J Intell Manuf (2018) 29:1155–1170
https://doi.org/10.1007/s10845-015-1160-3

Modeling truck scheduling problem at a cross-dock facility through a bi-objective bi-level optimization approach

Fateme Heidari[1] · **Seyed Hessameddin Zegordi**[2] · **Reza Tavakkoli-Moghaddam**[3]

Received: 19 December 2014 / Accepted: 12 October 2015 / Published online: 11 November 2015

Abstract Uncertainty and non-deterministic nature of the real world makes planning and scheduling in cross-docks a very complicated task for decision makers. These constant changes that happen all the time, often, lead to an increase in costs and/or a decrease in efficiency. Most of the uncertainty in cross-docks is caused by un-known truck arrival times. In this study we address the problem of scheduling incoming and outgoing trucks at a cross-dock facility, when vehicle arrival times are unknown, through a cost-stable scheduling strategy. Two meta-heuristics, MODE and NSGA-II, are used for solving the designed sample problems and are compared with a random search based genetic algorithm existing in the literature. Finally, performance of each algorithm is measured and analyzed using four metrics: quality, spacing, diversification and mean ideal distance. The results indicate that the proposed model MODE algorithm performs better in comparison with the other two methods.

Keywords Cross-dock facilities · Supply chain management · Unknown arrival time · Scheduling · Bi-objective bi-level optimization

✉ Seyed Hessameddin Zegordi
zegordi@modares.ac.ir

[1] Industrial Engineering of Tarbiat Modares University, Tehran, Iran

[2] Department of Industrial Engineering, Tarbiat Modares University, Tehran, Iran

[3] Department of Industrial Engineering, University of Tehran, Tehran, Iran

Introduction

In recent years, supply chain network design has become significantly important due to the rising competition among global markets. Organizations must improve customer service quality while reducing costs and increasing profits (Altiparmak et al. 2009). Any supply chain network consists of three main stages: procurement, production and distribution; each involves many facilities. Distribution centers play a crucial role in distribution stage. Cross-dock facilities that are recently taken into consideration in many industries are in fact, consolidation points in a distribution network. Cross docking is a warehouse management concept in which, items that are delivered to a warehouse by inbound trucks are immediately sorted out, reorganized based on customer demands, routed and loaded into outbound trucks for delivery to customers without the items being actually held in inventory at the warehouse. If any item is held in storage, it is usually for a brief period of time that is generally <24 h (Yu and Egbelu 2008) (Fig. 1).

While cross-docking reduces costs and creates new opportunities by omitting storing process, many organizations are still not using this strategy. There are many decision makings required in a cross-dock facility regarding the time window a market wants to plan for operational levels up to strategic levels (Van Belle et al. 2012). In this study we simultaneously address two operational problems; truck scheduling and truck allocation to cross-dock doors. Since truck arrival times are non-deterministic, for each truck, a time window is considered based on cross-dock operator's experience. Using the gathered information, allocation of trucks to doors and order of trucks in queues are determined. As mentioned earlier, uncertain arrival times cause less efficiency and more costs in a cross-dock facility; therefore, our proposed model is aimed to schedule and allocate

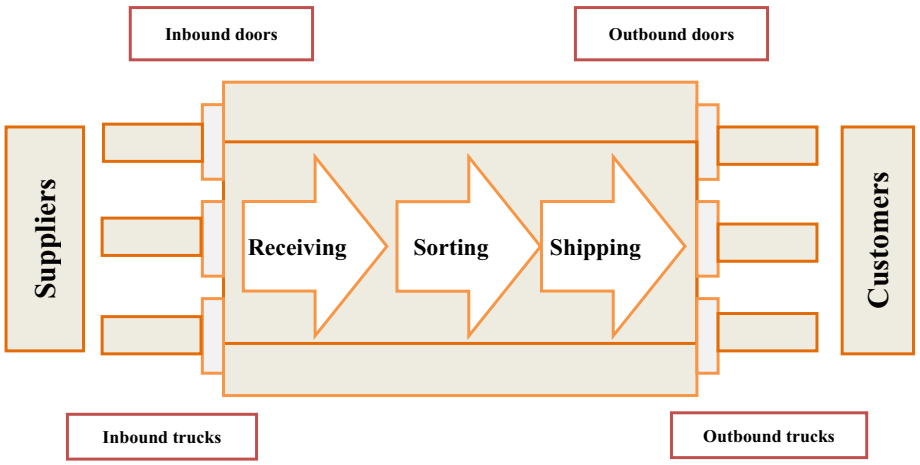


Fig. 1 Items flow in a cross-dock

trucks in such way that daily costs of cross-docking be stabilized.

Related work

Cross-docking is a relatively new logistic strategy; so there is still not a coherent literature about it. In fact, up until 2006 there has not been a single paper published on truck scheduling in cross-dock facilities. In comparison with traditional point to point distribution, cross-docking requires more transportation; this results in a slower distribution process. While keeping the temporary storage low in a cross-dock facility, to guarantee on-time delivery, a highly coordinated system among inbound and outbound trucks is necessary. Recently, some scheduling methods have been proposed for solving truck scheduling problems (Boysen and Fliedner 2010). Chen and Lee (2009) addressed the problem of incoming and outgoing truck sequence scheduling. Due to the limited space of Cross-dock facilities in their paper, it is assumed that at each moment, only one outgoing truck is available, which starts loading after preparation of all the shipments is done. Under the above circumstances, unloading, sorting and loading are pretty much effective on cross-dock's performance in a supply chain network. Chen and Song (2009) also proposed a method for bi-level hybrid cross-dock scheduling problem which is a general case of the problem presented in their previous paper. The former problem they addressed had two stages, and in each stage, there was a single machine while the latter problem consists of two stages that at least one of them has more than one parallel machine.

Boysen (2010) discussed a special case of cross-dock truck scheduling problem in food industry. The constant need for cooling the products in this industry makes it impossible to have a temporary storage in the terminal; therefore, all the shipments are immediately loaded on refrigerated trucks. Boysen et al. (2012) also addressed the problem of truck scheduling when outgoing truck schedules are predetermined, arrival times are assigned to incoming trucks and inbound trucks are assigned to doors. Their presented model objective was to minimize the lost benefit which was defined as the shipment's value. To solve the problem, a heuristic method called decomposition procedure and simulated annealing were implemented. Liao et al. (2013) addressed the problem of simultaneously allocating and sequence scheduling of incoming trucks while outgoing truck schedules are considered constant. Six different meta-heuristics (simulated annealing, tabu search, ant colony optimization, differential evolution and two different hybrid differential evolution algorithms) are used to solve the problem. Kuo (2013) proposed a method for optimizing trucks allocations to cross-dock doors as well as trucks sequence, in order to minimize total operation time. The obtained solutions are improved using variable neighborhood search. Four different simulated annealing algorithms are presented for evaluating and comparing the results.

Wang et al. (2014) developed a model that determines the strategy of renting and owning trucks in integration with internal truck scheduling and storage allocation problems in container terminals. To solve this complicated problem, they proposed a 2-level heuristic approach, in which the integration problem is decomposed into two levels. The results of using model show that even if the using cost of owned yard trucks is much lower than the cost of rented yard tucks, terminal companies should not purchase too many trucks when the purchasing price is too high. Wang et al. (2015) also integrated Yard truck scheduling and storage allocation problems a whole and minimize the weighted summation of total delay and total yard trucks travel time. To solve the problem, a genetic algorithm (GA) were implemented.

Golias et al. (2010) considered cross-dock scheduling problem with two objectives; minimizing total operational costs and outgoing trucks costs. A heuristic algorithm is used

for solving the problem. Boloori Arabani et al. (2011) also addressed the problem of cross-dock scheduling with two objectives. The first is to minimize the maximum completion time and the second is to minimize the total delay cost of outgoing trucks. Three different multi-objective optimization algorithms that are based on sub-population (SPGA-II, SPPSO-II, SPDE-II) are used for solving the presented problem. Konur and Golias (2013) presented a cost-stable, bi-level, bi-objective scheduling strategy to minimize the average total service cost. Genetic algorithms and simulations are used to solve the problem. To simplify the existing models in the literature many unrealistic assumptions have been made. In most of the previous works there are only one inbound and one outbound door. Cross-docks temporary stores are assumed to have infinite capacity while there are space limitations in real world. Although loading and unloading time is different for each truck (depending on type and amount of items existing in one truck, manpower and the provided facilities on each door) these times are considered the same in most of the existing researches.

More importantly, considering all the information at hand deterministic, most of the times, deterministic models are used for cross-dock scheduling in the literature. Cross-dock Scheduling is often carried out assuming deterministic truck arrival times, availability of all trucks at the beginning and known truck arrival sequences. There are so many unpredicted elements such as truck failures, traffic, climatic factors etc., which might affect truck delivery systems. As a matter of fact, the dynamic and un-certain nature of the real world makes planning and scheduling in cross-dock facilities a challenging job which requires more flexible models indeed.

The model presented in thisstudy takes the un-deterministic truck arrival times into account and solves truck scheduling problem at a cross-dock while stabilizing total cost of store and maximizing efficiency. We assume more than one door for the store and the problems of allocating trucks to doors and determining their sequence on each door are solved simultaneously. Only a few papers on cross-dock scheduling addressed bi-objective, bi-level problems. Our proposed model is the same model used in Konur and Golias (2013). Their paper only concentrated on incoming trucks, while in this study by expanding the existing model, outgoing trucks are considered in the model as well. The rest of this paper is organized as follows. In section "Mathematical model", bi-objective bi-level truck scheduling problem and the corresponding model are described thoroughly. In section "Bi-level formulation", meta-heuristic approaches that are used for solving the proposed model are discussed and finally result analysis, conclusion and future works are presented in sections "Computational results" and "Conclusion", respectively.

Mathematical model

The purpose of this research is to present a model for truck scheduling at a cross-dock facility using bi-objective, bi-level modeling approach. To do so, some assumptions are made. In the model presented in this paper there are more than one inbound and outbound doors, each door is exclusively allocated to incoming or outgoing trucks, preemption is not allowed, arrival times are unknown and non-deterministic and each event happens in a time window, process times are different from one another, there is no due date, using temporary storage is not allowed, number of trucks allocated to each door is not limited, indoor transportation time is constant, each truck exits the cross-dock when it is full, and finally each incoming product is allocated to a certain outgoing truck. In our presented model we use the following sets:

I_1: Set of inbound doors
I_2: Set of outbound doors
J_1: Set of incoming trucks
J_2: Set of outgoing trucks

and also the following parameters:

A_j (Arrival time of truck j), $j \in J_1, J_2$
p_{ij} (Total time required for loading or unloading truck j at door i), $i \in I_1, I_2$, $j \in J_1, J_2$
V_j (Total time required to sort truck j's unloaded shipment), $j \in J_1$
d_j (Penalty cost per unit time for making truck j wait), $j \in J_1, J_2$
w_{ij} (Process cost of truck j on door i), $i \in I_1, I_2$, $j \in J_1, J_2$

$$z_{aj} \begin{cases} 1 \text{ If a product from incoming truck } a \text{ be transferred to outgoing truck } j \\ 0 \text{ Otherwise} \end{cases}, a \in J_1, j \in J_2$$

Finally the following variables are used in the model. The main variables consist of:

$$x_{ij} \begin{cases} 1 \text{ If truck } j \text{ is assigned to door } i \\ 0 \text{ Otherwise} \end{cases}, i \in I_1, I_2, j \in J_1, J_2$$

$$y_{ab} \begin{cases} 1 \text{ If truck } a \text{ is the immidiate predecessor of truck } b \\ 0 \text{ Otherwise} \end{cases}, a, b \in J_1, J_2$$

The auxiliary variables are:
t_j (process start time of truck j), $j \in J_1, J_2$

$$f_j \begin{cases} 1 \text{ If process of truck } j \text{ on its assigned door happens at first} \\ 0 \text{ Otherwise} \end{cases}, j \in J_1, J_2$$

$$l_j \begin{cases} 1 \text{ If process of truck } j \text{ on its assigned door happens at last} \\ 0 \text{ Otherwise} \end{cases}, j \in J_1, J_2$$

Let X_1 be a $m_1 \times n_1$ matrix of x_{ij}'s ($i \in I_1$, $j \in J_1$) for incoming trucks and inbound doors and X_2 be a $m_2 \times n_2$ matrix of x_{ij}'s ($i \in I_2$, $j \in J_2$) for outgoing trucks and outbound doors. Let Y_1 be a $m_1 \times m_1$of y_{ab}'s ($a, b \in J_1$) and Y_2 be a $m_2 \times m_2$ of y_{ab}'s ($a, b \in J_2$). A schedule is then defined by pairs of (X_1, Y_1) and (X_2, Y_2). Let A_1 be a $1 \times m_1$ vector of A_j's ($j \in J_1$) and A_2 be a $1 \times m_2$ vector of A_j's($j \in J_2$). At first we assume that truck arrival times are predetermined. For a given A_1 and A_2, a schedule (X_1, Y_1), (X_2, Y_2), has a total service cost which includes of truck process cost and truck waiting time cost.

Let $w_{ij} = C_i \times p_{ij}$ in which C_i is process cost per unit of time on door i. Also T_1 is a $1 \times m_1$ vector of t_j's ($j \in J_1$) and T_2 is a $1 \times m_2$ of t_j's ($j \in J_1$). $d_j(t_j - A_j)$ is waiting time cost of truck j. Each truck depending on its distinct properties and also the Just in time requirements has different cost per unit of time. If truck arrival times are deterministic, cross-dock operator must schedule trucks in a way that minimizes total service cost or $TCS(X, Y, T, A)$ in which X = (X_1, X_2), Y = (Y_1, Y_2), T = (T_1, T_2) and A = (A_1, A_2).

$$TSC(X, Y, T, A) = \sum_{i \in I_1, I_2} \sum_{j \in J_1, J_2} w_{ij} x_{ij} + \sum_{j \in J_1, J_2} d_j (t_j - A_j) \quad (1)$$

In Eq. (1) if $C_i = 1$ and $d_i = 1$ then $TCS(X, Y, T, A)$ shows total service time. Assuming deterministic truck arrival times, *Deterministic Scheduling Problem* (DSP) is formulated as follows:

$$\text{(DSP)} \min_{(X,Y,T)} \quad TSC(X, Y, T, A) = \sum_{i \in I_1, I_2} \sum_{j \in J_1, J_2} w_{ij} x_{ij} + \sum_{j \in J_1, J_2} d_j \left(t_j - A_j\right)$$

s.t.

$$\sum_{i \in I_1} x_{ij} = 1 \quad \forall j \in J_1 \quad (2)$$

$$\sum_{i \in I_2} x_{ij} = 1 \quad \forall j \in J_2 \quad (3)$$

$$f_b + \sum_{a \in J_1 \neq b} y_{ab} = 1 \quad \forall b \in J_1 \quad (4)$$

$$f_b + \sum_{a \in J_2 \neq b} y_{ab} = 1 \quad \forall b \in J_2 \quad (5)$$

$$l_a + \sum_{b \in J_1 \neq a} y_{ab} = 1 \quad \forall a \in J_1 \quad (6)$$

$$l_a + \sum_{b \in J_2 \neq a} y_{ab} = 1 \quad \forall a \in J_2 \quad (7)$$

$$f_a + f_b \leq 3 - x_{ia} - x_{ib} \quad \forall i \in I_1, a, b \in J_1, a \neq b \quad (8)$$

$$l_a + l_b \leq 3 - x_{ia} - x_{ib} \quad \forall i \in I_1, a, b \in J_1, a \neq b \quad (9)$$

$$f_a + f_b \leq 3 - x_{ia} - x_{ib} \quad \forall i \in I_2, a, b \in J_2, a \neq b \quad (10)$$

$$l_a + l_b \leq 3 - x_{ia} - x_{ib} \quad \forall i \in I_2, a, b \in J_2, a \neq b \quad (11)$$

$$y_{ab} - 1 \leq x_{ia} - x_{ib} \leq 1 - y_{ab} \quad \forall i \in I_1, a, b \in J_1, a \neq b \quad (12)$$

$$y_{ab} - 1 \leq x_{ia} - x_{ib} \leq 1 - y_{ab} \quad \forall i \in I_2, a, b \in J_2, a \neq b \quad (13)$$

$$t_j \geq A_j \quad \forall j \in J_1 \quad (14)$$

$$t_j \geq \sum_{a \in J_1 \neq j} t_a y_{aj} + \sum_{i \in I_1} \sum_{a \in J_1 \neq j} p_{ia} x_{ia} y_{aj} \quad \forall j \in J_1 \quad (15)$$

$$t_j \geq A_j \quad \forall j \in J_2 \quad (16)$$

$$t_j \geq \sum_{a \in J_2 \neq j} t_a y_{aj} + \sum_{i \in I_2} \sum_{a \in J_2 \neq j} p_{ia} x_{ia} y_{aj} \quad \forall j \in J_2 \quad (17)$$

$$t_j \geq \left(t_a + \sum_{i \in I_1} p_{ia} x_{ia} + V_a \right) z_{aj} + A_j \left(1 - z_{aj}\right) \quad \forall a \in J_1, j \in J_2 \quad (18)$$

Equations (2) and (3) define the constraints ensuring single inbound and outbound door assignment for each truck. Equations (4) and (5) define the constraints guaranteeing that each truck either is processed as the first truck or has a successor. Equations (6) and (7) ensure that each truck either is processed as the last truck or has a predecessor. Equations (8)–(11) define the constraints restricting each door to have at most one first and at most one last truck. Equations (12) and (13) define the constraints restricting that a truck can only be processed immediately before or after another truck only if both trucks be on the same door. Equations (14)–(17) determines truck arrival times and Eqs. (14) and (16) ensure that process start time of each truck be after its arrival time. Finally, Eq. (18) defines the constraint indicating that outgoing trucks loading process can't start before completion of incoming trucks unloading process.

As mentioned earlier to solve DSP, A should be given. Since truck arrival times are subject to high variability, in this paper, a given time window is considered for arrival times of each truck at a cross-dock facility. But still the exact times of truck arrivals are not known in advance. That is, $A_j \in [A_j^l A_j^u]$ where A_j^l denotes the lower bound of truck j's arrival window and A_j^u denotes the upper bound of truck j's arrival window. We assume that the cross-dock operator knows each truck's arrival time window. Under unknown truck arrival times, the truck waiting costs associated with a given schedule are subject to uncertainty as well, whereas the total process times are fixed for the given schedule. The uncertainty in truck waiting times determines the range of the possible total service costs for a given schedule, i.e., the difference between the possible maximum and minimum total service costs. A low range schedule may imply high total

service costs. Therefore, in what follows, we formulate the cross-dock operator's problem to find a schedule which minimizes the average total service costs and the range of the total service costs as a bi-objective optimization problem:

$$
\begin{aligned}
R(X,Y,T) &= \left(\sum_{i\in I_1,I_2} \sum_{j\in J_1,J_2} w_{ij}x_{ij} \right. \\
&\left. + \max_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) \right) \\
&- \left(\sum_{i\in I_1,I_2} \sum_{j\in J_1,J_2} w_{ij}x_{ij} + \min_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) \right) \\
&= \max_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) - \min_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right)
\end{aligned} \tag{19}
$$

The first component of (19) is the maximum total service cost and the second component represents the minimum total service cost. If the objective function only minimizes the range of the total service costs, we might end up with a schedule which has high average service costs. Therefore, our second objective function tries to minimize the average total service cost defined as:

$$
\begin{aligned}
ATSC(X,Y,T) &= \sum_{i\in I_1,I_2} \sum_{j\in J_1,J_2} w_{ij}x_{ij} \\
&+ \frac{1}{2}\left(\max_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) + \min_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) \right)
\end{aligned} \tag{20}
$$

The first and second components of Eq. (20) are the total process costs and the arithmetic average of the possible maximum and minimum total waiting costs, respectively. The cross-dock bi-objective scheduling problem which is referred to as *Stable Scheduling Problem* (SSP) can be formulated as follows:

$$
\begin{aligned}
(\text{SSP}) \quad \min_{(X,Y,T)} \quad & ATSC(X,Y,T) \\
= & \sum_{i\in I_1,I_2} \sum_{j\in J_1,J_2} w_{ij}x_{ij} \\
& + \frac{1}{2}\left(\max_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right)\right. \\
& \left. + \min_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right)\right) \\
\min_{(X,Y,T)} \quad & R(X,Y,T) = \max_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) \\
& - \min_A \sum_{j\in J_1,J_2} d_j\left(t_j - A_j\right) \\
\text{s.t.} \quad & \text{Eqs.}(2-18)
\end{aligned}
$$

The first objective of SSP minimizes the average total service costs while the second objective minimizes the range of the total service costs. Constraints defined in Eqs. (2)–(18) have previously been explained.

Bi-level formulation

Objective functions of SSP, i.e., $ATSC(X, Y, T)$ and $R(X, Y, T)$, each has two optimization problems within themselves that have to be solved independently in another level. Therefore SSP must be reformulated as a bi-objective bi-level optimization problem. To do so, the following definitions are required: $A^{max}(X,Y)$ and $A^{min}(X,Y)$ represent truck arrival times while given a schedule (X, Y), total waiting cost has its maximum and minimum amount respectively. To determine $A^{max}(X,Y)$ and $A^{min}(X,Y)$ we have to solve another optimization problem called *Ranged Bound Problem* (RBP) in the second level:

$$
(\text{RBP}) \max_A / \min_A \quad \sum_{j\in J_1,J_2} d_j(t_j - A_j) \tag{21}
$$

s.t.

$$
A_j^l \le A_j \le A_j^u \quad \forall j \in J_1, J_2 \tag{22}
$$

$$
\sum_{a\in J_1\neq j} t_a y_{aj} + \sum_{i\in I_1} \sum_{a\in J_1\neq j} p_{ia}x_{ia}y_{aj} - A_j \le M\left(1 - z_j\right) \quad \forall j \in J_1 \tag{23}
$$

$$
A_j - \sum_{a\in J_1\neq j} t_a y_{aj} - \sum_{i\in I_1} \sum_{a\in J_1\neq j} p_{ia}x_{ia}y_{aj} \le M\left(z_j\right) \quad \forall j \in J_1 \tag{24}
$$

$$
\sum_{a\in J_1\neq j} t_a y_{aj} + \sum_{i\in I_1} \sum_{a\in J_1\neq j} p_{ia}x_{ia}y_{aj} + Mz_j \ge t_j \quad \forall j \in J_1 \tag{25}
$$

$$
A_j + M\left(1 - z_j\right) \ge t_j \quad \forall j \in J_1 \tag{26}
$$

$$
t_j \ge \sum_{a\in J_1\neq j} t_a y_{aj} + \sum_{i\in I_1} \sum_{a\in J_1\neq j} p_{ia}x_{ia}y_{aj} \quad \forall j \in J_1 \tag{27}
$$

$$
t_j \ge A_j \quad \forall j \in J_1 \tag{28}
$$

$$
\sum_{a\in J_2\neq j} t_a y_{aj} + \sum_{i\in I_2} \sum_{a\in J_2\neq j} p_{ia}x_{ia}y_{aj} - A_j \le M\left(1 - zz_j\right) \quad \forall j \in J_2 \tag{29}
$$

$$
A_j - \sum_{a\in J_2\neq j} t_a y_{aj} - \sum_{i\in I_2} \sum_{a\in J_2\neq j} p_{ia}x_{ia}y_{aj} \le M\left(zz_j\right) \quad \forall j \in J_2 \tag{30}
$$

$$A_j + M\left(2 - zz_j - xx_j\right) \geq t_j \quad \forall j \in J_2 \tag{31}$$

$$t_j \geq \sum_{a \in J_2 \neq j} t_a y_{aj} + \sum_{i \in I_2} \sum_{a \in J_2 \neq j} p_{ia} x_{ia} y_{aj} \quad \forall j \in J_2 \tag{32}$$

$$t_j \geq A_j \quad \forall j \in J_2 \tag{33}$$

$$\left(t_a + \sum_{i \in I_1} p_{ia} x_{ia} + V_a\right) z_{aj} + A_j\left(1 - z_{aj}\right) - A_j \leq M\left(1 - xx_j\right) \quad \forall a \in J_1, j \in J_2 \tag{34}$$

$$A_j - \left(t_a + \sum_{i \in I_1} p_{ia} x_{ia} + V_a\right) z_{aj} - A_j\left(1 - z_{aj}\right) \leq M\left(xx_j\right) \quad \forall a \in J_1, j \in J_2 \tag{35}$$

$$t_j - \sum_{a \in J_2 \neq j} t_a y_{aj} - \sum_{i \in I_2} \sum_{a \in J_2 \neq j} p_{ia} x_{ia} y_{aj} \leq M\left(1 - xx_j + zz_j\right) \quad \forall j \in J_2 \tag{36}$$

$$t_j \geq \left(t_a + \sum_{i \in I_1} p_{ia} x_{ia} + V_a\right) z_{aj} + A_j\left(1 - z_{aj}\right) \quad \forall a \in J_1, j \in J_2 \tag{37}$$

If the objective of RBP is maximization, then the optimal solution gives us $A^{max}(X, Y)$ and the objective function value at $A^{max}(X, Y)$ is the maximum truck waiting cost, which can be used to determine the upper bound of the range of the total service costs associated with schedule (X, Y). On the other hand if the objective of RBP is minimization, the optimal solution gives us $A^{min}(X, Y)$ and the objective function value at $A^{min}(X, Y)$ is the minimum truck waiting cost, which can be used to determine range of the total service costs lower bound, associated with schedule (X, Y). Equation (22) defines the constraint guaranteeing that the truck arrival time for each truck is within its given arrival time window. Equations (23)–(37) are used to define the loading or unloading process start times for each truck. Since RBP is optimized over A, the process start times of incoming and outgoing trucks should obey the followings:

For incoming trucks, there are two possible scenarios. If the assigned inbound door to a truck is idle when the truck arrives, the truck's process start time is equal to its arrival time. One the other hand, if the assigned inbound door to the truck is busy when the truck arrives, truck's process start time should be equal to the process finish time of its immediate predecessor. In the first case it means, truck j arrives after the process of its immediate predecessor, truck k, is finished such that $A_j \geq ft_k$. ft_k denotes finish time of truck k and is obtained by Eq. (38).

$$ft_k = \sum_{a \in J_1 \neq j} t_a y_{aj} - \sum_{i \in I_1} \sum_{a \in J_1 \neq j} p_{ia} x_{ia} y_{aj} \tag{38}$$

In this case, $t_j = A_j$. In particular, when $A_j \geq ft_k$, Eq. (23) enforces $z_j = 1$ and Eqs. (24) and (25) are valid. Also, Eq. (27) is valid by definition. When $z_j = 1$, it then follows from Eqs. (26) and (28) that $t_j = A_j$.

In the second case, when $A_j \leq ft_k$, we should have $t_j = ft_k$. When $A_j \leq ft_k$, Eq. (24) enforces $z_j = 0$ and Eqs. (23) and (26) are valid. Furthermore, Eq. (28) is valid by definition. When $z_j = 0$, it then follows from Eqs. (25) and (27) that $t_j = ft_k$, therefore, Eqs. (23)–(28) ensure that Eq. (39) holds.

$$t_j = \max\left\{A_j, \sum_{a \in J_1 \neq j} t_a y_{aj} - \sum_{i \in I_1} \sum_{a \in J_1 \neq j} p_{ia} x_{ia} y_{aj}\right\} \tag{39}$$

For outgoing trucks, four scenarios are possible. If the assigned outbound door to a truck is idle when the truck arrives and all the truck's shipment is ready for loading, the truck's process start time is equal to its arrival time. In fact in this case truck j arrives after, sorting of its shipment is done i.e. $Ct_j \leq A_j$, in which Ct_j according to Eq. (40) is:

$$Ct_j = \left(t_a + \sum_{i \in I_1} p_{ia} x_{ia} + V_a\right) z_{aj} + A_j\left(1 - z_{aj}\right) \tag{40}$$

If the shipment is ready for loading when outgoing truck j arrives, $xx_j = 1$ according to Eq. (34). Equation (35) holds as well and therefore $t_j = A_j$. When $A_j \geq ft_k$, Eq. (29) forces $zz_j = 1$ and Eq. (30) is valid by definition. $xx_j = 1$ and $zz_j = 1$ force Eq. (37) to be valid. It then follows from Eqs. (31) and (33) that $t_j = A_j$.

If all the truck's shipment is ready for loading but the outbound door to the truck is busy when the truck arrives, truck's process start time is equal to the process finish time of its immediate predecessor i.e. $t_j = ft_k$. In this case, $Ct_j \leq A_j$ and in Eq. (34), $xx_j = 1$. Equation (35) is valid by definition. Also $A_j \leq ft_k$ which indicates that $zz_j = 0$ in Eqs. (30) and (29) is also valid. $xx_j = 1$ and $zz_j = 0$ force Eq. (31) to be valid. It then follows from Eqs. (32) and (36) that $t_j = ft_k$.

If the assigned outbound door to the truck is idle when the truck arrives but truck's shipment is not ready for loading, truck's process start time is equal to the time required for its shipment to get sorted, stored and staged i.e. $t_j = Ct_j$. In this case, $Ct_j \geq A_j$ and in Eq. (35), $xx_j = 0$. Equation (34) is valid by definition. Also $A_j \geq ft_k$ which indicates that $zz_j = 1$ in Eqs. (29) and (30) is also valid. $xx_j = 0$ and $zz_j = 1$ force Eqs.(31) and (36) to be valid and therefore, $t_j = Ct_j$.

If the assigned outbound door to the truck is busy when the truck arrives and the truck's shipment is not ready for loading, the truck's process start time, t_j, is equal to $\max\{Ct_j, ft_k\}$. When $Ct_j \geq A_j$, $xx_j = 0$ according to Eqs. (35) and (34) is

valid by definition. Also when $A_j \leq ft_k, zz_j = 0$ according to Eqs. (30) and (29) is also valid. $xx_j = 0$ and $zz_j = 0$ force Eqs.(31) and (36) to be valid and considering other constraints we have, $t_j \geq Ct_j$ and $t_j \geq ft_k$ which means $t_j = \max\{Ct_j, ft_k\}$.

In the second level, minimizing or maximizing the total waiting costs, is attained by only changing A_j values. We use T^{max} and T^{min} to denote the m-vectors of t_j values for truck arrivals defined by $A^{max}(X, Y)$ and $A^{min}(X, Y)$, respectively. Next, we represent SSP as a bi-objective, bi-level optimization problem using the definitions of $A^{max}(X, Y)$, $A^{min}(X, Y)$, T^{max} and T^{min}. In particular, SSP is equal to:

(SSP-2)

$$\min_{(X,Y,T)} \quad ATSC(X, Y, T) = \sum_{i\in I_1,I_2} \sum_{j\in J_1,J_2} w_{ij}x_{ij} + \frac{1}{2}\left(\sum_{j\in J_1,J_2} d_j \left(t_j^{\max} - A_j^{\max}\right) + \sum_{j\in J_1,J_2} d_j \left(t_j^{\min} - A_j^{\min}\right)\right)$$

$$\min_{(X,Y,T)} \quad R(X, Y, T) = \sum_{j\in J_1,J_2} d_j \left(t_j^{\max} - A_j^{\max}\right) - \sum_{j\in J_1,J_2} d_j \left(t_j^{\min} - A_j^{\min}\right)$$

s.t.

$$Eqs.(2-19),$$

$$\left(A^{\max}, T^{\max}\right) = \arg\max \left\{ \sum_{j\in J_1,J_2} d_j \left(t_j - A_j\right) \quad s.t. Eqs.(23-38) \right\}$$

$$\left(A^{\min}, T^{\min}\right) = \arg\min \left\{ \sum_{j\in J_1,J_2} d_j \left(t_j - A_j\right) \quad s.t. Eqs.(23-38) \right\}$$

Solution methodology/solving the mathematical model

Linear bi-level, single-objective optimization problems (when both the upper and lower levels are linear problems) are discussed to be NP-hard (Hansen et al. 1992), hence the problem of solving a bi-objective, bi-level optimization problem which is discussed in this paper can be categorized as an NP-hard problem as well (Konur and Golias 2013). Since none of the existing mathematical modeling softwares are capable of solving a bi-objective bi-level problem, each level of such problems are solved separately using a commercial solver such as CPLEX (considering the fact that each level of the main problem is a mixed integer linear problem itself). Complexity of these problems, the fact that they are operational decision making problems and require to be solved on a daily basis in the real world (which shows the necessity of short computational time) leaves us no choice but to use metaheuristic algorithms. There are many evolutionary algorithms in the literature that are designed for finding non-dominated solutions of multi-objective decision problems such as NSGA-II (non-dominated sorting genetic algorithm). Given a proper criterion for comparing the problem's objectives, other algorithms designed for solving single objective decision problems such as MODE (which is also used in this paper) can also be used to solve bi-objective and multi-objective problems as well.

Our presented model is an extension to Konur and Golias's so in order to validate and evaluate the performance of the implemented solving methods in this paper, our model is also solved by the GASH algorithm presented in Konur and Golias's paper in 2013 and the final results are compared. The main difference of GASH with NSGA-II and MODE is that the latter two algorithms improve the solutions in the first level and then the improved solutions are sent to the second level. Konur and Golias (2013) expressed two propositions in case of scheduling merely inbound trucks. These two propositions are extended in this paper to handle the simultaneous scheduling of inbound and outbound trucks which can be the basis for solving RBP by metaheuristic algorithms.

Proposition 1 *Without loss of generality, suppose that the trucks to be served at a given inbound door i are 1, 2,. . ., k such that truck 1 is to be served before truck 2, truck 2 is to be served before truck 3, and so on. Then RBP at door i is minimized by setting*

$$A_j = \min\left\{A_j^u, \max\left\{ft_{j-1}, A_j^l\right\}\right\} \quad \forall j \in J_1 \tag{41}$$

When $ft_0 = 0$. *And for the outbound trucks we also have*

$$A_j = \min\left\{A_j^u, \max\left\{Ct_j, ft_{j-1}, A_j^l\right\}\right\} \quad \forall j \in J_2. \tag{42}$$

When $ft_0 = 0$. *In* (42) Ct_j indicates the preparation time of the unloaded items from incoming trucks that are about to be loaded on outgoing truck j and is calculated as follows:

$$Ct_j = \max\left\{(ft_a + V_a)\, z_{aj}\right\} \quad \forall a \in J_1, j \in J_2 \tag{43}$$

In (43), ft_a is the completion time of unloading truck a which is attained by (44)

$$ft_a = t_a + \sum_{i\in I_1} p_{ia}x_{ia} \quad \forall a \in J_1 \tag{44}$$

Proposition 2 *Without loss of generality, suppose that the trucks to be served at a given inbound door i are 1, 2, . . ., k such that truck 1 is to be served before truck 2, truck 2 is to be served before truck 3, and so on. Then there exists an optimal solution to RBP at door i* $(i \in I_1, I_2)$ *with the objective of maximization such that*

$$A_j \in \left\{A_j^u, A_j^l\right\} \quad j \in J_1, J_2 \quad \forall j \in \{1, 2, , 3, ..., k\} \quad (45)$$

In both evolutionary algorithms NSGA-II and MODE, the above propositions are used to solve the second level of the problem. In the following sections these two algorithms and their solving strategies are thoroughly explained.

Non-dominated sorting genetic algorithm (NSGA-II)

NSGA is one of the first evolutionary algorithms developed for solving multi-objective decision problems. NSGA-II is the second version of this algorithm that has the advantages of less computational complexity and better search through the search space by calculating the crowding distance. To solve the presented model, the steps of this algorithm are taken according to the existing works in the literature with the exception of using another single objective GA for optimizing the first level.

Single objective genetic algorithm for the first level improvement

In each chromosome of the GA, according to trucks arrival time and by implementing a single objective GA the quality of the first created solution is improved to attain a better schedule with fewer costs. This schedule consists of the appropriate allocation of trucks to doors and their suitable order of service. The fitness value in this level is in fact value of the DSP objective function which is calculated for each of the created chromosomes independently and at last information gathered by the initial population in the first level is sent to the second level for calculating ATSC and R. Mutation operator in this GA is the same as the one described in the following sections.

Solution representation

The created chromosomes in this algorithm have two separate rows. In the first row, numbers $1, \ldots, n_1$ (numbers of inbound doors) are randomly created m_1 (number of incoming trucks) times and in the second row numbers $1, \ldots, m_1$ are randomly arranged hence numbers of allocated trucks to each door is determined. Trucks' order of service on each door is based on FCFS (first come first serve) rule. Chromosome's representation is shown in Fig. 2. For outbound doors and trucks the same procedure is followed.

Mutation and cross-over operators

Due to the problem's nature and the chromosome structure that is used, the cross-over operator is not implemented; repair procedure of the infeasible solutions created by cross-over would be difficult and time consuming. Swap mutation is one of the operators used for mutation. On each door, a random truck is chosen and exchanged with the random truck chosen from the next door. The selected truck from the last door is also exchanged with the one chosen from the first door. Figure 2 shows a chromosome in which 3 doors (inbound/outbound) and 9 trucks(incoming/outgoing) exists and trucks 3, 5 and 7 are allocated to door 1while their order of service is as mentioned earlier based on FCFS rule. In each chromosome the order of service is shown from left to right on each door. In Fig. 2 for example trucks 5, 6 and 9 are selected on each door respectively and exchanged by swap mutation.

The relocation operator is also used for creating new chromosomes by choosing two doors randomly and relocating the trucks on each door with each other as shown in Fig. 3.

Termination criterion

Three different criteria are used for termination including: number of iterations, time limit and similarity of Pareto solutions in 100 consecutive iterations.

Bi-objective bi-level differential evolution algorithm

In cross docking, differential evolution (DE) algorithm has been first implemented by Boloori Arabani et al. (2011) for truck scheduling and determining trucks order of service in a cross-dock and their reports showed that DE algorithm has a better performance in comparison with GA, PSO, ACO and TS. Liao et al. (2013) also used DE as well as two other improved DE algorithms for truck scheduling and allocating problem. The results of their research indicated that DE produces better solutions compared to other solving methods used in their paper.

In this study, we use non-dominated sorting and crowding distance (like NSGA-II) criteria to solve a bi-objective problem using DE algorithm. In DE algorithm cross-over and mutation operators are defined for problems with continuous nature, however, the problem discussed in this paper as well as the suggested model require discrete vectors for showing the solution. Hence another method is implemented for vectors representation in which the vectors are created in a continuous manner and after achieving new solutions by implementing cross-over and mutation, vectors are changed

Fig. 2 Swap mutation

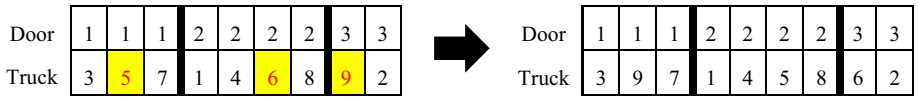


Fig. 3 Relocation mutation

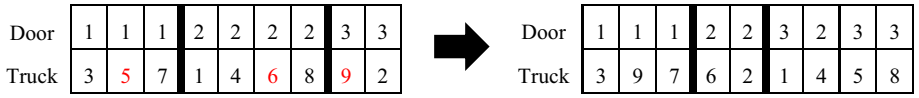


Fig. 4 Encoding with continuous numbers

0.3998	0.3333	0.19445	0.5121	0.65697	0.7774	0.6667	0.06868	0.3608
0.1196	0.5191	0.3795	0.3073	0.8704	0.5826	0.2904	0.8492	0.7532

into a discrete state. The main structure of this algorithm is explained thoroughly through the following sections.

Solution representation

Each solution is a vector of two rows with the first row indicating the number of doors and the second one representing the number of trucks that are encoded with continuous numbers as shown in Fig. 4 . These two rows represent the allocation of trucks to doors and also their order of service on each door and the vectors are created for both inbound and outbound trucks and doors.

Initialization

To create each member of the population, based on the number of doors and inbound and outbound trucks, two vectors are generated. For example in Fig. 4 to generate the first vector, considering the number of incoming trucks, random numbers in [0 1] are generated in two rows. All the initial population members are created the same way. To improve the quality of the members in the first level, as described earlier in NSGA-II, a single objective GA is used and to calculate the objective function value in the first level, all vectors must be decoded. As an example as shown in Fig. 5 to decode the first row of the vector created for 9 trucks and 3 inbound doors each element of the first row is multiplied by $n_1 = 3$ (the number of doors) and rounded up.

According to Fig. 6 decoding of the second row starts with sorting the array in an ascending order, then the position number of each element of the second row is considered as the truck number.

After decoding of the vectors, the number of allocated trucks to each door is determined, the order of service of each truck on each door is based on FCFS and finally a schedule is attained as shown in Fig. 7.

Mutation operator

To implement the mutation operator three distinguished solution vectors $\{X_{r1}, X_{r2}, X_{r3}\}$ are chosen randomly from the population at hand in such way that $i \neq r_3 \neq r_2 \neq r_1$. The chosen vectors are combined using the following equation:

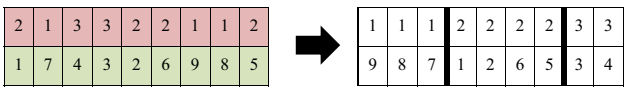


Fig. 7 The schedule attained by the decoded vector

Fig. 5 Decoding of the vector's first row

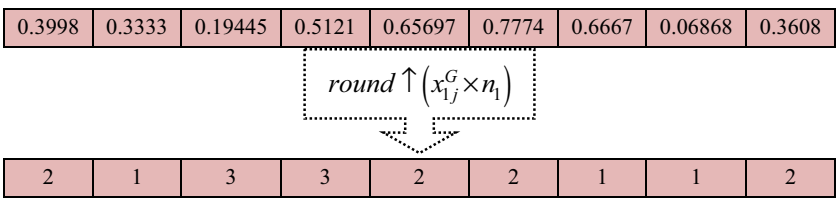
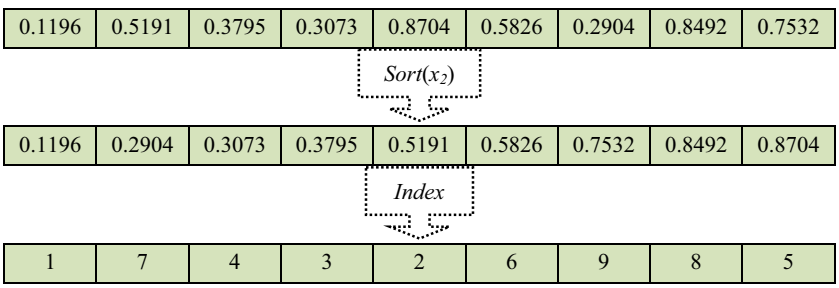


Fig. 6 Decoding of the vector's second row



$$V_i^G = X_{r_1}^G + F \times \left(X_{r_2}^G - X_{r_3}^G\right); \quad i = 1, ..., NP, \quad r_1, r_2, r_3 \in \{1, ..., NP\} \tag{46}$$

The DE's mutation operator ads a fraction of difference of two solution vectors to the third one; F is the mutation factor in [0, 1] range which controls DE's vector participation when creating new population.

Cross-over operator

After a mutated vector is created, cross-over is performed based on the following:

$$U_{i,j}^G = \begin{cases} V_{i,j}^G \text{ if } rand_j \leq CR;\ j = k \\ X_{i,j}^G \text{ otherwise} \end{cases}$$

In which $k \in \{1, ..., D\}$ is the random factor and is created for each i. This factor makes sure that a member of the experimental vector be transferred to the offspring population. $CR \in [0, 1]$ is the constant cross-over parameter and $rand_j$ is a uniform random number which is compared with cross-over operator in order to determine the offspring vector's elements. If $rand_j$ be equal or smaller than cross-over rate the element of the experimental vector is transformed to the next generation, otherwise, the corresponding position in offspring vector is chosen from the current population. $U_{i,j}^G$,$V_{i,j}^G$, and $X_{i,j}^G$ are the j^{th} member of the i^{th} offspring vector, the j^{th} member of the i^{th} experimental vector, and the j^{th} member of the i^{th} goal vector in G^{th} generation respectively (Storn and Price 1997).

Repair procedure

After performing mutation and cross-over on a vector, there is always the possibility of creating some smaller than 0 or larger than 1 numbers in the first row, in the repair procedure, these numbers are replaced by random numbers in [0 1].

Sorting and selection

The same as NSGA-II, in this algorithm, population members are sorted based on two criteria, population distance and non-dominated sorting. Members with the highest ranks are chosen as the new population. It is worth noting that in this paper, each newly generated member is compared with all the population in order to be selected and not only its parents.

Termination

Termination criteria are the same as NSGA_II algorithm and all the non-dominated members that are a part of Pareto frontier are selected as final answers.

Computational results

In this section the experiments calculations and results are explained.

Design of experiments (DOE)

The main purpose of DOE is answering to the following question: is there a significant difference between solutions quality obtained by the three different solving procedures, (47) is the hypothesis test and the Significant level of error is less than or equal to 5 %.

$$\begin{aligned} &H_0 : \alpha_1 = \alpha_2 = \alpha_3 \\ &H_1 : \text{ Otherwise} \end{aligned} \tag{47}$$

The sample problems are created based on two factors, number of trucks and number of doors and each sample problem is solved by all the three algorithms (GASH, NSGA-II and MODE) hence factorial design is considered as the best choice to compare the results due to the fact that there exists more than a single factor and one of them is more important than the other one, and is used as the completely random basic design. Equation(48) shows the designed model used for statistical experiments

$$\begin{aligned} &Y_{ijk} = \mu + T_{ij} + \varepsilon_{ijk} \qquad i = 1, 2, 3 \quad j = 1, ..., 20 \quad k = 1, ..., 3 \\ &T_{ij} = J_j + H_i + (H * J)_{ij} \end{aligned} \tag{48}$$

where Y_{ijk} is the response variable (here, this response variable is the quality metric (QM), one of the metrics used for comparing the solution qualities obtained by different algorithms which is discussed in section "Comparison metrics")., μ is the average of society, J_j is the effect of problem size, H_i is the effect of algorithm and finally ε_{ijk} is the random error. The presented model is valid when all the observations follow normal distribution and based on Fig. 8 which is the normal probability plot drawn by SAS software, it can be concluded that observations distribution (when Significant level of error is equal to 1 %) is nearly normal; so analysis of variance (ANOVA) methods can be used for precisely examining the results.

DOE is also performed in SAS when Significant level of error = 5 % and Table 1 shows the results obtained by ANOVA.

According to Table 1, since p value ≤ 0.0001, in significant level of error of 5 %, H_0 is rejected i.e., there is a significant difference between the algorithms solutions. To choose the algorithm with the best results Tukey test is used and the results of this test are shown in Table 2.

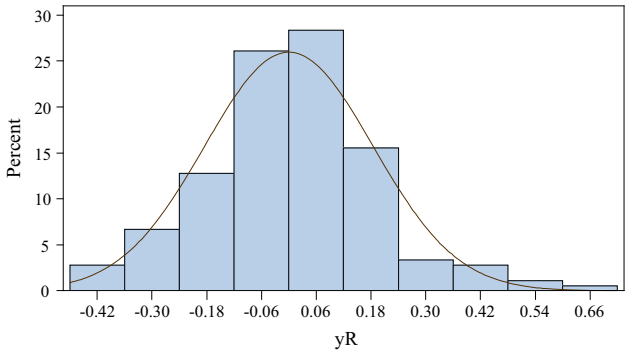


Fig. 8 Normal probability plot for QM in statistical experiments

Table 1 Analysis of variance on algorithms results

Source	DF	Type III SS	Mean square	F Value	Pr > F
Size	19	0.00000000	0.00000000	0.00	1.0000
Algorithm	2	5.42855601	2.71427801	53.50	<.0001
Size*algorithm	38	6.73637734	0.17727309	3.49	<.0001

Table 2 Results of Tukey test

t Grouping	Mean	N	algorithm
Means with the same letter are not significantly different			
MODE	0.57187	60	1
NSGA-II	0.26469	60	2
GASH	0.16344	60	3

Generating sample problems

We consider a cross-dock facility which operates in three 8h (480 min) shifts. Other model parameters are determined based on Konur and Golias (2013) paper. The arrival time window for each truck is generated as follows: first, a random value in [0, 480] is generated, which is used as the mid-time window, i.e., $\frac{A_j^l + A_j^u}{2}$. Then, a random time window length is generated for each truck considering the problem characteristics, i.e., $A_j^u - A_j^l \sim$ U [0, 30](truck arrivals must fall into [0 480] i.e., $A_j^l \geq 0$, $A_j^u \leq 480$). For each problem we let $p \sim$ U[30, 60] in min (time required for loading or unloading of incoming or outgoing trucks), $w = 2p$money units per minute (handling cost per minute is 2 money units for each door) $d_j \sim$ U[1, 2] money units per minute. We focus on 20 distinct problems shown in Table 3 that for each, three instances were created and solved.

Parameter tuning

Parameter tuning is an essential subject when it comes to using metaheuristic algorithms which can be effective on the performance of these algorithms as well. We tuned the parameters of all metaheuristic algorithms that we used, by consecutively solving the model with each algorithm and choosing the best parameters by trial and error. Quality of the solutions and computation time were the two most important factors while selecting the parameters. It was noticed that increasing some parameters such as initial population more than a certain value only increases the computation time and does not affect the quality of the final solutions. The tuned parameters are shown in Table 4.

Table 5 shows the results of solving the created problems in Table 3 by the three different algorithms. In the two algorithms used in this paper (NSGA-II and MODE) the solutions of first level of the problem are improved by a GA and sent to the next level hence the quality of the final solutions are better in comparison to the solutions of GASH. As shown in Table 5 the solutions obtained by MODE algorithm are better than the other two with respect to time and the objective value.

Table 3 Properties of the solved sample problems

Problem no.	Number of doors		Number of trucks		Problem no.	Number of doors		Number of trucks	
	Inbound	Outbound	Incoming	Outgoing		Inbound	Outbound	Incoming	Outgoing
1	2	2	5	5	11	10	10	50	50
2	2	2	10	10	12	10	10	80	80
3	2	2	15	15	13	10	10	100	100
4	2	2	20	20	14	12	12	50	50
5	3	3	10	10	15	12	12	80	80
6	3	3	15	15	16	15	15	100	100
7	3	3	20	20	17	15	15	150	150
8	3	3	30	30	18	15	15	200	200
9	5	5	20	20	19	20	20	150	150
10	5	5	30	30	20	20	20	200	200

Table 4 Tuned parameters of all the metaheuristic algorithms

Tuned parameters	Parameters	Metaheuristic algorithms
GA	Population size	30
	Iterations	50
MODE	Population size	50
	Iterations	2000
	F_{max}	0.8
	F_{min}	0.2
	Cross-over rate	0.2
NSGA-II	Population size	50
GASH	Iterations	2000

Table 5 The results

Problem no.	NSGA-II			MODE			GASH		
	ATSC	R	CPU time (min)	ATSC	*R*	CPU time (min)	ATSC	R	CPU time (min)
1	736.5	101.0	7.2	759.0	64.0	3.1	730.5	113.0	8.3
2	2112.3	32.5	10.3	2166.8	0.5	5.9	2249.5	1.0	11.0
3	3448.3	267.5	16.9	4115.0	218.0	11.5	3731.5	277.0	17.0
4	7950.4	189.3	19.6	7286.0	469.5	12.9	7523.5	444.5	18.6
5	1608.5	0.0	12.4	1583.0	111.0	8.0	1487.3	117.5	8.0
6	2829.3	267.5	18.6	3310.3	60.5	18.2	4348.5	0.0	8.8
7	4528.8	291.0	16.3	4713.8	187.0	17.1	5422.5	281.0	18.9
8	11872.3	82.5	37.8	10819.8	1016.5	29.1	12160.0	144.0	51.4
9	3452.8	124.5	29.5	3581.8	3.5	19.4	3334.3	83.5	11.3
10	7167.8	547.0	27.8	6923.8	559.7	26.7	9121.3	50.7	17.4
11	10276.7	640.7	31.9	10343.5	741.0	55.4	13343.3	281.3	27.4
12	24169.0	2416.0	45.6	26721.5	2850.0	93.0	36204.0	1018.0	55.0
13	43287.5	3306.0	139.9	45397.5	2779.0	126.1	48686.0	2549.0	99.1
14	10010.7	406.0	36.1	9655.5	335.7	39.2	9910.0	617.3	22.8
15	21159.4	2105.3	58.5	23548.5	2024.5	84.2	28230.4	1708.3	43.9
16	25308.3	3707.5	94.0	27057.8	3177.5	64.5	31211.0	4283.0	75.6
17	56733.5	6367.0	118.1	59563.5	6637.0	90.5	66125.5	6657.0	139.8
18	63256.0	3821.0	158.0	67249.5	4161	115.1	69122.1	4255.0	200.0
19	83211.2	5012.1	200.0	74050.0	4726	138.4	86657.3	5266.0	190.0
20	91345.1	5534.3	200.0	87855	8528	144.7	93812.0	6874.0	200.0

Comparison metrics

To compare the selected algorithms the following comparison metrics are used,

Quality metrics (QM)

This metrics considers all Pareto solutions obtained by each algorithm and performs non-dominated sorting process on all the solutions. The quality of each algorithm is proportional to the Pareto solutions of that specific algorithm that in the range of the new Pareto frontier. The more value for this metrics, the more is the quality of the selected algorithm.

Spacing metrics (SM)

This metrics demonstrate the uniformity of pareto solutions in solution space:

$$SM = \frac{\sum_{i=1}^{n} \left| \bar{d} - d_i \right|}{(n-1)\,\bar{d}} \tag{49}$$

In Eq. (49) d_i is the Euclidean distance of two neighboring pareto solutions in solution space, $\bar{d}$ is the average of all d_i s and n indicates the number of non-dominates solutions.

Table 6 Comparison metrics

Problem no.	Quality metric (QM)			Spacing metric (SM)			Diversity metric (DM)			Mean ideal distance (MID)		
	NSGA-II	MODE	GASH	NSGA-II	MODE	GASH	NSGA-II	MODE	GASH	NSGA-II	MODE	GASH
1	0.1525	0.8475	0	0.7987	0.7816	0.9044	1.3017	1.3017	1.3707	0.7429	0.7376	0.8141
2	1	0	0	0.6169	0.0258	1.4329	1.0272	0.7701	0.5918	0.4658	0.6296	0.7032
3	0.1111	0.7778	0.1111	0.5716	0.7636	0.5382	0.924	1.4142	1.1612	0.5228	0.5546	0.603
4	0.5455	0.3636	0.0909	0.3859	0.5647	0.617	1.1739	1.0304	1.2682	0.5481	0.4853	0.6675
5	0	1	0	0.6902	0.5007	0.9494	1.1965	0.927	0.8399	0.6344	0.5104	0.8884
6	0.6667	0.2222	0.1111	0.5306	1.0439	0.5575	1.048	1.3372	1.1544	0.5276	0.68	0.5909
7	0	1	0	0.1794	1.2115	0.3698	0.8816	1.2028	0.9477	0.6489	0.3114	0.5
8	0	1	0	0.4937	0.7323	0.6348	1.3202	1.3114	1.0802	0.7516	0.6799	0.7331
9	0	1	0	1.0082	0.6627	1.1026	0.6308	1.0034	0.7477	0.6889	0.3436	0.8069
10	0	1	0	0.2574	0.7696	0.7098	1.0396	1.2611	1.112	0.5842	0.4503	0.6991
11	0.2941	0.7059	0	0.3576	0.7558	0.5409	0.5082	1.4142	0.5561	0.5203	0.5526	0.5349
12	0.294	0.7059	0	0.6864	0.7765	0.5408	0.6857	1.4142	0.6263	0.682	0.6096	0.7136
13	0.4286	0.5357	0.0357	0.6548	0.8059	0.537	0.6655	1.3505	0.6951	0.6549	0.6263	0.6273
14	0.5	0.1	0.4	0.5419	0.6832	0.702	0.5043	1.3487	0.7195	0.3255	0.6115	0.4973
15	0	1	0	0.6403	0.4565	0.5026	0.6736	0.5355	0.8349	0.7719	0.2856	0.8426
16	0	1	0	0.5441	0.7442	0.6677	0.3241	1.2426	0.4062	0.7218	0.5265	1.0224
17	0.3	0	0.7	0.7043	0.9832	0.5358	0.3977	1.2373	1.1167	0.7497	0.5819	0.5611
18	0.5	0.1667	0.3333	0.2779	0.7349	0.6791	0.5135	1.2665	1.2624	0.5758	0.7252	0.6762
19	0.2143	0.0714	0.7143	0.6713	0.6764	0.4984	0.9776	1.136	1.0622	0.6412	0.8012	0.5557
20	0.3077	0.4615	0.2308	0.1509	0.699	0.6552	0.496	1.1656	0.5404	0.7999	0.9428	0.8482

Lower amounts of SM shows more uniform non-dominates solutions and therefore a better algorithm.

Diversification metrics (DM)

Clearly this metrics indicates the diversity of an algorithm pareto solutions and is calculated as follows:

$$D = \sqrt{\sum_{i=1}^{n} \max(\|x_t^i - y_t^i\|)} \tag{50}$$

in Eq. (50), $\|x_t^i - y_t^i\|$ is the Euclidean distance between x_t^i and the non-dominated solution y_t^i (Tavakkoli-Moghaddam et al. 2010)

Mean ideal distance metrics (MID)

This metrics is equal to the distance of pareto solutions of an algorithm to the ideal solution and is calculated as follows:

$$MID = \frac{\sum_{i=1}^{n} \sqrt{\left(\frac{f_{1i} - f_1^{best}}{f_{1,total}^{\max} - f_{1,total}^{\min}}\right)^2 + \left(\frac{f_{2i} - f_2^{best}}{f_{2,total}^{\max} - f_{2,total}^{\min}}\right)^2}}{n} \tag{51}$$

In Eq. (51), n is the number of pareto solutions while $f_{i,total}^{\max}$ and $f_{i,total}^{\min}$ are the maximum and the minimum value of the objective function among all the algorithms objective functions values. The results of calculating the comparison metrics for each sample problem and using each algorithm are shown in Table 6.

As we can see in Table 6, the solutions obtained by MODE algorithm has better qualities in comparison with the solutions attained by the other two methods (NSGA-II and GASH). According to SM metrics all three algorithms have low uniformity between their pareto solutions and high DM which shows the algorithms produce diverse solutions and can search the solution space properly. MID results also show a good performance by these three methods. In Fig. 9, the pareto solutions acquired by each of the three algorithms are shown schematically and better performance of MODE algorithms is completely notable according to Fig. 9. It also worth noting that in some of the difficult problems NSGA-II and GASH are also capable of producing solutions near the pareto frontier.

Comparison between algorithms based on their CPU time

As described in sections "Termination criterion" and "Termination", in this paper, we considered three different termination criteria for the algorithms; including: number

of iterations, time limit and similarity of Pareto solutions in 100 consecutive iterations. Since MODE algorithm finds the non-dominated Pareto solutions in less iteration and also according to the third termination criterion (similarity of Pareto solutions in 100 consecutive iterations), CPU time to reach the final solutions in this algorithm is less than NSGA-II and GASH.

MODE differs from NSGA-II in the mutation and recombination phases. Unlike NSGA-II, where perturbation occurs at random, MODE uses weighted differences between solution vectors to perturb the population. MODE implements the step sizes of DE to adaptively adjust the solutions' fitness, while NSGA-II uses random mutation operators. It is clear that the reason for NSGA-II's poor performance on truck problems is the uncorrelated variable perturbation during mutation phase.

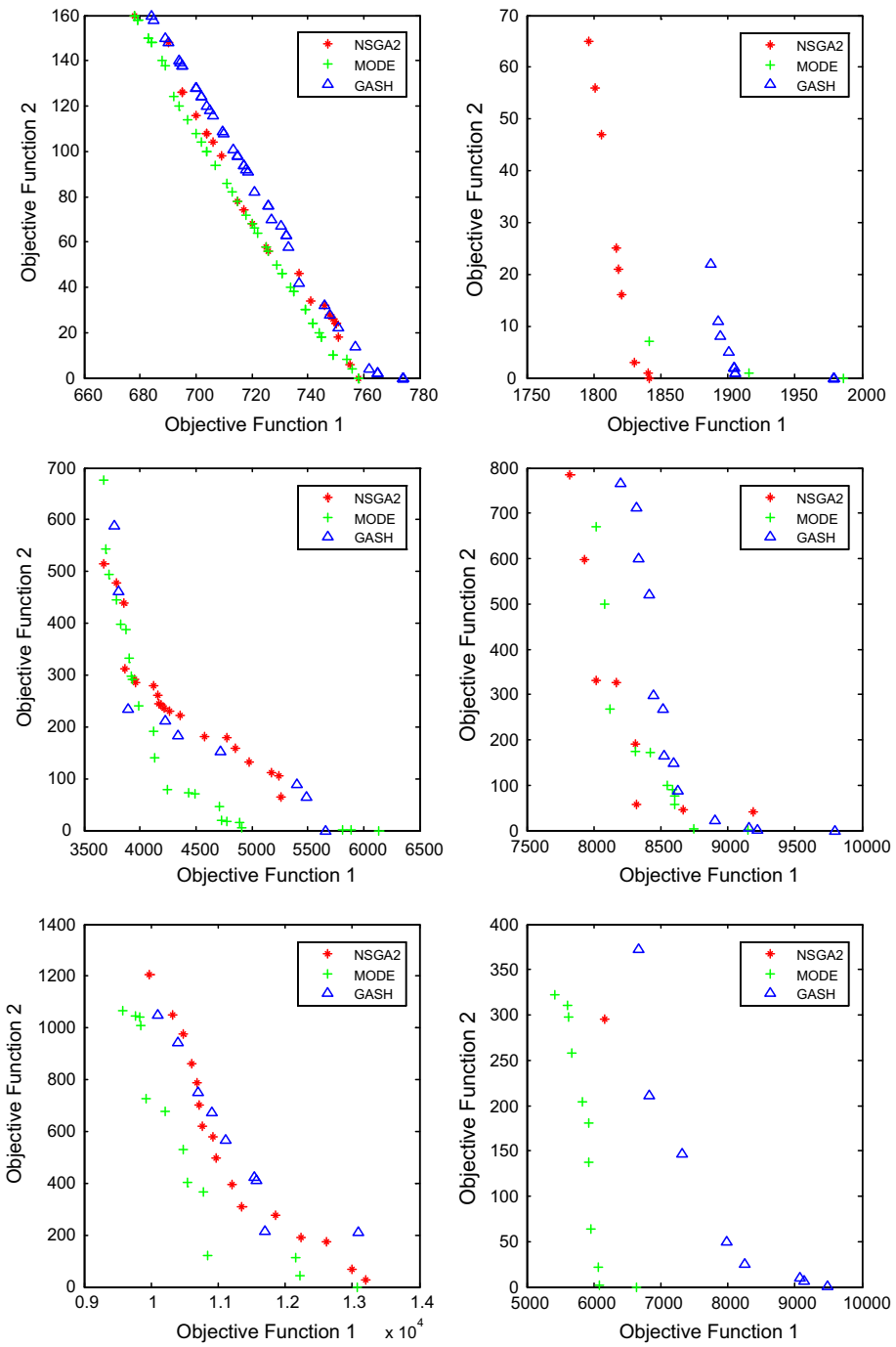
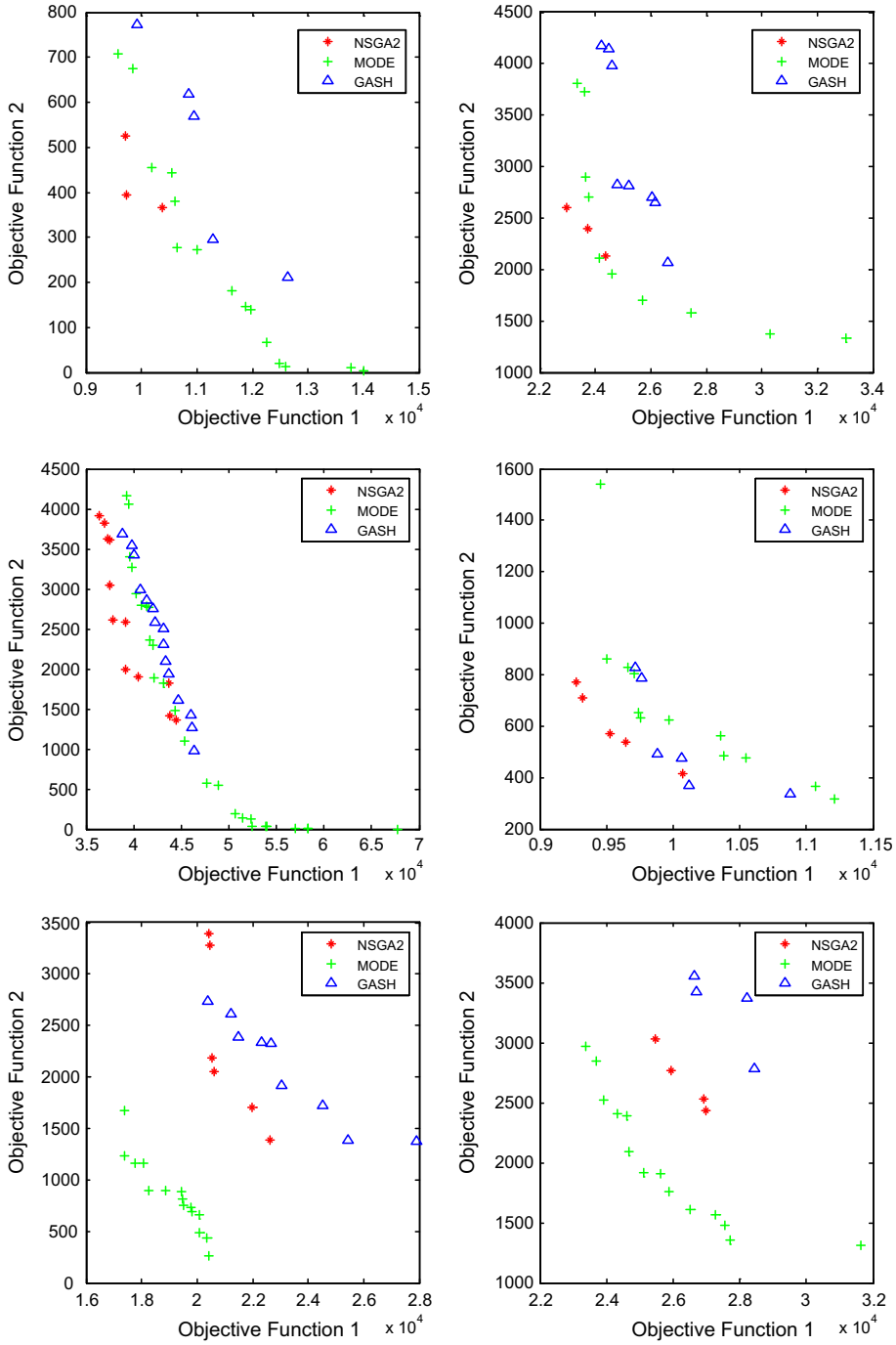


Fig. 9 Pareto solutions of the three algorithms for sample problems number 3, 4, 8, 10, 11, 12, 13, 14, 15 and 16

Fig. 9 continued



Conclusion

In thisstudy for the first time a mathematical model is presented that simultaneously address the problem of scheduling incoming and outgoing trucks in a cross-dock facility while truck arrival times are non-deterministic. The main objective of the bi-level model that is presented here is stabilizing the imposed costs as well as minimizing the total cost in cross-dock facility. Two metaheuristic algorithms, NSGA-II and MODE are used to solve the designed sample problems and the obtained results of these two methods are compared with a random search based GA algorithm existing in the literature.Different properties of each algorithms is examined through four metrics of quality, spacing, diversification and uniformity and pareto solutions of each algorithm are calculated and plotted. In this study only the basic concept of NSGA-II and MODE is used and parameter settings, Cross-Over and Mutation are suggested in accordance with this

problem. Therefore DOE's results shows that there is a significant difference between the solutions attained by each algorithm and MODE performs significantly better than the other two algorithms in terms of solution quality. Finally, considering the internal process of a cross-dock facility, a probability distribution for truck arrival times and simulating the model and post distribution in a cross-dock are all the subjects of the future work.

References

Altiparmak, F., Gen, M., Lin, L., & Karaoglan, I. (2009). A steady-state genetic algorithm for multi-product supply chain network design. *Computers & Industrial Engineering*, *56*, 521–537.

Boloori Arabani, A., Zandieh, M., & Fatemi Ghomi, S. M. T. (2011a). A cross-docking scheduling problem with sub-population multi-objective algorithms. *The International Journal of Advanced Manufacturing Technology*, *58*, 741761.

Boloori Arabani, A. R., Fatemi Ghomi, S. M. T., & Zandieh, M. (2011b). Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage. *Expert Systems with Applications*, *38*, 1964–1979.

Boysen, N. (2010). Truck scheduling at zero-inventory cross docking terminals. *Computers & Operations Research*, *37*, 32–41.

Boysen, N., Briskorn, D., & Tschöke, M. (2012). Truck scheduling in cross-docking terminals with fixed outbound departures. *OR Spectrum*, *35*, 479–504.

Boysen, N., & Fliedner, M. (2010). Cross dock scheduling: Classification, literature review and research agenda. *Omega*, *38*, 413–422.

Chen, F., & Lee, C.-Y. (2009). Minimizing the makespan in a two-machine cross-docking flow shop problem. *European Journal of Operational Research*, *193*, 59–72.

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

Golias, M. M., Ivey, S., Haralambides, H., & Saharidis, G. (2010). *Maximizing throughput and minimizing tardiness and earliness at a cross dock facility: A bi-objective formulation*. Washington: Annual Meeting of the Transportation Research Board.

Hansen, P., Jaumard, B., & Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, *13*, 1194–1217.

Konur, D., & Golias, M. M. (2013). 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*, 71–91.

Kuo, Y. (2013). Optimizing truck sequencing and truck dock assignment in a cross docking system. *Expert Systems with Applications*, *40*, 5532–5541.

Liao, T. W., Egbelu, P. J., & Chang, P. C. (2013). Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truckschedule in multi-door cross docking operations. *International Journal of Production Economics*, *141*, 212–229.

Storn, R., & Price, K. (1997). Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*, 341–359.

Tavakkoli-Moghaddam, R., Azarkish, M., & Sadeghnejad-Barkousaraie, A. (2010). Solving a multi-objective job shop scheduling problem with sequence-dependent setup times by a Pareto archive PSO combined with genetic operators and VNS. *The International Journal of Advanced Manufacturing Technology*, *53*, 733750.

Van Belle, J., Valckenaers, P., & Cattrysse, D. (2012). Cross-docking: State of the art. *Omega*, *40*, 827–846.

Wang, Z. X., Chan, F. T. S., Chung, S. H., & Niu, B. (2015). Minimization of delay and travel time of yard trucks in container terminals using an improved GA with guidance search. *Mathematical Problems in Engineering*, *2015*, 1–12.

Wang, Z. X., Chan, F. T. S., Chung, S. H., & Niu, B. (2014). A decision support method for internal truck employment. *Industrial Management and Data Systems*, *114*, 1378–1395.

Yu, W., & Egbelu, P. J. (2008). Scheduling of inbound and outbound trucks in cross docking systems with temporary storage. *European Journal of Operational Research*, *184*, 377–396.