Baniamerian, A., Bashiri, M., and Tavakkoli-Moghaddam, R. "Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking", *Applied Soft Computing*, **75**, pp. 441–460 (2019).
21. Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M., and Mirjalili, S. "Hybrid optimizers to solve a tri-level programming model for a tire closed-loop supply chain network design problem", *Applied Soft Computing*, **70**, pp. 701–722 (2018).
22. Samadi, A., Mehranfar, N., Fathollahi Fard, A.M., and Hajiaghaei-Keshteli, M. "Heuristic-based metaheuristics to address a sustainable supply chain network design problem", *Journal of Industrial and Production Engineering*, **35**(2), pp. 102–117 (2018).

23. Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M., and Tavakkoli-Moghaddam, R. "A bi-objective green home health care routing problem", *Journal of Cleaner Production*, **200**, pp. 423–443 (2018).

24. Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. "Optimization by simulated annealing", *Science*, **220**(4598), pp. 671–680 (1983).

25. Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M., and Mirjalili, S., "Multi-objective stochastic closed-loop supply chain network design with social considerations", *Applied Soft Computing*, **71**, pp. 505–525 (2018).

26. Storn, R. and Price, K., *Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces*, ICSI Berkeley, **3** (1995).

27. Fard, A.M.F. and Hajiaghaei-Keshteli, M. "A bi-objective partial interdiction problem considering different defensive systems with capacity expansion of facilities under imminent attacks", *Applied Soft Computing*, **68**, pp. 343–359 (2018).

28. Hajiaghaei-Keshteli, M. and Aminnayeri, M. "Solving the integrated scheduling of production and rail transportation problem by Keshtel algorithm", *Applied Soft Computing*, **25**, pp. 184–203 (2014).

29. Hajiaghaei-Keshteli, M. and Aminnayeri, M. "Keshtel Algorithm (KA); a new optimization algorithm inspired by Keshtels' feeding", In *Proceeding in IEEE Conference on Industrial Engineering and Management Systems*, pp. 2249–2253 (2013).

30. Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M., and Tavakkoli-Moghaddam, R. "A lagrangian relaxation-based algorithm to solve a home healthcare routing problem", *International Journal of Engineering*, **31**(10), pp. 1734–1740 (2018).

31. Taguchi, G., *Introduction to Quality Engineering: Designing Quality into Products and Processes*, White Plains, Asian Productivity Organization/UNIPUB, USA (1986).

32. Cheraghalipour, A., Hajiaghaei-Keshteli, M., and Paydar, M.M. "Tree Growth Algorithm (TGA): A novel approach for solving optimization problems", *Engineering Applications of Artificial Intelligence*, **72**, pp. 393–414 (2018).

33. Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M., and Tavakkoli-Moghaddam, R. "The Social Engineering Optimizer (SEO)", *Engineering Applications of Artificial Intelligence*, **72**, pp. 267–293 (2018).

34. Chen, F. and Song, K. "Minimizing makespan in two-stage hybrid cross docking scheduling problem", *Computers & Operations Research*, **36**(6), pp. 2066–2073 (2009).

35. Soltani, R. and Sadjadi, S.J. "Scheduling trucks in cross-docking systems: A robust meta-heuristics approach", *Transportation Research Part E: Logistics and Transportation Review*, **46**(5), pp. 650–666 (2010).

36. Boysen, N., Fliedner, M., and Scholl, A. "Scheduling inbound and outbound trucks at cross docking terminals", *OR Spectrum*, **32**(1), pp. 135–161 (2010).

37. Forouharfard, S. and Zandieh, M. "An imperialist competitive algorithm to schedule of receiving and shipping trucks in cross-docking systems", *The International Journal of Advanced Manufacturing Technology*, **51**(9–12), pp. 1179–1193 (2010).

38. Larbi, R., Alpan, G., Baptiste, P., and Penz, B. "Scheduling cross docking operations under full, partial and no information on inbound arrivals", *Computers and Operations Research*, **38**(6), pp. 889–900 (2011).

39. Arabani, A.B., Ghomi, S.F., and Zandieh, M. "Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage", *Expert systems with Applications*, **38**(3), pp. 1964–1979 (2011).

40. Shakeri, M., Low, M.Y.H., Turner, S.J., and Lee, E.W. "A robust two-phase heuristic algorithm for the truck scheduling problem in a resource-constrained cross-dock", *Computers & Operations Research*, **39**(11), pp. 2564–2577 (2012).

41. Berghman, L., Briand, C., Leus, R., and Lopez, P. "The truck scheduling problem at cross-docking terminals", *Paper Presented at the International Conference on Project Management and Scheduling* (PMS 2012) (2012).

42. Davoudpour, H., Hooshangi-Tabrizi, P., and Hoseinpour, P. "A genetic algorithm for truck scheduling in cross docking systems", *Journal of American Science*, **8**(2), pp. 96–99 (2012).

43. Sadykov, R. "Scheduling incoming and outgoing trucks at cross docking terminals to minimize the storage cost", *Annals of Operations Research*, **201**(1), pp. 423–440 (2012).

44. Boysen, N., Briskorn, D., and Tschöke, M. "Truck scheduling in cross-docking terminals with fixed outbound departures", *OR Spectrum*, **35**(2), pp. 479–504 (2013).

45. Van Belle, J., Valckenaers, P., Berghe, G.V., and Cattrysse, D. "A tabu search approach to the truck scheduling problem with multiple docks and time windows", *Computers & Industrial Engineering*, **66**(4), pp. 818–826 (2013).

46. Bjelić, N., Popović, D., and Ratković, B. "Genetic algorithm approach for solving truck scheduling problem with time robustness", Paper Presented at the *Proceedings of the 1st Logistics International Conference*, LOGIC (2013).

47. Joo, C.M. and Kim, B.S. "Scheduling compound trucks in multi-door cross-docking terminals", *The International Journal of Advanced Manufacturing Technology*, **64**(5–8), pp. 977–988 (2013).

48. Konur, D. and Golias, M.M. "Cost-stable truck scheduling at a cross-dock facility with unknown truck arrivals: A meta-heuristic approach", *Transportation Research Part E: Logistics and Transportation Review*, **49**(1), pp. 71–91 (2013).

49. Ladier, A. and Gülgün, A. "Scheduling truck arrivals and departures in a crossdock: Earliness, tardiness and storage policies", *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management* (IESM) IEEE (2013).

50. Ladier, Anne-Laure, and Gülgün Alpan. "Crossdock truck scheduling with time windows: earliness, tardiness and storage policies", *Journal of Intelligent Manufacturing*, **29**(3), pp. 569–583 (2018).

51. Madani-Isfahani, M., Tavakkoli-Moghaddam, R., and Naderi, B. "Multiple cross-docks scheduling using two meta-heuristic algorithms", *Computers & Industrial Engineering*, **74**, pp. 129–138 (2014).

52. Amini, A., Tavakkoli-Moghaddam, R., and Omidvar, A. "Cross-docking truck scheduling with the arrival times for inbound trucks and the learning effect for unloading/loading processes", *Production & Manufacturing Research*, **2**(1), pp. 784–804 (2014).

53. Mohtashami, A., Tavana, M., Santos-Arteaga, F.J., and Fallahian-Najafabadi, A. "A novel multi-objective meta-heuristic model for solving cross-docking scheduling problems", *Applied Soft Computing*, **31**, pp. 30–47 (2015).

54. Khalili-Damghani, K., et al. "A customized genetic algorithm for solving multi-period cross-dock truck scheduling problems", *Measurement*, **108**, pp. 101–118 (2017).

55. Wisittipanich, W. and Hengmeechai, P. "Truck scheduling in multi-door cross docking terminal by modified particle swarm optimization", *Computers & Industrial Engineering*, **113**, pp. 793–802 (2017).

Biographies

Amir Golshahdi-Roudbaneh received his MS degree in Industrial Engineering from the University of Science and Technology, Mazandaran. His research interests include scheduling, cross-docking, and combinatorial optimization.

Mostafa Hajiaghaei-Keshteli is a faculty member in the Department of Industrial Engineering at the University of Science and Technology, Mazandaran, Iran. He received his PhD in Industrial Engineering from Amirkabir University of Technology, Tehran. His research interests include sustainable logistics, supply chain network design, and combinatorial optimization.

Mohammad Mahdi Paydar is Associate Professor in the Department of Industrial Engineering at the Babol Noshirvani University of Technology, Babol, Iran. He received his PhD in Industrial Engineering from Iran University of Science and Technology in 2014. His research interests include supply chain design, operations research, multi-objective optimization, and cellular manufacturing. He has published more than 70 papers in international journals and conferences, including expert systems with applications, computers and mathematics with applications, computers and operations research, Journal of Cleaner Production, computers and industrial engineering, knowledge-based systems, Scientia Iranica, applied mathematical modelling etc.