Scheduling trucks in a multi-door cross-docking system with time windows

G. OZDEN* and I. SARICICEK

1 Eskisehir Osmangazi University, Eskisehir Vocational School, Machinery Program, Eskisehir, Turkey
2 Eskisehir Osmangazi University, Department of Industrial Engineering, Eskisehir, Turkey

Abstract. Cross-docking is a strategy that distributes products directly from a supplier or manufacturing plant to a customer or retail chain, reducing handling or storage time. This study focuses on the truck scheduling problem, which consists of assigning each truck to a door at the dock and determining the sequences for the trucks at each door considering the time-window aspect. The study presents a mathematical model for door assignment and truck scheduling with time windows at multi-door cross-docking centers. The objective of the model is to minimize the overall earliness and tardiness for outbound trucks. Simulated annealing (SA) and tabu search (TS) algorithms are proposed to solve large-sized problems. The results of the mathematical model and of meta-heuristic algorithms are compared by generating test problems for different sizes. A decision support system (DSS) is also designed for the truck scheduling problem for multi-door cross-docking centers. Computational results show that TS and SA algorithms are efficient in solving large-sized problems in a reasonable time.

Key words: multi-door cross-docking center, time window, logistics, decision support system, meta-heuristics.

1. Introduction

Cross-docking is the operation of conveying products through distribution centers without warehousing them. In a traditional warehouse, the products move from receiving to storage and then from storage to shipping processes. The cross-docking system's success depends on as short a storage period as possible in the receiving/shipping plant [1]. By using a cross-docking system, goods are delivered to the cross-docking center via inbound trucks and are directly sorted, repackaged, routed and loaded into outbound trucks to be delivered, thus bypassing storage [2].

One primary activity of a cross-docking system in particular entails the effective coordination of inbound and outbound trucks. As order volumes increase or when deliveries are uncoordinated, the inventory can increase, and thus cross-docking centers must be managed efficiently. Meanwhile, the truck scheduling problem is one of the most significant operational problems that consist in assigning and sequencing trucks at receiving or shipping doors in cross-docking centers. McWilliams, Stanfield and Geiger [3] were the first to address the issue of short-term truck scheduling and to examine a terminal of a postal service provider to minimize the time span of the transfer operation. They also proposed a simulation-based scheduling algorithm that uses a genetic algorithm (GA). Yu and Egbelu [4] proposed a mathematical model to obtain the best schedule for all inbound and outbound trucks to minimize the total operation time. Because the truck scheduling problem is NP-hard [5], meta-heuristics have often been applied. Vahdani and Zandieh [2] applied GA, TS, SA, electromagnetism-like algorithm (EMA) and variable neighborhood search (VNS), while Boloori Arabani, Fatemi Ghomi and Zandieh, [6] proposed GA, TS, particle swarm optimization (PSO), ant colony optimization (ACO) and differential evolution (DE) algorithms to schedule trucks in cross-docking centers that were previously suggested by Yu and Egbelu.

Van Belle, Vâlckenaers and Cattrysse [7] classify studies regarding the truck scheduling problem into three groups. The first group indicates a cross-docking center with a single receiving and single shipping door [2, 4, 6, 8–12]. The truck scheduling problem's objective is to decide where and when inbound and outbound trucks should be loaded or unloaded at multi-door cross-docking centers. Studies in the second group consider cross-docks with multiple receiving and shipping doors but solely address scheduling the inbound trucks [3, 13–17]. The last category indicates scheduling both inbound and outbound trucks at multiple receiving and shipping doors. A number of studies in this category are available in the literature. Boysen [18] suggests SA algorithm and the dynamic programming method for truck scheduling in the cross-docking process. Meanwhile, Lee, Kim and Joo [19] introduce a mixed integer programming (MIP) model to obtain door assignment and docking sequences for all trucks. The researchers propose a GA to maximize the number of products that need to be shipped within a given working period. Joo and Kim [20] consider a truck scheduling problem for three groups of trucks: inbound trucks, compound trucks and outbound trucks. The researchers first introduce compound trucks that play the roles of outbound and inbound trucks, as in this paper. Self-evolution algorithm and GA are proposed for truck scheduling to minimize the makespan. The study by Van Belle, Vâlckenaers,
2. Problem description

The cross-docking center examined in this study has a receiving dock where products are unloaded from inbound trucks and a shipping dock where products are loaded onto outbound trucks with multiple doors, and the docks are separated from one another (Fig. 1).

Inbound trucks arrive at the assigned receiving door successively and unload products onto the receiving dock. Outbound trucks arrive at the assigned shipping door successively and load products from the shipping dock. Compound trucks are described as both inbound and outbound ones. A compound truck is an inbound truck when it arrives at the receiving dock and it is an outbound one when it goes to the receiving dock to load products. Compound trucks unload products onto the receiving dock and transfer to the shipping dock to load products. The problem is the determination of when and where inbound, outbound and compound trucks should be processed at multi-door cross-docking centers. The minimizing of overall earliness and tardiness of outbound trucks is crucial in a cross-docking center considering the time window aspect. The time window for outbound trucks is shown in Fig. 2.

A time window is a time interval in which the outbound trucks’ loading operation should be completed. If the loading operation is finished within the time window, no tardiness or earliness is incurred, and the process is considered to be completed on time. Otherwise, the operation causes a penalty, de-
pending on whether the loading operation of a truck is finished before or after the time window. In Fig. 2, \( e_j, d_j, GD_j, b_j \) and \( C_j \) refer to the beginning of the time window, the end of the time window, the arrival time of outbound truck \( j \), and then the start time and completion time of loading onto outbound truck \( j \), respectively.

Minimizing overall earliness and tardiness can be of particular use for the cross-docking systems that use the “just in time” approach to deliver products in the customer’s time window due to dates. The punctuality of product deliveries affects the system performance. There are time windows for outbound trucks that are determined by the customer. Outbound trucks should be delivered to the customer within these specific time windows.

According to the notation proposed by Boysen and Fliedner [23], the truck scheduling problem consists of three main elements which are denoted as tuple \( \alpha/\beta/\gamma \) where \( \alpha \) is the door environment, \( \beta \) stands for operational characteristics and \( \gamma \) means the objectives. This truck scheduling problem can be denoted as follows:

\[
\hat{E}/r_j, \bar{d}_\alpha, \text{change} / \sum(E_j + T_j).
\]

Here \( E_j, r_j, \bar{d}_\alpha, \text{change} \) and \( \sum(E_j + T_j) \) represents that each door is either exclusively dedicated to inbound or outbound operations, and that arrival times are different per truck, deadlines for outbound trucks, interchangeability of products as well as the objective to minimize overall earliness and tardiness, respectively.

The assumptions of this problem are defined as follows:

- Inbound, outbound and compound trucks must remain at the respective doors until their unloading or loading operations are finished and must leave as soon as they finish their operation.
- After products are unloaded, they are transferred to the shipping doors and stay until the appropriate outbound truck arrives there.
- Because the freight is shipped in standardized pallets, the unit loading time and unloading time are assumed to be fixed. The unit unloading time and the unit loading time remain the same regardless of product type and are calculated for one unit of product.
- The expected arrival time is known for each inbound, outbound and compound truck.
- There are compound trucks that both unload and load products.
- Inbound and outbound trucks hold different products and they are known.
- The sequence in which goods are unloaded or loaded is not considered.
- The truck changeover time is the same for all trucks and unit transfer time from the receiving area to the shipping area is the same for all product types.
- The time windows are considered for outbound trucks. The time windows are not defined as hard constraints; however, tardiness and earliness of outbound trucks should be minimized.

### 2.1. Parameters

- \( I : \{1, \ldots, n_i\} \): the set of inbound trucks (index \( i \))
- \( O : \{1, \ldots, n_o\} \): the set of outbound trucks (index \( j \))
- \( C : \{1, \ldots, n_c\} \): the set of compound trucks (index \( i \in I \) and index \( j \in O \))
- \( P : \{1, \ldots, n_p\} \): the set of product types (index \( k \))
- \( R : \{1, \ldots, n_r\} \): the set of receiving doors (index \( m \))
- \( S : \{1, \ldots, n_s\} \): the set of shipping doors (index \( n \))

- \( r_{ik} \): The number of products of type \( k \) that is loaded onto inbound truck \( i \)
- \( s_{jk} \): The number of products of type \( k \) that is needed for outbound truck \( j \)
- \( CT \): Truck changeover time (\( TL + TL \))
- \( UT \): The unit unloading time of products
- \( LT \): The unit loading time of products
- \( TA \): Transfer time for compound truck from receiving dock to shipping dock
- \( GL_i \): Arrival time of inbound truck \( i \) at cross-docking center
- \( GD_j \): Arrival time of outbound truck \( j \) at cross-docking center
- \( TE \): The truck entering time to a door
- \( TL \): The truck leaving time from a door
- \( e_j \): Beginning of the time window
- \( d_j \): End of the time window
- \( V \): The moving time of products from receiving dock to shipping dock
- \( M \): A positive large number which is at least as large as the sum of loading and unloading times and truck changeover times for all trucks.

### 2.2. Decision variables

- \( a_i \): Start time of unloading for inbound truck \( i \)
- \( L_i \): Completion time of unloading for inbound truck \( i \)
- \( b_j \): Start time of loading for outbound truck \( j \)
- \( C_j \): Completion time of loading for outbound truck \( j \)
- \( x_{ijk} \): The number of products of type \( k \) that are transferred from inbound truck \( i \) to outbound truck \( j \)
- \( E_j \): Earliness of outbound truck \( j \) \( E_j = \max \{ e_j - C_j, 0 \} \)
- \( T_j \): Tardiness of outbound truck \( j \) \( T_j = \max \{ C_j - d_j, 0 \} \)

- \( g_{ij} = \begin{cases} 1, & \text{if any products are transferred from inbound truck } i \text{ to outbound truck } j \\ 0, & \text{otherwise} \end{cases} \)

- \( p_{ijm} = \begin{cases} 1, & \text{if truck } i \text{ is assigned before truck } j \text{ in the sequence at receiving door } m(i \neq j); \text{ or if inbound truck } i \text{ is the first truck at receiving door } m(i = j) \\ 0, & \text{otherwise} \end{cases} \)

- \( q_{ijn} = \begin{cases} 1, & \text{if truck } j \text{ is assigned before truck } i \text{ in the sequence at shipping door } n(j \neq i); \text{ or if outbound truck } j \text{ is the first truck at shipping door } n(j = i) \\ 0, & \text{otherwise} \end{cases} \)

- \( y_{im} = \begin{cases} 1, & \text{if inbound truck } i \text{ is assigned to receiving door } m \\ 0, & \text{otherwise} \end{cases} \)

- \( z_{jn} = \begin{cases} 1, & \text{if outbound truck } j \text{ is assigned to shipping door } n \\ 0, & \text{otherwise} \end{cases} \)
2.3. Mathematical model

Min \( \sum_{j=1}^{n_o} (E_j + T_j) \)  
\[ \sum_{j=1}^{n_o} q_{ijn} = z_{jn} \quad \forall (j \in O, n \in S) \quad (1) \]

s.t.

\( a_i + (UT \cdot \sum_{k=1}^{n_p} r_{ik}) \leq L_i \)  
\( \forall (i \in I) \quad (2) \)
\( \sum_{j=1}^{n_o} q_{ijn} \leq z_{jn} \quad \forall (j \in O, n \in S) \)  
\( i \neq j \quad (16) \)

\( L_i + CT \leq a_j + M \cdot \left(1 - \sum_{m=1}^{n_p} p_{ijm}\right) \)  
\( \forall (i, j \in I) \)  
\( i \neq j \quad (3) \)
\( \sum_{j=1}^{n_o} x_{ijk} = r_{ik} \quad \forall (i \in I, k \in P) \quad (17) \)

\( L_i + CT \leq b_j + M \cdot \left(1 - \sum_{n=1}^{n_o} q_{ijn}\right) \)  
\( \forall (i \in I, j \in O) \)  
\( i \neq j \quad (5) \)
\( E_j \geq e_j - (C_j + TL) \)  
\( \forall (j \in O) \quad (20) \)

\( C_j + CT \leq b_i + M \cdot \left(1 - \sum_{n=1}^{n_o} q_{ijn}\right) \)  
\( \forall (i \in I, j \in O) \)  
\( i \neq j \quad (4) \)
\( \sum_{i=1}^{n_o} x_{ijk} \leq s_{jk} \quad \forall (j \in O, k \in P) \quad (18) \)

\( b_j + (LT \cdot \sum_{k=1}^{n_p} s_{jk}) \leq C_j \)  
\( b_j \geq (C_j + TL) - d_j \)  
\( \forall (j \in O) \quad (21) \)
\( \sum_{k=1}^{n_o} x_{ijk} \leq M \cdot g_{ij} \quad \forall (i \in I, j \in O) \quad (19) \)

\( a_i \geq GL_i + TE \cdot \left(\sum_{m=1}^{n_p} p_{im}\right) \)  
\( \forall (i \in I) \quad (6) \)
\( T_j \geq (C_j + TL) - d_j \)  
\( \forall (j \in O) \quad (21) \)
\( a_i + (UT \cdot \sum_{k=1}^{n_p} r_{ik}) + TL + TA \leq b_i \)  
\( \forall (i \in C) \quad (22) \)

\( L_j + V \leq b_j + M \cdot \left(1 - g_{ij}\right) \)  
\( \forall (i \in I, j \in O) \quad (8) \)
\( g_{ij} \in \{0, 1\} \quad \forall (i \in I, j \in O) \quad (24) \)

\( \sum_{m=1}^{n_o} y_{im} = 1 \)  
\( \forall (i \in I) \quad (9) \)
\( z_{jn} \in \{0, 1\} \quad \forall (j \in O, n \in S) \quad (25) \)

\( \sum_{i=1}^{n_o} p_{im} = 1 \)  
\( \forall (m \in R) \quad (10) \)
\( p_{ijm} \in \{0, 1\} \quad \forall (i, j \in I, m \in R) \quad (26) \)

\( \sum_{j=1}^{n_o} p_{ijm} = y_{im} \)  
\( \forall (i \in I, m \in R) \quad (11) \)
\( q_{ijn} \in \{0, 1\} \quad \forall (i \in I, j \in O, n \in S) \quad (27) \)

\( y_{im} \leq 1 \)  
\( \forall (i \in I, m \in R) \quad (12) \)
\( b_j, C_j, E_j, T_j \geq 0 \)  
\( \forall (j \in O) \quad (28) \)

\( \sum_{j=1}^{n_o} p_{ijm} \leq y_{im} \)  
\( i \neq j \quad (13) \)
\( a_i, L_i \geq 0 \)  
\( \forall (i \in I) \quad (29) \)

\( \sum_{n=1}^{n_o} z_{jn} = 1 \)  
\( \forall (j \in O) \quad (13) \)
\( x_{ijk} \geq 0 \)  
\( \forall (j \in O, k \in P, i \in I) \quad (30) \)

\( \sum_{j=1}^{n_o} q_{ijn} = 1 \)  
\( \forall (n \in S) \quad (14) \)

Objective function (1) is the overall tardiness and earliness of loading operations of outbound trucks. Constraint sets (2–3) ensure the precedence relation of inbound trucks assigned
to the same receiving door. Constraint sets (4–5) ensure the precedence relation of outbound trucks assigned to the same shipping door. Constraint sets (6–7) guarantee that start time of unloading for an inbound truck and start time of loading for an outbound truck must be later than the arrival times of these trucks at the cross-docking center and the truck entering time if these trucks are the first trucks at receiving or shipping doors. Constraint set (8) connects the start time of loading for an outbound truck to the completion time of unloading for an inbound truck if any products are moved between the trucks. Constraint set (9) ensures that each inbound truck is assigned to only one door at the receiving dock. In constraint set (10), the variable \( p_{ij} \) becomes 1 if inbound truck \( i \) is at the beginning of the sequence at the assigned door. Constraint set (11) dictates that an inbound truck is immediately preceded by one inbound truck if it is assigned to a door. Constraint set (12) dictates that an inbound truck must be succeeded by at most one inbound truck if it is assigned to a door. Constraint set (13) indicates that each outbound truck is assigned to only one door in the shipping area. Constraint set (14) guarantees that only one outbound truck is assigned at the first sequence at each shipping door. In this constraint, the variable \( q_{jj} \) becomes 1 if outbound truck \( j \) is the first positioned truck at door \( n \). Constraint set (15) dictates that an outbound truck is immediately preceded by one inbound truck if it is assigned to a shipping door. Constraint set (16) dictates that an outbound truck must be succeeded by at most one truck if it is assigned to a door. Constraint set (17) ensures that the total number of products type \( k \) that transfer from inbound truck \( i \) to all outbound trucks is exactly equal to the number of products type \( k \) that was already loaded into inbound truck \( i \). Constraint set (18) ensures that the total number of products type \( k \) that transfer from all inbound trucks to outbound truck \( j \) is exactly equal to the number of products type \( k \) needed for outbound truck \( j \). Constraint set (19) guarantees the exact relation between the \( x_{ijk} \) variables and the \( g_{ij} \) variables. Constraint set (20, 21) evaluates earliness and tardiness for outbound trucks. Constraint set (22) connects the start time of unloading to the starting time of loading for a compound truck. Constraints (23–30) impose binary and non-negativity conditions on the variables.

Test problems are required to compare optimum solutions obtained by implementing the MIP model in the GAMS 23.3, CPLEX 12.1 solver with the results of the suggested meta-heuristics to evaluate their performance. To measure the effectiveness of the meta-heuristic algorithms, small-sized and large-sized problems are generated.

### 3. Generation of test problems

The following factors are defined to generate test problems.

#### 3.1. Factors for test problems.

**Truck per door factor \((\mu)\):** This is the average number of trucks loaded or unloaded per receiving or shipping door. The truck per door factor is defined as follows:

\[
\mu = \frac{O}{S} \quad \text{(31)}
\]

\[
\mu = \frac{I}{R} \quad \text{(32)}
\]

In Eq. (31) and (32), \( \mu \), \( O \), \( S \), \( I \) and \( R \) represent the truck per door factor, the number of outbound trucks, the number of inbound trucks and the number of receiving doors, respectively.

**Average number of products per truck \((\bar{p})\):** The average numbers of products per inbound and outbound trucks are defined as follows:

\[
\bar{p}_i = \frac{T_{\text{UP}}}{I} \quad \text{(33)}
\]

\[
\bar{p}_o = \frac{T_{\text{LP}}}{O} \quad \text{(34)}
\]

In Eq. (33) and (34), \( p_i \), \( p_o \), \( T_{\text{UP}} \), \( T_{\text{LP}} \) show the average numbers of products per inbound truck and the average numbers of products per outbound truck, the total number of unloaded products and the total number of loaded products, respectively.

**Time window (e-to-d) tightness factor \((\delta)\):** This is used to control the range of the time window and is denoted as:

\[
\delta = \beta - \alpha \quad \text{(35)}
\]

In Eq. (35), \( \beta \) and \( \alpha \) refer to the factors for upper bound and the lower bound of the time window, respectively. The bounds of the time window are denoted as:

\[
d_j = \beta \cdot \pi \quad \text{(36)}
\]

\[
e_j = \alpha \cdot \pi \quad \text{(37)}
\]

In Eq. (36) and (37), \( d_j \), \( e_j \), \( \pi \) represents the upper bound of the time window, the lower bound of the time window and the factor for the bounds of the time window, respectively.

The factor for the time window is denoted as:

\[
\pi = \bar{p}_i U T + G D_j + \bar{p}_o L T \quad \text{(38)}
\]

where \( \bar{p}_i \), \( U T \), \( G D_j \), \( \bar{p}_o \), and \( L T \) denote the average number of products per inbound truck, unit unloading time, arrival time of an outbound truck, average number of products per outbound truck and unit loading time, respectively.

**Maximum arrival time for all trucks:** This is the upper bound of the arrival time of the trucks at the cross-docking terminal. The maximum arrival time is derived as:

\[
2 G D_j = 2 \bar{p} \mu \rho \left( 2 - \theta \mu \rho \right) \quad \text{(39)}
\]
In Eq. (39), $\rho$ represents the arrival time range factor and is calculated by the following formulation:

$$\rho = 2\bar{GD}_j/\hat{C}_{\text{max}}.$$  

(40)

Where $\hat{C}_{\text{max}}$ denotes the estimated total operation time and is defined as:

$$\hat{C}_{\text{max}} = (2 - \theta \bar{GD}_j + \rho)\mu.$$  

(41)

In Eq. (41), $\theta$ shows the effect of the arrival times on total operation time. A similar formulation was used by Kaplan and Rabadi [24].

**Arrival time range factor ($\rho$):** Depending on the estimated $C_{\text{max}}$, the variability of arrival times is indicated by $\rho$. This is a criterion of how dispersed the arrival times are as compared with the estimated total operation time ($\hat{C}_{\text{max}}$).

**Arrival times of inbound ($GL_j$) and outbound ($GD_j$) trucks:** Arrival times follow uniform distribution at the interval of $[0, 2GL_j]$ and $[0, 2GD_j]$ where $GD_j$ is the trucks’ arrival time average.

**3.2. Data generation.** The parameters used to generate test problems are derived by coding a data generation program in Python 3.4 software. The user interface is demonstrated in Fig. 3, where the user loads, saves and enters a new data set.

In the screenshot as shown in Fig. 3, the user is asked for the numbers of inbound, outbound, and compound trucks, the loading and unloading time, the truck changeover time, the truck transfer time, the transfer time of the goods as well as the numbers of receiving doors, shipping doors and types of goods. Based on the information provided, the system gener-
Tabu search is a meta-heuristic procedure introduced by Glover [28, 29]. TS uses a neighborhood search technique by progressing iteratively from one solution to another until a stopping condition is satisfied. The size of a tabu list and the size of the neighborhood are used in 4 and 5 in small-sized test problems, respectively, and the size of a tabu list and the size of the neighborhood are used in 6 and 9 in large-sized test problems, respectively. The algorithmic description of the TS is as follows [26]:

Step 1: Let \( k = 1 \)
Start with schedule \( Z_1 \) and set \( Z_0 = Z_1 \)

Step 2: Set \( Z_c \) (Generate neighborhood of \( Z_k \))
If the move \( Z_{k+1} \) is forbidden by a change on the tabu list;
Set \( Z_{k+1} = Z_t \) and go to step 3.
If the move \( Z_{k+1} \) is not forbidden by any change on the tabu list;
Set \( Z_{k+1} = Z_z \);
Add the reverse move to the top of the tabu list and remove the entry at the bottom.
If \( F(Z_c) < F(Z_0) \), set \( Z_0 = Z_c \) go to Step 3.

Step 3: Set \( k = k + 1 \)
Stop if stopping criteria are satisfied; otherwise go to Step 2.

4.2. Initial solution and neighborhood-generation mechanism

Initial solution. In the first step, inbound trucks are sorted by arrival times from earliest to the latest. Then, the inbound truck which arrives earlier than the other trucks is assigned one by one to the next receiving doors. If certain inbound trucks have the same arrival time at the cross-docking center, the truck that has the most goods is assigned a receiving door first. Outbound trucks are assigned to the shipping doors in a similar manner. Figure 4 demonstrates an example of initial solution \( (Z_c) \) as a sequence of inbound and outbound truck assignments to the receiving and shipping doors.

![Figure 4](image-url)

Fig. 4. Example of initial solution \( (Z_c) \) as a sequence of inbound and outbound truck assignments to receiving and shipping doors.

Two receiving and two shipping doors are separated with ‘*’ in Fig. 4, the inbound truck sequence at the receiving door 1 is 1–3–4 and at the receiving door 2 it is 5–2 (the arrival time sequence for inbound trucks is 1–5–3–2–4). Similarly, the outbound truck sequence at shipping door 1 is 3–4 and at the shipping door 2 it is 1–2 (the arrival time sequence for outbound trucks is 3–1–4–2).
the inbound truck sequence

<table>
<thead>
<tr>
<th>4</th>
<th>5</th>
<th>*</th>
<th>3</th>
<th>2</th>
<th>*</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>*</td>
<td>5</td>
<td>2</td>
<td>*</td>
<td>1</td>
</tr>
</tbody>
</table>

current sequence

new sequence

the outbound truck sequence

<table>
<thead>
<tr>
<th>1</th>
<th>3</th>
<th>5</th>
<th>*</th>
<th>2</th>
<th>*</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>5</td>
<td>*</td>
<td>2</td>
<td>*</td>
<td>4</td>
</tr>
</tbody>
</table>

current sequence

new sequence

Fig. 5. Current and new \((Z_j)\) sequences for inbound and outbound trucks

**Neighborhood-generation mechanism.** The swap move is used to generate neighborhoods. While generating neighborhoods, two random numbers are derived and two trucks or a truck and a door "*" are interchanged by the corresponding number. The mechanism to generate a neighborhood \((Z_j)\) for the sequences of inbound and outbound trucks is shown in Fig. 5.

In Fig. 5, two random numbers 2 and 4 are generated and two cells of the current sequence of inbound trucks are changed. In the new sequence, inbound truck 4 and 3 are unloaded at the first receiving door; inbound truck 5 and 2 are unloaded at the second receiving door and inbound truck 1 is unloaded at the third receiving door. In a similar way, two random numbers 3 and 4 are generated and two cells of the current sequence of outbound trucks are changed. In the new sequence, outbound trucks 3 and 1 are loaded at the first shipping door, outbound trucks 5 and 2 are loaded at the second shipping door and outbound truck 4 is loaded at the third shipping door.

**Product assignment algorithm.** At the neighborhood generation stage for meta-heuristics, during the decision-making regarding where and when inbound/outbound trucks will be unloaded/loaded, decisions regarding product assignment from inbound trucks to outbound trucks are made. For this purpose, the following algorithm is designed and coded via the Python 3.4 software.

In the algorithm, \(t\), \(m\) and \(y\) refer to the decision time for truck-door assignment, the number of outbound trucks and the number of shipping doors, respectively. \(w_j\) is the waiting time of outbound truck \(j\) while loading. The start time of loading for outbound trucks can be defined as in the \(b_j = \max GD_j, a_i + (LT \cdot \sum_{k=1}^{np} s_{jk}) + CT\) formula. In this expression, the start time of loading for an outbound truck is equal to or later than the arrival time of the outbound truck at the cross-docking center and the finish time of the loading for previous truck at the same door. While specifying the start time of loading for outbound trucks, the parameters that need to be controlled are the arrival time of outbound trucks at the cross-docking center, the finish time of loading for the previous truck at the same door, the availability of shipping doors, the ready time when all products need to be loaded onto the outbound truck and the time window of the outbound truck.

**4.3. Results.** The different arrival times for inbound and outbound trucks and the time windows for outbound trucks are generated by using the indicated data generation program. Parameters are dependent on the problem instances in terms of the three levels of lower and upper bound coefficients of the time window, namely, optimistic \((\alpha = 0, \beta = 2)\), possible \((\alpha = 0.25, \beta = 1.75)\), and pessimistic \((\alpha = 0.5, \beta = 1.5)\), for both small and large-sized test problems. There are three levels of the arrival time range factor \((p = 0.1; 0.2; 0.3)\) for small and large-sized test problems. The interval of arrival time of the trucks at the cross-docking center remains very narrow because there is no great difference between arrival times of the trucks at the cross-docking terminal.

Small and large-sized problem sets are randomly generated considering the total number of inbound and outbound trucks. Small test problems with less than or equal to 7 inbound and 7 outbound trucks are solved by using GAMS 23.3. The optimum
solutions determined by the CPLEX 12.1 solver of small-sized problems are compared with the solutions of TS and SA in Table 1. The termination criterion is the iteration number and it is same for each algorithm. The initial temperature is 100°C and the cooling ratio is 90% for the SA algorithm. As a result of computational experiments, tabu parameters are obtained for the TS algorithm. The size of the tabu list and the size of the neighborhood are used in 4 and 5 in small-sized test problems, respectively. Each test problem is solved ten times by using both algorithms. Meta-heuristics algorithms are coded in Python 3.4 software. All experiments are performed on a PC with 3.0 GHz Intel Core i7 processor and 12 GB RAM.

Table 1 includes the results of problems with less than or equal to 7 inbound trucks and 7 outbound trucks. The optimum solutions obtained by the CPLEX solver are compared with the solutions of SA and TS algorithms. The optimum solutions obtained using the CPLEX solver and the relative percentage deviation (RPD) of the meta-heuristic algorithms are shown in Table 2. The optimum solutions are found by using both meta-heuristics for 14 test problems from 18 test problems. The relative deviation of SA is 0.65 and TS is 0.66 for small test problems. Both meta-heuristics have almost the same RPD values.

According to Table 1, it takes a long time to find an optimum solution for a problem greater than 6 inbound and 6 outbound trucks in the CPLEX solver. When the number of trucks increases, computational time increases dramatically. Lee, Kim and Joo [19] point out that the optimization tool does not give

<table>
<thead>
<tr>
<th>Test problems</th>
<th>I</th>
<th>O</th>
<th>R</th>
<th>S</th>
<th>(α, β)</th>
<th>Objective value</th>
<th>Computational time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p CPLEX SA TS</td>
<td>p CPLEX SA TS</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>(0; 2)</td>
<td>0.1 0 0 0</td>
<td>0.57 17.00 84.00</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>(0.25; 1.75)</td>
<td>0.2 133 133 133</td>
<td>0.55 18.00 93.00</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>(0.5; 1.5)</td>
<td>0.3 237 237 237</td>
<td>0.64 17.00 91.00</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>(0; 2)</td>
<td>0.1 21 21 21</td>
<td>1.47 23.00 112.00</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>(0.25; 1.75)</td>
<td>0.2 65 65 65</td>
<td>2.28 21.00 114.00</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>(0.5; 1.5)</td>
<td>0.3 473 481 481</td>
<td>3.53 22.00 118.00</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>(0; 2)</td>
<td>0.1 0 0 0</td>
<td>0.22 21.00 122.00</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>(0.25; 1.75)</td>
<td>0.2 0 0 0</td>
<td>0.52 23.00 124.00</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>(0.5; 1.5)</td>
<td>0.3 47 47 47</td>
<td>0.96 24.00 128.00</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>(0; 2)</td>
<td>0.1 0 0 0</td>
<td>0.38 19.00 117.00</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>(0.25; 1.75)</td>
<td>0.2 15 15 15</td>
<td>3.56 20.00 108.00</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>(0.5; 1.5)</td>
<td>0.3 374 374 374</td>
<td>3.38 21.00 110.00</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>(0; 2)</td>
<td>0.1 0 0 0</td>
<td>32.09 28.00 152.00</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>(0.25; 1.75)</td>
<td>0.2 261 261 261</td>
<td>74.70 30.00 155.00</td>
</tr>
<tr>
<td>15</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>(0.5; 1.5)</td>
<td>0.3 712 730 730</td>
<td>476.67 30.00 158.00</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>(0; 2)</td>
<td>0.4 459 478 478</td>
<td>1650.04 23.00 153.00</td>
</tr>
<tr>
<td>17</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>(0.25; 1.75)</td>
<td>0.2 483 483 483</td>
<td>1483.16 27.00 146.00</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>(0.5; 1.5)</td>
<td>0.3 985 1017 1020</td>
<td>14354.84 29.00 157.00</td>
</tr>
</tbody>
</table>

Average 22.94 124.44

Table 2 RPD (%) values for small-sized problems

<table>
<thead>
<tr>
<th>Test problems</th>
<th>CPLEX Optimal Solution</th>
<th>RPD (%)</th>
<th>Test problems</th>
<th>CPLEX Optimal Solution</th>
<th>RPD (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SA TS</td>
<td></td>
<td></td>
<td>SA TS</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0 0.00 0.00 0.00</td>
<td></td>
<td>10</td>
<td>0 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>133 0.00 0.00 0.00</td>
<td></td>
<td>11</td>
<td>15 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>237 0.00 0.00 0.00</td>
<td></td>
<td>12</td>
<td>374 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>21 0.00 0.00 0.00</td>
<td></td>
<td>13</td>
<td>0 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>65 0.00 0.00 0.00</td>
<td></td>
<td>14</td>
<td>261 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>473 1.69 1.69 1.69</td>
<td></td>
<td>15</td>
<td>712 2.53 2.53 2.53</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0 0.00 0.00 0.00</td>
<td></td>
<td>16</td>
<td>459 4.14 4.14 4.14</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0 0.00 0.00 0.00</td>
<td></td>
<td>17</td>
<td>483 0.00 0.00 0.00</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>47 0.00 0.00 0.00</td>
<td></td>
<td>18</td>
<td>985 3.25 3.53 3.55</td>
<td></td>
</tr>
</tbody>
</table>

Average 0.65 0.66

RPD (%) = Solution of the meta heuristic − Optimal solution
          Optimal solution × 100

results for problems over 6 inbound and 6 outbound trucks exactly because of the long computational time. The results in Table 1 support the above-indicated study. Objective values of the SA and TS algorithms and computational times show that SA and TS provide suitable results in a reasonable time. They can thus be used for large-sized problems.
Data sets for large-sized problems are used for comparing the relative performance of SA and TS in Table 3. The initial temperature is 100°C and the cooling ratio is 90% for the SA algorithm. As a result of computational experiments, tabu parameters are obtained for the TS algorithm. The size of the tabu list and the size of the neighborhood are used in 6 and 9, respectively. Each test problem is solved ten times using both algorithms.

In Table 3, the algorithms are compared with each other to show their performance. TS outperforms SA for the large-sized test problems. It is observed that TS gives better results than SA when the number of trucks increases. However, the SA algorithm consumes less computational time than the TS algorithm when the iteration number is the same for each algorithm. Table 4 shows the relative percentage deviation (RPD) of the meta-heuristic algorithms.

Table 4 shows that the RPD values of the SA algorithm are greater than the RPD values of the TS algorithm. While the RPD value average of the SA algorithm is 11.18%, the RPD value average of the TS algorithm is only 2.73%. The RPD value average of the TS algorithm is 11.18%, the RPD value average of the SA algorithm is 11.18%, the RPD value average of the TS algorithm is only 2.73%. The RPD value average of the SA algorithm is 11.18%, the RPD value average of the TS algorithm is only 2.73%.
values of the two algorithms are shown in Fig. 6 with interval plots at a 95% confidence level.

Graph (a) in Fig. 6 shows that there are statistically significant differences between RPD values of the SA and TS algorithms. Graph (b) and graph (c) also indicate the differences between two algorithms with the different time windows and the number of trucks, respectively. These results show that TS manifests better performance for the truck scheduling problem with any truck number and any \((\alpha, \beta)\) values of time windows.

The results indicate that the computational time increases when the number of trucks increases for each algorithm. Figure 7 shows the changes in the average computational times of SA and TS in terms of the number of trucks. The problem has several characteristics in common with the parallel machine scheduling problem. Similar results are shown in the parallel machine scheduling study [27].

Figure 7 indicates the relation between the number of trucks and the computational time by comparing the algorithms. It can be observed that SA takes less time than TS. For large-sized problems, SA provides suitable results with an average computational time (102.23 seconds) that is 87% shorter than that of the TS.

The decision maker generates the schedules by using SA or TS algorithms in DSS. He/she can select an appropriate schedule by comparing the results of the algorithms. Fig. 7 shows a screenshot of the system for solving the problem (Fig. 8).

<table>
<thead>
<tr>
<th>Test problems</th>
<th>Best solution of the two meta-heuristics</th>
<th>RPD (%)</th>
<th>Test problems</th>
<th>Best solution of the two meta-heuristics</th>
<th>RPD (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2027</td>
<td>14.90</td>
<td>16</td>
<td>3892</td>
<td>4.19</td>
</tr>
<tr>
<td>2</td>
<td>2675</td>
<td>0.00</td>
<td>17</td>
<td>3906</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>3290</td>
<td>19.54</td>
<td>18</td>
<td>3566</td>
<td>29.98</td>
</tr>
<tr>
<td>4</td>
<td>1692</td>
<td>5.56</td>
<td>19</td>
<td>2717</td>
<td>8.94</td>
</tr>
<tr>
<td>5</td>
<td>1917</td>
<td>18.10</td>
<td>20</td>
<td>3114</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>3142</td>
<td>24.47</td>
<td>21</td>
<td>5025</td>
<td>10.89</td>
</tr>
<tr>
<td>7</td>
<td>1362</td>
<td>28.49</td>
<td>22</td>
<td>3272</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>1507</td>
<td>29.79</td>
<td>23</td>
<td>4137</td>
<td>0.00</td>
</tr>
<tr>
<td>9</td>
<td>2505</td>
<td>28.49</td>
<td>24</td>
<td>5234</td>
<td>14.04</td>
</tr>
<tr>
<td>10</td>
<td>3043</td>
<td>19.54</td>
<td>25</td>
<td>2915</td>
<td>11.37</td>
</tr>
<tr>
<td>11</td>
<td>3701</td>
<td>18.10</td>
<td>26</td>
<td>3303</td>
<td>14.26</td>
</tr>
<tr>
<td>12</td>
<td>5735</td>
<td>24.47</td>
<td>27</td>
<td>5646</td>
<td>2.82</td>
</tr>
<tr>
<td>13</td>
<td>3143</td>
<td>28.49</td>
<td>28</td>
<td>3382</td>
<td>32.20</td>
</tr>
<tr>
<td>14</td>
<td>3351</td>
<td>28.49</td>
<td>29</td>
<td>5278</td>
<td>14.48</td>
</tr>
<tr>
<td>15</td>
<td>5306</td>
<td>28.49</td>
<td>30</td>
<td>6447</td>
<td>5.63</td>
</tr>
</tbody>
</table>

| Average       | 11.18                                   | 2.73    |               |                                         |         |

RPD (%) = \(\frac{\text{Solution of the meta heuristic} - \text{Best solution of the two meta-heuristics}}{\text{Best solution of the two meta-heuristics}} \times 100\)
The user can set the algorithm’s parameters such as the temperature and cooling ratio or the size of the tabu list and neighborhood. In Fig. 8, the highlighted rows show the best solutions iteration by iteration. The user can choose the best solution so far from the last highlighted row for overall earliness and tardiness. The user can monitor the number of products of type $k$ that are transferred from inbound truck $i$ to outbound truck $j$, the start time of unloading for inbound truck $i$, the completion time of unloading for inbound truck $i$, the start time of loading for outbound truck $j$, the completion time of loading for outbound truck $j$, and the sequences of the trucks at the doors as shown in Fig. 9. The outbound time table shows which outbound trucks
fit into the time windows. If an outbound truck doesn’t fit the time window, an error (earliness or tardiness) value is shown in the last column (Fig. 9).

In multi-door cross-docking systems, it is essential to coordinate activities such as unloading and loading and product assignment. The proposed DSS makes it possible to solve the truck scheduling problem by assigning products from inbound trucks to outbound trucks.

5. Conclusion

This paper focuses on a truck scheduling problem within pre-defined time windows in multi-door cross-docking centers. Since the punctuality and accurateness of product deliveries affect the performance of the cross-docking system, the predominant objective is to minimize earliness and tardiness. There are different arrival times for all trucks and due time intervals of outbound trucks that should be taken from customers. In this study, the problem is formulated as a mixed integer programming model to find an optimum solution. The model is used to evaluate the performance of the meta-heuristic algorithms for small-sized problems.

The numbers of product types loaded onto inbound trucks and needed for outbound trucks are known by using the RFID technology at cross-docking centers. Therefore, it is possible to assign products from inbound trucks to outbound trucks by using the information about products on inbound trucks. Product assignment is then effected by means of using a proposed product-assignment algorithm. The product-assignment algorithm uses the information obtained as a parameter to be subjected to the RFID technology.

The truck scheduling problem is NP-hard. SA and TS meta-heuristics are proposed to solve the large-sized problems. Experimental results show that TS manifests better performance for the truck scheduling problem with any truck number and any ($\alpha$, $\beta$) values of time windows. According to the relation between the number of trucks and computational time when comparing the algorithms, it can be noted that SA takes less time than TS for the same iteration number.

In the cross-docking systems, it is required to coordinate activities such as unloading and loading and product assignment. A DSS is designed and proposed for solving truck scheduling problems to minimize overall earliness and tardiness for outbound trucks within time windows. The proposed DSS makes it possible to solve the truck scheduling problem by assigning products from inbound trucks to outbound trucks. The decision maker generates the schedules by using SA or TS algorithms in DSS. He/she can select an appropriate schedule by comparing the results of the algorithms.

For future work, the cross-docking system with time windows will be able to be modelled as multi-objective. Product and door constraints will then be added to the model.

Acknowledgements. This work is supported by the Committee of Eskisehir Osmangazi University Scientific Research Projects under the Project Code 2015–734, titled “Door Assignment and Truck Scheduling in Cross-Docking”.

References


