Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 42 (2020) 475–482

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

International Conference on Industry 4.0 and Smart Manufacturing (ISM 2019)

Managing loading and discharging operations at cross-docking terminals

M. Flavia Monaco[a,*], Marcello Sammarra[b]

[a]*Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica, Università della Calabria, Via P. Bucci 44E, 87036 Rende (CS), Italy*
[b]*Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche , Via P. Bucci 7-8C, 87036 Rende (CS), Italy*

Abstract

A cross-docking terminal is a relevant node within a distribution chain. Actually, at this intermediate logistic platform between suppliers and retailers, incoming flows of possible different commodities are consolidated into single shipments, with respect to the retailers' orders, and directly delivered, skipping thus the storage phase. In such a context the synchronization of the inbound and outbound trucks is a necessary condition to guarantee fast and congestion-free transshipment operations. In this paper we propose a Mixed Integer Linear Program and a heuristic algorithm for managing the loading and discharging operations, with the aim of minimizing the completion time of the whole transshipment process.

Keywords: scheduling; heuristic; logistics

1. Introduction and related works

In the last two decades, cross-docking strategies have gained a key role in logistics industries. As underlined in [9], nowadays customers, in a broad sense, have become *even more volatile and impatient than before*. This high pressure by customers, calling for fast deliveries of their orders, is completely dumped on suppliers and manufacturers, in general. However, distribution processes based on cross-docking strategies present benefits both for customers and suppliers.

A cross-docking terminal is a logistic platform, where products arriving by inbound trucks, are arranged with respect to customers' requirements and loaded into the outbound trucks, possibly skipping the storage phase or at most being stored for a short amount of time (typically less than 24 hours). By reducing the inventory costs, cross-docking have a positive effect for the suppliers, given that it allows to reduce the distribution costs. On the other hand, the absence of the inventory phase in the distribution process allows to reduce the customers' lead times, so having positive effects also on them. In some sense, cross-docking can be viewed as the counterpart, on the logistic side, of the well established *Just in time* strategy in production planning, where the objective of reducing as much as possible the inventory levels is pursued.

The scientific literature on cross-docking problems is very wide, and also a summary analysis goes behind the aim of this paper. We refer the interested reader to the review papers [2, 11, 3, 8]. We just want to underline that, among all the problems arising in the management of a cross-docking terminal, in this paper we are concerned with one of the most relevant ones at the operational decision level, that is the synchronization of inbound and outbound trucks, also known as *truck scheduling problem*.

Generally, truck scheduling problems can be divided into two main classes: scheduling of inbound trucks, while it is assumed that the outbound trucks are already scheduled or assigned on a midterm horizon; scheduling of inbound and outbound trucks. Our problem belongs to the second class. Therefore, our aim is the coordinate scheduling of the inbound and outbound trucks at a cross-docking terminal, with the objective of minimizing the time needed to complete all the required operations. As in [4, 7], we consider a basic layout constituted by two doors (or gates), one for the discharging and one for the loading operations. However, differently from the previous cited papers, the transshipment flows between inbound and outbound trucks are explicitly considered as part of the decision problem, while in [4, 7] they are assumed to be known.

The paper is organized as follows. In Section 2 we detail the problem and provide a mixed integer linear program. The Lagrangian decomposition technique is discussed in Section 3,

* Corresponding author. Tel.: +39-0984-494709 ; fax: +39-0984-494781
E-mail address: monaco@dimes.unical.it (M. Flavia Monaco).

10.1016/j.promfg.2020.02.045

that is followed by the description of the Lagrangian heuristics in Section 4. Numerical results are discussed in Section 5. Finally, conclusions are drawn in Section 6.

2. Mathematical Model

We consider a cross-docking terminal with one door at both inbound and outbound sides. A truck at a time can be processed at the docking doors. The products shipped from a given set of inbound trucks have to be unloaded at the first door, sorted on the basis of the customer demands, moved across the dock, and then loaded onto a set of outboubd trucks, at the second door. We assume, as it is usual in this context, that all trucks are ready at the beginning of the planning horizon, their processing times are known, while the transshipment time inside the terminal is negligible. Moreover, preemption in the unloading/loading of each truck is not allowed. Therefore, the processing of an outbound truck cannot start before the unloading of all inbound trucks carrying loads for it has been completed. The problem is to decide the unloading and loading sequences of the trucks so as to minimize the completion time of the whole process, that coincides with the time at which the last outbound truck has been completely loaded and can leave the terminal (*makespan*).

Here is the list of the notation we adopt to formulate the truck-scheduling problem.

Nomenclature

Main Notation

I	Set of inbound trucks, $\|I\| = n$
J	Set of outbound trucks, $\|J\| = m$
P	Set of products shipped from the inbound to the outbound trucks, $\|P\| = c$
K	Set of sequence positions at the inbound gate, $\|K\| = n$
K^1	$K \setminus \{1\}$
H	Set of sequence positions at the outbound gate, $\|H\| = m$
H^1	$H \setminus \{1\}$
t_i	Processing time of the inbound truck $i \in I$
τ_j	Processing time of the outbound truck $j \in J$
M^I	Sum of the inbound trucks processing times, $M^I = \sum_{i \in I} t_i$
M	Sum of all trucks processing times, $M = M^I + \sum_{j \in J} \tau_j$
d_{ip}	quantity of product $p \in P$ delivered by truck $i \in I$
r_{jp}	quantity of product $p \in P$ required by truck $j \in J$
u_{ij}^p	maximum quantity of product $p \in P$ transferable from $i \in I$ to $j \in J$, $u_{ij}^p = \min\{d_{ip}, r_{jp}\}$

We define the following decision variables:

Nomenclature

Decision Variables

$x_{ik} = 1$ if the truck $i \in I$ is the $k-$th one in the inbound sequence, 0 otherwise
$y_{jh} = 1$ if the truck $j \in J$ is the $h-$th one in the outbound sequence, 0 otherwise
$z_{ij}^p \geq 0$ quantity of product $p \in P$ shipped from truck $i \in I$ to truck $j \in J$
$v_{ij}^p = 1$ if a strictly positive amount of product $p \in P$ is shipped from $i \in I$ to $j \in J$, 0 otherwise
$S_i^I \geq 0$ starting processing time of the truck $i \in I$
$C_i^I \geq 0$ completion time of the truck $i \in I$
$S_j^O \geq 0$ starting processing time of the truck $j \in J$
$C_j^O \geq 0$ completion time of the truck $j \in J$
C_{max} the makespan, $C_{max} = \max_{j \in J}\{C_j^O\}$

Note that the variables S_i^I and S_j^O have been introduced for sake of readability, even though they are redundant (see constraints (5) and (9) in the model). With the above notation, the Mixed Integer Linear Model for the truck-scheduling problem under investigation is the following:

$$Z = \min\ C_{max} \tag{1}$$

$$\sum_{k \in K} x_{ik} = 1 \qquad i \in I \tag{2}$$

$$\sum_{i \in I} x_{ik} = 1 \qquad k \in K \tag{3}$$

$$S_i^I \geq C_l^I - M^I(2 - x_{ik} - x_{l\,k-1}) \qquad i \neq l \in I, k \in K^1 \tag{4}$$

$$C_i^I = S_i^I + t_i \qquad i \in I \tag{5}$$

$$\sum_{h \in H} y_{jh} = 1 \qquad j \in J \tag{6}$$

$$\sum_{j \in J} y_{jh} = 1 \qquad h \in H \tag{7}$$

$$S_j^O \geq C_l^O - M(2 - y_{jh} - y_{l\,h-1}) \qquad j \neq l \in J, h \in H^1 \tag{8}$$

$$C_j^O = S_j^O + \tau_j \qquad j \in J \tag{9}$$

$$\sum_{j \in J} z_{ij}^p = d_{ip} \qquad i \in I, p \in P \tag{10}$$

$$\sum_{i \in I} z_{ij}^p = r_{jp} \qquad j \in J, p \in P \tag{11}$$

$$z_{ij}^p \leq u_{ij}^p v_{ij}^p \qquad i \in I, j \in J, p \in P \tag{12}$$

$$S_j^O \geq +C_i^I - M^I(1 - v_{ij}^p) \qquad i \in I, j \in J, p \in P \tag{13}$$

$$C_{max} \geq C_j^O \qquad j \in J \tag{14}$$

$$x_{ik} \in \{0, 1\} \qquad i \in I, k \in K \tag{15}$$

$$y_{jh} \in \{0, 1\} \qquad j \in J, h \in H \tag{16}$$

$$v_{ij}^p \in \{0, 1\} \qquad i \in I, j \in J, p \in P \tag{17}$$

$$z_{ij}^p \geq 0 \qquad i \in I, j \in J, p \in P \qquad (18)$$
$$S_i^I \geq 0 \qquad i \in I \qquad (19)$$
$$S_j^O \geq 0 \qquad j \in J \qquad (20)$$

In this model, constraints (2) and (3) are the standard assignment constraints. They impose that each inbound truck i is assigned to one and only one position k in the sequence of trucks to be discharged at the inbound door. Constraints (4) and (5) allow to compute the completion times for the inbound trucks I. In particular, constraints (4) ensure that S_i^I is not smaller than C_l^I, whenever l and i are consecutive in the inbound sequence; otherwise they are redundant. Constraints (6) and (7) are the same as (2) and (3) for the truck sequence at the outbound door. Constraints (8) and (9) play the same role of (4) and (5), since they are needed to compute the completion times of the outbound trucks J. Note that, since M is an upper bound on C_{max}, constraints (8) are trivially satisfied for each pair of trucks that are not consecutive in the outbound sequence. Equations (10) and (11) are the flow conservation constraints, and ensure that all the products entering the cross-docking centre will be correctly delivered to the outbound trucks. Constraints (12) logically link z's and v's variables, imposing that $z_{ij}^p = 0$ if product p is not sent from the inbound truck i to the outbound truck j; conversely, if $v_{ij}^p = 1$, then the flow variable z_{ij}^p cannot be greater than u_{ij}^p. Constraints (13) are the cross-docking constraints. They impose the correct relation between the start processing time of the outbound truck j and the completion time of the inbound truck i, whenever they are involved in the exchange of some products, while they are redundant when $v_{ij}^p = 0$. Finally, constraints (14) are the definition of the objective function (1), and (15) - (20) define the domain of the variables.

3. The Lagrangian Relaxation of the model

It is easy to recognize that relations (13) play the role of coupling constraints, that is *complicating* constraints in a Lagrangian Relaxation framework. If they are relaxed and dualized in the objective function, the resulting problem will decompose in three subproblems. To this aim, let λ be a vector of Lagrangian multipliers associated to constraints (13), such that $\lambda_{ij}^p \geq 0,\ \forall i \in I,\ j \in J,\ p \in P,\ \lambda_{ij}^p = 0$ if $u_{ij}^p = 0$. The Lagrangian relaxed problem is the following:

$$Z_{LR}(\lambda) = \min \left\{ C_{max} + \sum_{i,j,p} \lambda_{ij}^p \left(C_i^I - M^I(1 - v_{ij}^p) - S_j^O \right) \right\} \quad (21)$$
$$s.t. \quad (2) - (12), (14) - (20)$$

It is well known that, for each choice of the Lagrangian multipliers vector $\lambda \geq 0$, the optimal value of the Lagrangian relaxed problem provides a lower bound on the optimal value of problem (1)-(20), i.e.: $Z \geq Z_{LR}(\lambda)$.

Defining

$$\rho_i = \sum_{j \in J} \sum_{p \in P} \lambda_{ij}^p \qquad \forall i \in I \qquad \rho = (\rho_1, \ldots, \rho_n)^\top \qquad (22)$$
$$\sigma_j = \sum_{j \in J} \sum_{p \in P} \lambda_{ij}^p \qquad \forall j \in J \qquad \sigma = (\sigma_1, \ldots, \sigma_m)^\top \qquad (23)$$

$$s = \sum_{j \in J} \sigma_j \tau_j - M^I \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} \lambda_{ij}^p \qquad (24)$$

the optimal value of the Lagrangian Relaxed problem (21) can be written in the following form:

$$Z_{LR}(\lambda) = s + Z_{LR}^I(\rho) + Z_{LR}^O(\sigma) + M^I Z_{LR}^T(\lambda) \qquad (25)$$

where $Z_{LR}^I(\rho)$, $Z_{LR}^O(\sigma)$ and $Z_{LR}^T(\lambda)$ are, respectively, the optimal values of the following three subproblems:

$$PI(\rho) \begin{cases} Z_{LR}^I(\rho) = \min \sum_{i \in I} \rho_i C_i^I \\ s.t. \quad (2) - (5), (15), (19) \end{cases} \qquad (26)$$

$$PO(\sigma) \begin{cases} Z_{LR}^O(\sigma) = \min \left(C_{max} - \sum_{j \in J} \sigma_j C_j^O \right) \\ s.t. \quad (6) - (9), (14), (16), (20) \end{cases} \qquad (27)$$

$$PT(\lambda) \begin{cases} Z_{LR}^T(\lambda) = \min \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} \lambda_{ij}^p v_{ij}^p \\ s.t. \quad (10) - (12), (17), (18) \end{cases} \qquad (28)$$

$PI(\rho)$ is a single machine scheduling problem at the inbound door: $1||\sum_i w_i C_i$, where $w_i = \rho_i$ and $C_i = C_i^I$, $i \in I$ (we note, in passing, that constraints (4) and (5) can be omitted). Therefore, its optimal solution is obtained applying the Weighted Shortest Processing Time (WSPT) [10].

Similarly for the outbound door, $PO(\sigma)$ is a single machine scheduling with a non standard objective function: $1||(C_{max} - \sum_j w_j C_j)$, where $w_j = \sigma_j$ and $C_j = C_j^O$, $j \in J$. However, it is possible to prove that, whenever $\sum_{j \in J} \sigma_j \leq 1$, the optimal schedule can be computed by applying the Weighted Longest Processing Time (WLPT) rule [10], while the problem is unbounded from below when $\sum_{j \in J} \sigma_j > 1$.

Finally, subproblem $PT(\lambda)$ is a multi-commodity transportation problem with variable upper bound constraints, that naturally separates into $c = |P|$ single commodity problems, one for each product $p \in P$. For a fixed $p \in P$, the problem is to move the product p from the inbound trucks $i \in I$, each with a given supply d_{ip}, to the outbound trucks $j \in J$, each demanding r_{jp}, satisfying supply and demand constraints. If a positive amount of commodity p is transported from i to j, then it can not exceed u^p_{ij}, and a fixed cost λ^p_{ij} has to be paid. The objective function to minimize is the sum of the fixed costs. Therefore each single commodity subproblem is a pure fixed charge transportation problem [1, 6], and thus it is hard to be solved to optimality, due to its combinatorial structure. A heuristic procedure for solving $PT(\lambda)$ for a given $p \in P$ is described in [1]. It consists in relaxing constraints (12), solving a standard transportation problem with unit transportation costs $w^p_{ij} = \lambda^p_{ij}/u^p_{ij}$, and then setting $v^p_{ij} = 1$ if at least one unit of product p is transported from i to j.

4. Lagrangian heuristics

Solving $PI(\rho)$, $PO(\sigma)$, and $PT(\lambda)$ for a given set of non negative Lagrangian multipliers, such that $\sum_{ijp} \lambda^p_{ij} \leq 1$ (see the discussion on $PO(\sigma)$ in the previous Section), we get the Lagrangian solutions x, C^I, y, C^O, z, and v. If C^I, C^O, and v satisfy constraints (13), they also give a feasible solution to the model (1)-(20), of value $\bar{C}_{max} = \max_{j\in J}\left\{C^O_j\right\}$.

In case C^I, C^O, and v violate some of the relaxed constraints (13), a feasible solution can be obtained computing $\bar{y}, \bar{C}^O$, $\bar{C}_{max}$ by very simple recovering heuristics and setting $\bar{x} = x, \bar{C}^I = C^I, \bar{v} = v, \bar{z} = z$. In the first recovering heuristics H1, the Lagrangian solution y, C^O is transformed in a feasible solution simply by increasing the completion time of each outbound truck by the same amount, while taking the same sequence provided by y. In the second recovering heuristics H2, the outbound trucks are first sorted by the earliest starting processing times which are compatible with the cross-docking constraints, and then re-scheduled. The steps of H1 and H2 are detailed in Algorithms 1 and 2 reported in Appendix A.

Our Lagrangian heuristics LH consists in solving, within an iterative scheme, the Lagrangian sub-problems $PI(\rho)$, $PO(\sigma)$, and $PT(\lambda)$, executing the recovering algorithms H1 and H2, and varying the Lagrangian multipliers at each iteration. By this way many feasible solutions to the cross-docking problem defined by model (1)-(20) are generated and, at the end, LH returns the best one among them. We use the following rule for varying the Lagrangian multipliers from iteration r to iteration $r + 1$:

$$\lambda^p_{ij}(r+1) = \max\left\{0, \lambda^p_{ij}(r) + t(r)\gamma^p_{ij}(r)\right\} \tag{29}$$

In equation (29), $\gamma^p_{ij}(r)$ is defined by

$$\gamma^p_{ij}(r) = -S^O_j(r) + C^I_i(r) - M^I(1 - v^p_{ij}(r)) \tag{30}$$

and $t(r)$ is a step-size defined as

$$t(r) = \alpha \frac{C^*_{max} - LB(r)}{\sqrt{\sum_{i\in I}\sum_{j\in J}\sum_{p\in P}\gamma^p_{ij}(r)^2}} \tag{31}$$

where α is a smoothing parameter, $LB(r)$ is the lower bound returned by solving the Lagrangian problem (21) at the r-th iteration, and C^*_{max} is the best makespan found so far. Observe that $\gamma^p_{ij}(r)$ is the negative of the slack variable in the (i, j, p) constraint (13) at the current Lagrangian solution, and corresponds to the (i, j, p) component of a subgradient vector of the Lagrangian function evaluated at the current solution [5]. From this point of view, LH is actually a *subgradient algorithm* for maximizing $Z_{LR}(\lambda)$ in problem (21), that is for solving the so called *Lagrangian Dual problem*. Therefore, LH returns also the best lower bound, which allows to measure the quality of the best computed feasible solution. The pseudocode of LH is reported in Algorithm 3 of Appendix A.

5. Numerical results

5.1. Description of the test suite

To test our algorithm, we have considered two sets of instances A and B. As for the set A, it consists of 250 instances of different dimensions in terms of n and m. For each possible dimension, 10 instances are considered. The instances of the set A have been proposed in [7] for a truck scheduling problem where the transshipment plan is already known, and therefore for a problem that is substantially different from the one we are dealing with. However, these instances can be turned into suitable instances fitting the model and the algorithm presented in this paper. Such a transformation is detailed in Appendix B. We summarize in Table 1 the dimensions of the instances in A, after the procedure of transformation, and we refer the reader to the above cited paper for further details. Here we just remark that $t_i, i \in I$ and $\tau_j, j \in J$ are uniformly distributed in the range 1 to 10.

Table 1. Dimensions of instances in the set A.

	$\|I\|$	$\|J\|$	$\|P\|$	Total instances
A_1	$n = 5$	$m \in \{3, 4, 5, 6, 7\}$	$c = m$	50
A_2	$n = 10$	$m \in \{6, 8, 10, 12, 14\}$	$c = m$	50
A_3	$n = 20$	$m \in \{12, 16, 20, 24, 28\}$	$c = m$	50
A_4	$n = 40$	$m \in \{24, 32, 40, 48, 56\}$	$c = m$	50
A_5	$n = 60$	$m \in \{36, 48, 60, 72, 84\}$	$c = m$	50

Since the instances of Set A are intrinsically representative of a particular and easier case of our truck scheduling problem, we have generated a new set of instances (set B), where the products arriving by the inbound trucks must be optimally transferred to the outbound trucks. The set B consists of three subsets of 40 instances each, differing in the values of n, m, and c as detailed in Table 2. To obtain the test suite B, the values of n, m, c of each row in Table 2 have been fully combined each other. Then, for each possible combination, 10 different instances have been generated by choosing

- $t_i, i \in I$ and $\tau_j, j \in J$ uniformly distributed in the range 1 to 10;
- $d_{ip}, i \in I, p \in P$ uniformly distributed in the range 0 to 1000;
- $r_{jp}, j \in J, p \in P$ uniformly distributed in the range 0 to $\sum_{i \in I} d_{ip}$, in such a way that $\sum_{i \in I} d_{ip} = \sum_{j \in J} r_{jp},\ p \in P$;

Table 2. Dimensions of instances in the set B.

	$\|I\|$	$\|J\|$	$\|P\|$	Total instances
B_1	$n \in \{10, 15\}$	$m \in \{10, 15\}$	$c = 3$	40
B_2	$n \in \{15, 20\}$	$m \in \{15, 20\}$	$c = 5$	40
B_3	$n \in \{30, 40\}$	$m \in \{30, 40\}$	$c = 7$	40

5.2. Implementation details

The LH Algorithm has been coded in C++ and Cplex 12.8 has been used for benchmarking purposes. We have ran the experiments on a machine equipped with a 3.1 GHz CPU and 16GB of RAM, giving Cplex one hour of time limit and letting the LH Algorithm to perform at most 500 iterations. The initial Lagrangian multipliers have been set as follows

$$\lambda_{ij}^{p}(0) = \begin{cases} \dfrac{1}{nmc} & \forall i \in I, j \in J, p \in P \ : u_{ij}^{p} > 0 \\ 0 & \text{otherwise} \end{cases}$$

so ensuring that their sum does not exceed one. The smoothing parameter α in equation (31) has been chosen equal to 10^{-6}. We finally recall that in solving the Lagrangian Relaxation problem, the sub-problem $PT(\lambda)$ can not be solved to optimality. As disclosed in Section 3, we solve a relaxation of $PT(\lambda)$ instead, that is a standard transportation problem. To this aim we have adopted one of the well known algorithms for determining a basic feasible solution to a transportation problem, in particular the greedy one.

5.3. Discussion of the numerical results

In order to evaluate the effectiveness of the proposed Lagrangian heuristics, we compare the results obtained by LH with those returned by Cplex. In particular this comparison is done in terms of the quality of the solutions returned by the two methods and in terms of computation times. We report in Figures 1 and 2, as average values attained on ten instances of the same dimension, the best objective function values (Upper Bound - UB), along with the best lower bounds (LB) on the optimal makespan, computed by Cplex and LH. The average computation times, for each subsets of instances, are summarized in Table 3.

Looking at Figure 1, we observe that Cplex and LH have basically the same behaviour for $3 \leq c \leq 32$, as far as the upper bounds are concerned. In particular, for $3 \leq c \leq 7$, i.e. for the A_1 instances, the best upper bound and the best lower bound values computed by Cplex coincide, meaning that for these set of instances Cplex always returned the optimal solution. This fact is confirmed by the computation times for the A_1 instances in Table 3. On the same instances LH performs as well as Cplex in terms of UB values, while the Lagrangian lower bound is smaller than the best lower bound computed by Cplex. Starting from the smallest instances in the set A_2, the quality of the solutions returned by Cplex deteriorates and the number of optimal solutions drastically reduces. Actually, for these instances Cplex returns four optimal solutions and no optimal one in all the remaining instances of set A (see also Table 3). For $c \geq 32$ our algorithm outperforms Cplex in terms of both UB and LB values, by an amount that increases as the instance dimension increases. In particular, from Figure 1 it is evident that the best lower bound returned by Cplex is really weak and, as a consequence, the corresponding optimality gap results to be very high. On the contrary, the Lagrangian LB returned by LH is significantly higher, so producing lower values of the optimality gap and, therefore, a more accurate estimation of the solution quality.

Table 3. Average computation time (seconds).

Set	Cplex	LH
A_1	1.53	0.06
A_2	3483.87	0.38
A_3	3600.00	3.83
A_4	3600.00	48.42
A_5	3600.00	221.15
B_1	3565.49	0.19
B_2	3600.00	0.71
B_3	3600.00	6.09

Passing to the results on the set B, we note that (see Figure 2) Cplex exhibits the same behaviour observed on instances of comparable size in set A. Actually, it is able to find only one optimal solution in subset B_1, while in all the other cases it reaches the imposed time limit returning solutions with very high optimality gaps. Conversely, the behaviour of LH on the instance set B is not so similar to the one observed for instances of set A. In particular, on the smallest size instances the UB values computed by LH are slightly worse than those of Cplex, while the opposite relation holds for the largest instances of the subset B_3. The difference between the two UB values, independently

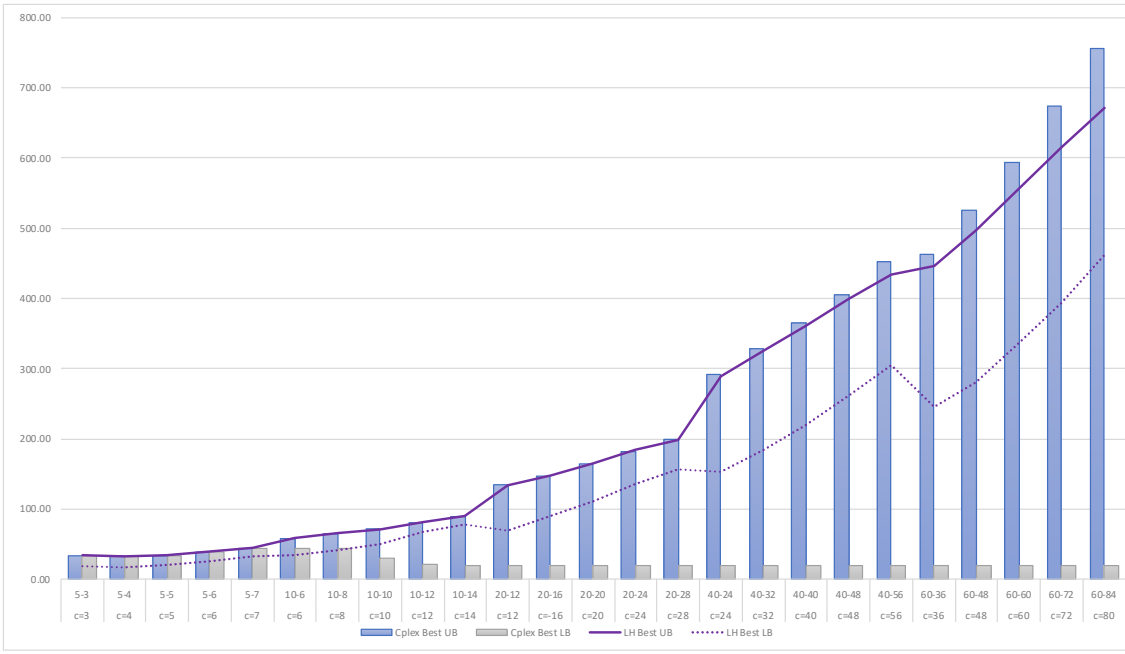

Fig. 1. Comparison of numerical results on Set A instances.

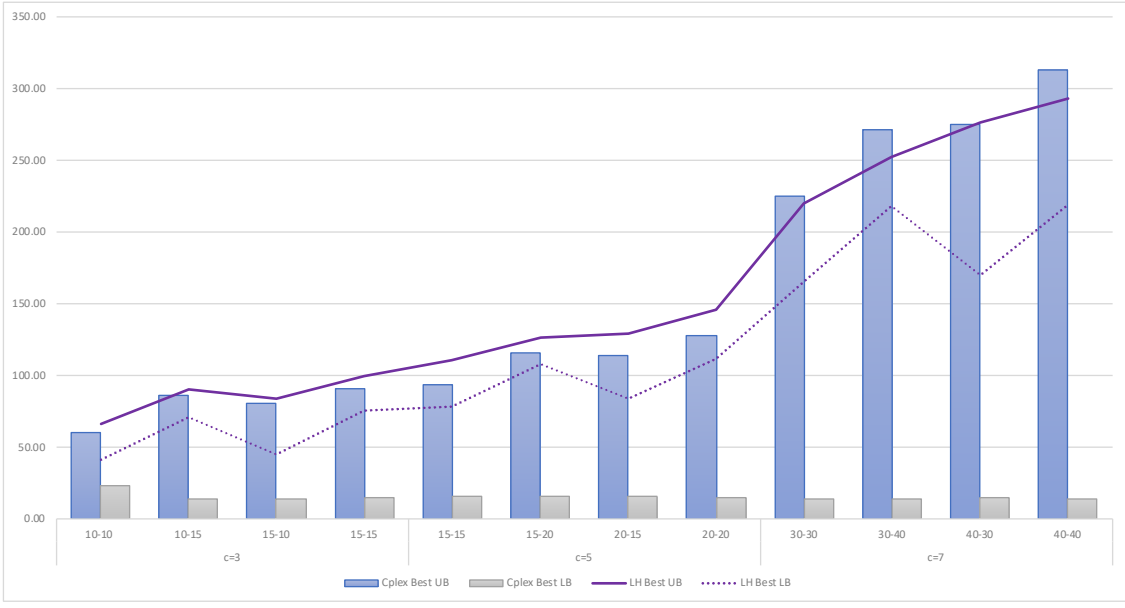

Fig. 2. Comparison of numerical results on Set B instances.

on its sign, is moderate. Of course, the less satisfactory results of LH on Set B are due to the poor accuracy we use in solving the subproblem $PT(\lambda)$ (see discussion on this issue in Section 3 and subsection 5.2). However this drawback is widely counterbalanced both by the computation times and the values of the Lagrangian lower bounds. Actually, in very short computation times (never reaching seven seconds) LH is able to compute feasible solutions of certified good quality.

6. Conclusions

In this paper we have presented Mixed Integer Linear Program and a Lagrangian heuristic algorithm for a truck schedul-

ing problem at a cross-docking terminal. By means of Lagrangian Relaxation of the model, we have been able to decompose the problem into three independent subproblems: two scheduling problems at the inbound and outbound doors of the terminal, and a multicommodity fixed charge transportation problem. Including suitable repairing heuristics in a subgradient algorithm scheme, we have designed a Lagrangian heuristic procedure and tested it on a wide set of instances. Numerical results have shown the effectiveness of our approach, both in terms of quality of the computed solutions and computation time.

Appendix A. Pseudocodes of heuristics H1, H2, and LH

Algorithm 1 Recovering Heuristics **H1**(C^I, y, C^O, v)

$\Delta^{max} = 0$
for all $i \in I, j \in J, p \in P$ **do**
  if $v_{ij}^p = 1$ **and** $C_i^I - C_j^O + \tau_j > \Delta^{max}$ **then**
    $\Delta^{max} = C_i^I - C_j^O + \tau_j$
for all $j \in J$ **do**
  $\bar{C}_j^O = C_j^O + \Delta^{max}$
$\bar{C}_{max} = \max_{j \in J} \left\{ \bar{C}_j^O \right\}$
return $\bar{y} = y$, $\bar{C}^O$, $\bar{C}_{max}$

Algorithm 2 Recovering Heuristics **H2**(C^I, y, C^O, v)

for all $j \in J$ **do**
  $\Delta_j^{min} = \max_{\substack{i \in I \\ p \in P}} \{C_i^O \mid v_{ij}^p = 1\}$ /* *minimum feasible starting processing time for truck* j */
Sort the outbound trucks $j \in J$ in increasing order of the corresponding Δ_j^{min}
Schedule the outbound trucks $j \in J$ with respect to the previous sorting; compute $\bar{C}^O$ and $\bar{y}$ accordingly.
$\bar{C}_{max} = \max_{j \in J} \left\{ \bar{C}_j^O \right\}$
return $\bar{y}$, $\bar{C}$, $\bar{C}_{max}$

Algorithm 3 Lagrangian Heuristics **LH**

$LB = -\infty$ /* *Lower bound provided by the Lagrangian problem* */
x^*, C^{I^*}, y^*, C^{O^*}, z^*, v^*, $C^*_{max} = +\infty$ /* *Best solution* */
$\bar{x}$, $\bar{C}^I$, $\bar{y}$, $\bar{C}^O$, $\bar{z}$, $\bar{v}$, $\bar{C}_{max}$ /**Solution returned by H1 and H2**/
$x(r)$, $C^I(r)$, $y(r)$, $C^O(r)$, $z(r)$, $v(r)$ /* *Lagrangian solution at iteration* r */
$r = 0$
for all $i \in I, j \in N, p \in P$ **do**
  $\lambda_{ij}^p(0) = 0$ /* *initialize the multipliers*/
while $r \leq MaxNumberOfIterations$ **do**
  Compute ρ and σ
  if $\sum_{j \in J} \sigma_j > 1$ **then**
    $r = MaxNumberOfIterations$
  else
    Solve $PI(\rho), PO(\sigma), PT(\lambda)$, getting $x(r)$, $C^I(r)$, $y(r)$, $C^O(r)$, $z(r)$, $v(r)$
    Set $\bar{x} = x(r)$, $\bar{C}^I = C^I(r)$, $\bar{z} = z(r)$, $\bar{v} = v(r)$
    $(\bar{y}, \bar{C}^O, \bar{C}_{max})$=**H1**$(C^I(r), y(r), C^O(r), v(r))$
    if $\bar{C}_{max} < C^*_{max}$ **then**
      $C^*_{max} = \bar{C}_{max}$, $x^* = \bar{x}$, $C^{I^*} = \bar{C}^I$, $y^* = \bar{y}$, C^{O^*}, $z^* = \bar{z}$, $v^* = \bar{v}$
    $(\bar{y}, \bar{C}^O, \bar{C}_{max})$=**HA2**$(C^I(r), y(r), C^O(r), v(r))$
    if $\bar{C}_{max} < C^*_{max}$ **then**
      $C^*_{max} = \bar{C}_{max}$, $x^* = \bar{x}$, $C^{I^*} = \bar{C}^I$, $y^* = \bar{y}$, C^{O^*}, $z^* = \bar{z}$, $v^* = \bar{v}$
    Update the Lagrangian multipliers by eq. (29)
    $r = r + 1$
return x^*, C^{I^*}, y^*, C^{O^*} , z^*, v^*, C^*_{max}, LB

Appendix B. The instance set A

The set of instances A has been proposed in [7], for a cross-docking problem with one inbound and one outbound door, where the transshipment flows of commodities between inbound and outbound trucks are fixed. That is, for each outbound truck $j \in J$ the subset $I_j \subseteq I$ of inbound trucks sending some load units to j is known. With reference to our model (1)-(20), under this setting constraints (10), (11), (12) and the involved variables z and v are useless, while constraints (13) become

$$S_j^O \geq C_i^I \quad \forall j \in J, i \in I_j \tag{B.1}$$

It is easy to transform the instances A so as they can be adopted by our model and algorithm. The transformation procedure preserves the original dimensions in terms of $|I| = n$ and $|J| = m$, while creating suitable values for $|P| = c$ and for $d_{ip}, r_{jp}, i \in I, j \in J, p \in P$. It is described in Algorithm 4.

Algorithm 4 Instance transformation

$|P| = |J| = m$
for all $\in I, p \in P$ **do**
  $d_{ip} = 0$
for all $j \in J, p \in P$ **do**
  $r_{jp} = 0$
for all $j \in J$ **do**
  $r_{jj} = |I_j|$;
for all $j \in J, i \in I_j$ **do**
  $d_{ij} = d_{ij} + 1$

Basically, Algorithm 4 returns an instance for our truck scheduling problem, starting from an instance defined in [7], such that:

- the number of products is equal to the number of outbound trucks, and each product is required by one and only one outbound truck.
- each inbound truck carries at most one unit of each product.

References

[1] Balinsky, M.L., 1961. Fixed cost transportation problems. Naval Research Logistics Quarterly 8, 41–54.
[2] Boysen, N., Fliedner, M., 2010. Cross dock scheduling: Classification, literature review and research agenda. Omega 38, 413–422.
[3] Buijs, P., Vis, I.F., Carlo, H.J., 2014. Synchronization in cross-docking networks: A research classification and framework. European Journal of Operational Research 239, 593–608.
[4] Chiarello, A., Gaudioso, M., Sammarra, M., 2018. Truck synchronization at single door cross-docking terminals. OR Spectrum 40, 395–447.
[5] Fisher, M.L., 1981. The lagrangian relaxation method for solving integer programming problems. Management Science 27, 1–18.
[6] Fisk, J., McKeown, P., 1979. The pure fixed charge transportation problem. Naval Research Logistics 26, 631–641.
[7] Fonseca, G.B., Nogueira, T.H., Ravetti, M.G., 2019. A hybrid lagrangian metaheuristic for the cross-docking flow shop scheduling problem. European Journal of Operational Research 275, 139–154.
[8] Ladier, A.L., Alpan, G., 2016a. Cross-docking operations: Current research versus industry practice. Omega 62, 145–162.
[9] Ladier, A.L., Alpan, G., 2016b. Robust cross-dock scheduling with time windows. Computers & Industrial Engineering 99, 16 – 28.
[10] Pinedo, M.L., 2016. Scheduling. Theory, Algorithms, and Systems. Springer, Cham.
[11] Van Belle, J., Valckenaers, P., Cattrysse, D., 2012. Cross-docking: State of the art. Omega 40, 827–846.