Operations Research Perspectives 6 (2019) 100126

Contents lists available at ScienceDirect

Operations Research Perspectives

journal homepage: www.elsevier.com/locate/orp

Output maximization container loading problem with time availability constraints

Pedro B. Castellucci*[,a,b], Franklina M.B. Toledo[b], Alysson M. Costa[a]

[a] The University of Melbourne, Australia
[b] Universidade de São Paulo, Brazil

ARTICLE INFO

Keywords:
Container loading
Cross-docking
Packing
Stochastic dynamic programming

ABSTRACT

Research on container loading problems has helped increase the occupation rate of containers in different practical situations. We consider these problems within a context which might pressure the loading process, leading to sub-optimal solutions. Some facilities like cross-docks have reduced storage space which might force early loading activities. We propose a container loading problem which accounts for this limited storage by explicitly considering the schedule of arrival for the boxes and the departure time of the trucks. Also, we design a framework which handles the geometric and temporal characteristics of the problem separately, enabling the use of methods found in the literature for solving the extended problem. Our framework can handle uncertainty in the schedule and be used to quantify the impact of delays on capacity utilization and departure time of trucks.

1. Introduction

Container loading problems have the goal of improving the effectiveness of logistic systems by increasing the capacity utilization of the trucks/containers. Its most popular version is the output maximization, which consists in selecting a subset of items (boxes) that maximize the volume (or value) loaded into a limited number of containers. However, the logistic environment can pressure the loading process to the use of less effective solutions. That is the case with cross-docking. Cross-docks are facilities with limited storage space which operate by synchronizing the inbound and outbound flow of trucks to potentially improve rates of consolidation, delivery and lead times and costs [8].

One of the most popular optimization problems in the cross-docking literature is the scheduling of inbound and outbound trucks at its docks. To provide insights on promising solution methodologies and behavior of the system, some researchers have focused on cases with one inbound and one outbound dock [2,10,12,17,28,32,44,48], but cases with multiple inbound and outbound docks have been becoming more frequent [4,7,19,31,45,47]. The scheduling problem is usually modeled with the goal of minimizing total operation time [6,7,11,17,19,23,28,32,39,44,47,48]. It is also possible to find examples of other metrics: Storage (volume or cost) or material handling [31,34,37], travel distance [18] and earliness and/or tardiness alone [4,10] or combined with others such as travel time [45] and probability of breakdowns [2].

The combination of limited storage space and pressure for the early dispatching of trucks in a cross-dock imposes additional challenges to the resulting container loading problems. The container loading literature already contemplates a wide range of practical constraints related to the weight of the containers [15,20], axle weight distribution [33,38], the position of individual boxes [13,26,40,43,46,50] and the arrangement of the boxes [1,9,27,35,42]. Two popular variations are the supported and unsupported cases. In the supported case, each box must have its bottom (partially or fully) supported either by other boxes or by the floor of the container. The unsupported case does not have this requirement. Some solution methods (e.g. Gonçalves and Resende [24] and Araya et al. [3]) can be adapted to handle both cases. For an extensive review on practical constraints and solution methodologies see [14,49], respectively. Also, Silva et al. [41] perform extensive comparative experiments on models for container loading problem. Although these practical constraints might be present in container loading in cross-docking environments, we are not aware of any research dealing with the specific needs of cross-docks: (i) considering the loading time of trucks and (ii) time availability of the boxes, due to limited storage.

In fact, the problems we could find which are related to ours the most are the Temporal Knapsack Problem and the Temporal Bin Packing Problem presented by Bartlett et al. [5], De Cauwer et al. [21], respectively. For both studies (knapsack and bin packing) their authors propose and an extension of a one dimensional bin knapsack and bin

* Corresponding author.
E-mail addresses: pbc@icmc.usp.br (P.B. Castellucci), fran@icmc.usp.br (F.M.B. Toledo), alysson.costa@unimelb.edu.au (A.M. Costa).

https://doi.org/10.1016/j.orp.2019.100126
Received 26 March 2019; Received in revised form 25 September 2019; Accepted 25 September 2019
Available online 12 October 2019
2214-7160/ © 2019 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/BY-NC-ND/4.0/).

packing problems. Bartlett et al. [5] aims at maximizing the valued loaded at each time period in a planning horizon De Cauwer et al. [21] aims at minimizing the time each item remains in one of the bins. An exact solution method for the Temporal Bin Packing is proposed by Dell'Amico et al. [22]. In a sense, we propose an extension of these temporal packing problems since we consider the three-dimensional geometry of the problem and a schedule for the arrival of boxes.

We explore two specific challenges of the loading process at a cross-dock door. How to select a subset of boxes to (i) maximize the loaded volume (output maximization) and (ii) minimize the ready time of the truck/container. Furthermore, we mind the schedule of arrival of boxes. Since the availability of the boxes in time might be uncertain, we propose a stochastic dynamic programming framework which can accommodate this uncertainty and take advantage of the existing methods to solve the geometric aspects of the problem. Finally, in the loading process, we focus on the unsupported case with orthogonal packing and the possibility of rotating boxes over three axes, although the framework can be applied with any other practical geometric constraints by selecting an appropriate algorithm (from the container loading literature) for the sub-problems.

2. Container loading problem with time availability constraints (CLPTAC)

We propose an output maximization version of a container loading problem with time availability constraints (CLPTAC-Om). The objective is to position a subset of boxes inside a container maximizing the loaded volume and minimizing the ready time of the container to be dispatched. The boxes must not overlap with each other, must be fully inside the container and cannot be loaded before they are available. Also, we assume boxes and containers' faces must be parallel to each other (orthogonal packing). We define deterministic and stochastic versions of the problem with respect to the arrival time of the boxes. Moreover, we assume that boxes leaving the dock go to the same destination, which is a common situation in cross-docks with destination exclusive dock holding patterns (Ladier and Alpan [30]).

More formally, assume a finite time horizon $[0, T]$. Let $\mathcal{B}$ be the set of boxes, each box $i \in \mathcal{B}$ is defined by its length along each axis (l_{i1}, l_{i2}, l_{i3}) and the time it becomes available for loading t_i. Simulating the arrival process of the boxes, we assume that, once a box becomes available, it remains available until it is shipped to its destination. We consider a single container with dimensions (L_1, L_2, L_3). Both objectives, namely, maximizing the loaded volume and minimizing the ready time of the container dispatched are combined into a single function by weighting their relative importance. For the deterministic version, the arrival time of boxes $t_i \in [0, T]$, $i \in \mathcal{B}$, is defined *a priori*. For the stochastic version, only the probability distribution of t_i is known. The remainder of this paper focus on the stochastic version, although the dynamic programming framework we propose (Section 3) can also be applied to the deterministic version.

3. A dynamic programming framework for CLPTAC-Om

We propose a framework to solve the stochastic CLPTAC-Om which decomposes the temporal and geometric aspects of the problem. The temporal characteristics are handled by the evolution of the dynamic programming algorithm while the geometric constraints are handled as part of the sub-problems solved for each possible state. For this framework, we uniformly discretize the time horizon $[0, T]$ into a set of time periods $\mathcal{T}$ and define $\mathcal{B}_t$, $t \in \mathcal{T}$, to be the set of boxes available at time period $t \in \mathcal{T}$. An effective discretization might not be trivial for general systems, but for cross-docks a simple uniform discretization is justifiable due to usual high and stable demands.

The dynamic programming approach is formulated as an optimum stopping problem [36]. We want to decide when to stop waiting for new shipments and load the boxes into the truck. The trade-off is: the longer we wait, the more boxes we have available for packing yielding a potentially better volume occupation; on the other hand, the sooner we stop waiting (and perform the loading) the smaller is the ready time for the selected boxes.

Let the state of the system be the set of boxes $\mathcal{X}_t$ available at period $t \in \mathcal{T}$ and $C(\mathcal{X}_t, s_t)$ be the cost of decision $s_t \in \{0, 1\}$, $t \in \mathcal{T}$, in state $\mathcal{X}_t$. The possible decisions are to perform the loading or to wait for more boxes, henceforth coded as 0 (load) and 1 (wait). Also, let $\mathcal{E}(\mathcal{X}_t)$ be a solution of a traditional output maximization container loading problem with boxes in $\mathcal{X}_t$, namely, a solution that maximizes the volume loaded with boxes represented by $\mathcal{X}_t$ without any temporal constraints and $\overline{\mathcal{E}}(\mathcal{X}_t)$ be the volume of boxes that were not loaded. Finally, let μ be a parameter indicating the cost of empty volume in the container. Thereby, with $\mathcal{T}^*$ being the set of all time periods but the last and $\overline{\mathcal{B}}_t$ being the set of all shipments that might be available at time period $t \in \mathcal{T}$, the cost functions of the dynamic programming algorithm are given by (1)–(3).

$$C(\mathcal{X}_t, 0) = \frac{1}{|\mathcal{T}|} + \mu\left(1 - \frac{\mathcal{E}(\mathcal{X}_t)}{L_1 L_2 L_3}\right), \quad t \in \mathcal{T}^*, \mathcal{X}_t \subseteq \overline{\mathcal{B}}_t, \tag{1}$$

$$C(\mathcal{X}_t, 1) = \frac{1}{|\mathcal{T}|}, t \in \mathcal{T}^*, \mathcal{X}_t \subseteq \overline{\mathcal{B}}_t, \tag{2}$$

$$C(\mathcal{X}_{|\mathcal{T}|}) = \frac{1}{|\mathcal{T}|} + L_1 L_2 L_3 \lceil \frac{\overline{\mathcal{E}}(\mathcal{X}_{|\mathcal{T}|})}{L_1 L_2 L_3} \rceil, \mathcal{X}_{|\mathcal{T}|} \subseteq \overline{\mathcal{B}}_{|\mathcal{T}|}. \tag{3}$$

Eq. (1) define the cost of the decision to stop the waiting process and proceed with the loading with available boxes $\mathcal{X}_t$ at time $t \in \mathcal{T}^*$. If the decision is to wait for more boxes, the respective partial cost is given by (2). The terminal cost, i.e., the cost of reaching the last period in the planning horizon is defined by a bound on the volume of the trucks needed to transport all remaining boxes (3) weighted by the size of the truck. In a sense, it is the inventory holding costs, which should be high to preserve the cross-docking philosophy of reduced inventory. We want to optimize the objective (4), in which $\mathbb{E}$ denotes the expected value due to the uncertainty in the arrival times of the boxes.

$$\text{Min } \mathbb{E}\left(\sum_{t \in \mathcal{T}^*} \left[C(\mathcal{X}_t, s_t) s_{t-1} \right] + C(\mathcal{X}_{|\mathcal{T}|}) s_{|\mathcal{T}|-1} \right). \tag{4}$$

In which we used $s_0 = 1$. Expression (4) is equivalent to minimizing the sum of the ready times (in time periods) and the cost of the empty volume that it is possible to achieve with the boxes available until we decide to stop waiting, $s_t = 0$. Note that, if the decision in a particular period is to wait, $s_t = 1$, we sum the cost (regarding ready time) of that period, $\frac{1}{|\mathcal{T}|}$. Otherwise, $s_t = 0$, we still sum the cost related to the ready time (plus the cost of empty volume). In other words, we assume that the loading process only ends at the end of a period. We use $|\mathcal{T}|$ and $L_1 L_2 L_3$ as normalizing factor for ready time and volume, respectively.

We can find an optimal solution for the stochastic case by using the value functions (5) and (6).

$$\mathcal{V}_t(\mathcal{X}_t) = \text{Min}\{c_t(\mathcal{X}_t, 0), \ c_t(\mathcal{X}_t, 1)\}, \quad t \in \mathcal{T}^*, \mathcal{X}_t \subseteq \overline{\mathcal{B}}_t, \tag{5}$$

$$\mathcal{V}_{|\mathcal{T}|}(\mathcal{X}_t) = \frac{1}{|\mathcal{T}|} + L_1 L_2 L_3 \lceil \frac{\overline{\mathcal{E}}(\mathcal{X}_{|\mathcal{T}|})}{L_1 L_2 L_3} \rceil, \quad \mathcal{X}_t \subseteq \overline{\mathcal{B}}_{|\mathcal{T}|}. \tag{6}$$

Eq. (5) denote the best option between loading boxes currently available, $c_t(\mathcal{X}_t, 0)$, and waiting for more boxes, $c_t(\mathcal{X}_t, 1)$. Eq. (6) compute the cost in the final time period. The cost of each decision in (5) is computed as expressed in (8) and (7), respectively, for $t \in \mathcal{T}^*$.

$$c_t(\mathcal{X}_t, 0) = \frac{1}{|\mathcal{T}|} + \mu\left(1 - \frac{\mathcal{E}(\mathcal{X}_t)}{L_1 L_2 L_3}\right), t \in \mathcal{T}^*, \mathcal{X}_t \subseteq \overline{\mathcal{B}}_t, \tag{7}$$

$$c_t(\mathcal{X}_t, 1) = \frac{1}{|\mathcal{T}|} + \sum_{\bar{b} \subseteq \mathcal{X}_{t+1}} \rho_{t+1}(\bar{b} \mid \mathcal{X}_t) \, \mathcal{V}_{t+1}(\bar{b}), \quad t \in \mathcal{T}^*, \mathcal{X}_t \subseteq \overline{\mathcal{B}}_t. \tag{8}$$

In which $\rho_t(\mathcal{X}_t)$ is the probability of being in state $\mathcal{X}_t$ in time period t.

Eq. (7) compute the cost of performing the load, solving a container loading problem with boxes currently available $\mathcal{E}(\mathcal{X}_t)$. Eq. (8) compute the expected cost for the future time periods given the current state $\mathcal{X}_t$. Note the use of the future value $V_{t+1}(\cdot)$, characteristic of the dynamic programming framework.

Note that Eqs. (5) and (6) are independent of the probabilistic model for the arrival of the boxes. Just to illustrate our approach, we defined the following probabilistic model for the arrival times of boxes in $\mathcal{B}_t$. We assume the arrival time of subset $\mathcal{B}_{t_1}$ is independent of the arrival time of $\mathcal{B}_{t_2}$, $t_1 \neq t_2$, and the probability of a subset $\mathcal{B}_t$ arriving at time $i \in \mathcal{T}$ is given by:

$$\pi_i(\mathcal{B}_t) = \begin{cases} \alpha_{it}(1-\alpha_{it})^{i-t}, & \text{if } i \geq t, \\ 0 & \text{if } i < t, \end{cases} \tag{9}$$

in which $\alpha_{it} \in (0, 1)$, $i, t \in \mathcal{T}$, is a probability index associated with subset $\mathcal{B}_t$ arriving at time $i \in \mathcal{T}$, $i \geq t$. Within this framework, α_{it}, can be interpreted as a reliability profile for the company in charge of delivering the subset (cargo) $\mathcal{B}_t$. If $\alpha_{it} = 1$, $i, t \in \mathcal{T}$, by convention, we assume that we have the deterministic case, in which we known the arrival time of each box *a priori*. In other words, the boxes arrive on time. For simplicity, we used $\alpha_{it} = \alpha$, $i \in \mathcal{B}_t$, $t \in \mathcal{T}$, which can be viewed as a reliability index for the corresponding logistic environment. Thereby, given π, we can compute the probability of a set of shipments $\overline{\mathcal{B}}_t$ begin available at time period $t \in \mathcal{T}$.

The framework does not specify how to solve the container loading problem associated with $\mathcal{E}(\mathcal{X}_t)$. In fact, any solution method from the literature could be used, which means practical constraints are naturally incorporated into the framework. Here, we choose one of the best performing models [41], proposed by Chen et al. [16], which allows for rotation of boxes. Even though that are method capable of achieving higher average occupation, here we are interested in the understanding the effect of the temporal dimension on the problem.

3.1. The dynamic programming algorithm

The solution of Eqs. (5) and (6) can be obtained with the dynamic programming approach outlined in Algorithm 1. The reader interested in the theoretical background in dynamic programming can be referred to the book by Kumar and Varaiya [29], for example. The goal of the algorithm is to compute (or fill) the costs matrix M_{ji}, $j \in \mathcal{T}$, $i \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. The cost matrix provides the best cost from the current period j and current state i to the end of the planning horizon. Similarly, the best cost is matched with the best decision that is stored in matrix D_{ji}, $j \in \mathcal{T}$, $i \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. Thereby, after running the algorithm, we have the optimal decision for each combination of i and j, $j \in \mathcal{T}$, $i \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. The algorithm starts by setting initial values in matrices M_{ji} and D_{ji}. Then, for each combination of time period (t) and possible state ($\mathcal{X}_t \in \overline{\mathcal{B}}_t$), the algorithm compares the expected cost of waiting with the loading cost (according to (5)) and updates matrices M_{ji} and D_{ji}. Note that, on lines 2 and 6, we need to solve a container loading problem. We could precompute all instances of the container loading problem and use the results when needed. We solved these instances using the model proposed by Chen et al. [16], allowing rotation of the boxes. Computational experiments on solving these sub-problems are reported in Subsection 4.2. Then, in Subsection 4.3, we present results when using the matrices M_{ji} and D_{ji} inside a simulation of the arrival of boxes.

Fig. 1 exemplifies a scenario with $\mathcal{T} = \{1, 2, 3\}$. Each cell is a bit array ($\mathcal{B}_1\mathcal{B}_2\mathcal{B}_3$) representing the current state, i.e., set of boxes that have actually arrived at a point in time. The bit array $\overline{b}_1 = 100$, for example, means that shipment 1 has arrived while shipments 2 and 3 have not. In each time period, for each bit array with non-zero probability of occurrence, we need to evaluate the decisions of waiting and loading (Eq. (5)). The full blue lines represent the computation of the sum in Eq. (5) for all $\rho_i(\mathcal{X}_3 = \mathcal{B}_3) \neq 0$.

4. Computational experiments

We design computational experiments with two goals in mind: (i) gain some insight in the difficulty of solving this new class of problems, which can also indicate the effect of having better packing algorithms; and (ii) understanding some effects of the uncertainty in the arrival time of the shipments (set of boxes). For this, we generated two sets of instances, the first based on instances proposed by Ivancic et al. [25] and the second based on [9,20] (Subsection 4.1). Then, we report results in solving each state of the dynamic programming (Subsection 4.2) and solving the case with uncertainty (Subsection 4.3). All the experiments were run in an Intel® Core-i7-2600 CPU at 3.40GHz. All instances of the model were solved with Gurobi 8.0.[1]

4.1. Instance generation

The benchmark problems for the CLPTAC-Om were generated based on instances for the traditional Container Loading Problem. We generated two sets. One, which we call IMM-TAC, is based on 47 instances proposed by Ivancic et al. [25] and the other (BR-TAC) is based 50 instances from the benchmark proposed [9,20]. Originally, both sets provided the three-dimensional geometry for the boxes and the containers. For each box, we randomly generated a time from which it becomes available following a discrete uniform probability distribution in $[1, |\mathcal{T}|]$, $|\mathcal{T}| = 10$. Therefore, we are assuming a planning horizon with ten time slots (equivalent to 40 minutes slots in an eight-hour working day).

In Ivancic et al.'s benchmark, instances have from two to five types of boxes, the number of boxes varies from 47 to 181 and we consider only one container per instance – containers have different dimensions across instances. In Bischoff and Ratcliff and Davies and Bischoff's benchmark, the number of types of boxes varies from three to 50 and they are grouped by number of types of boxes (3, 5, 8, 10, 12, 15, 20, 30, 40 and 50). The dimensions of the container is constant across all instances. We select five instances from each group, thus, we generated 50 instances based on [9,20]. All instances are provided as supplementary material.

4.2. Solving the sub-problems of the dynamic programming

Solving the stochastic model for an instance of the CLPTAC-Om may be computationally expensive. Each evaluation of function $\mathcal{V}_t(\cdot)$ needs to solve a deterministic version of the container loading problem (via $\mathcal{E}(\cdot)$), which for real-world cases can be impractical itself, using an exact method. Therefore, to validate our framework, we solved each container loading sub-problem, associated with $\mathcal{E}$ (and $\overline{\mathcal{E}}$), of the dynamic programming with the time limit to five seconds to find a solution for each of the sub-problems related to V_t, $t \in \mathcal{T}$. Therefore, there are $2^{|\mathcal{T}|-1} = 2^{10} - 1 = 1023$ sub-problems for each instance. Results for each test set are presented in Tables 1 and 2, respectively.

Table 1 shows that, on average across all IMM-TAC instances, 77% of the container could be filled. The lowest rate of occupation was achieved in instance 18, however, this is close to the optimal solution for the instance since all but one of the respective sub-problems were solved to optimality. Also, the number of optimal solutions proven is around 28%, on average. In the column *Trivial solution*, we report the number of sub-problems for which the solver found only the trivial solution of not loading any boxes. This happened very rarely for this set of instances.

The average occupation for the BR-TAC instances was lower than the ones from IMM-TAC (see Table 2). This could be associated with the number of boxes in the instances. BR-TAC instances can fit a higher number of boxes inside the container, on average, this leads to harder to

[1] The source codes are available as supplementary material.

Data: Cost of empty volume (μ), set of time periods ($\mathcal{T}$), set of possible shipments available in each time period ($\overline{\mathcal{B}}_1, \ldots, \overline{\mathcal{B}}_{|\mathcal{T}|}$), geometry of the container (L_1, L_2, L_3), probability function ρ.

Result: Continuation cost matrix M_{ji} and decision matrix D_{ji}, $j \in \mathcal{T}$, $i \in \overline{\mathcal{B}}_{|\mathcal{T}|}$.

$M_{ji} = \infty, \quad i \in \mathcal{T}^*, j \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. /* Initial cost is infinity */

$M_{ji} = \frac{1}{|\mathcal{T}|} + L_1 L_2 L_3 \left\lceil \frac{\overline{\mathcal{E}}(\mathcal{X}_{|\mathcal{T}|})}{L_1 L_2 L_3} \right\rceil, i = |\mathcal{T}|, j \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. /* except for the last period. */

$D_{ji} = 0, \; i = |\mathcal{T}|, j \in \overline{\mathcal{B}}_{|\mathcal{T}|}$. /* Decision for the last period is to load. */

foreach $t \in \{|\mathcal{T}| - 1, |\mathcal{T}| - 2, \ldots, 1\}$ **do** /* From last to first period */

  foreach $j \in \overline{\mathcal{B}}_t$ **do** /* For every possible state */

    $ec = \frac{1}{|\mathcal{T}|} + \sum_{\overline{b} \subseteq \overline{\mathcal{B}}_{t+1}} \rho_{t+1}(\overline{b} \mid \overline{\mathcal{B}}_t) \, M_{\overline{b},i+1}$ /* Compute expected cost of waiting. */

    $st = \frac{1}{|\mathcal{T}|} + \mu\left(1 - \frac{\mathcal{E}(\mathcal{X}_t)}{L_1 L_2 L_3}\right)$ /* Compute loading cost. */

    if $ec < sc$ **then** /* If waiting is cheaper. */

      $M_{ji} = ec$

      $D_{ji} = 1$

    else /* If loading is cheaper. */

      $M_{ji} = st$

      $D_{ji} = 0$

Algorithm 1. DynSolve. Outline of the dynamic programming algorithm.

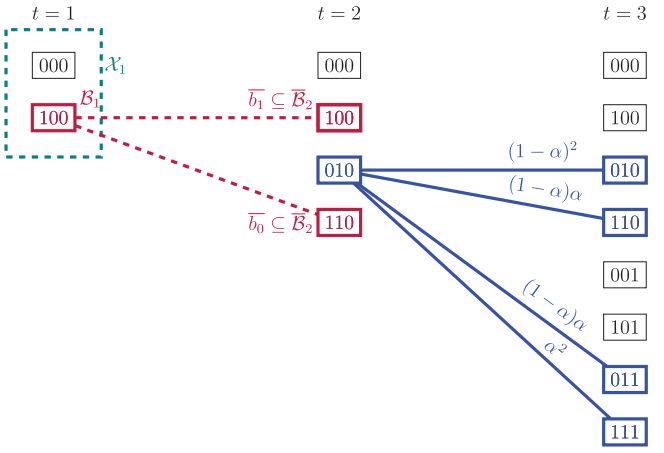


Fig. 1. Example of a scenario with $\mathcal{T} = \{1, 2, 3\}$. It is a graphical example of Eq. (5). The dashed red lines show possible next states for $\mathcal{X}_1$. The full blue lines show the states which need to be evaluated at state 010 at time $t = 2$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

might not arrive on time as defined in Section 3. To evaluate this stochastic version we used Algorithm 1. The algorithm produces a set of decisions for all combinations of shipments $\{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_{|\mathcal{T}|}\}$ that might be available at time period $t \in \mathcal{T}$. We experimented with different trade-off parameters for the cost of empty spaces $\mu \in \{1, 2, 4\}$. In the following, we denote CLPTACμ instances in which the cost of empty spaces is $\mu \in \{1, 2, 4\}$.

The goal of these experiments is to evaluate the effect of the cost of empty volume (μ) and of the reliability index (α) on total ready time and volume occupation. For this, with the optimal policy from Eq. (5), we run the simulation of the arrival of boxes 5000 times for each $\alpha \in \{0.5, 0.6, ..., 0.9\}$. At each time period, we use the results (matrix D_{ji}) to decide if we wait for more boxes or perform the loading. Note that the optimal decisions D_{ji} is only computed once, at the beginning, and then we use this information in the simulation. Also, regarding the computational time, each run of the simulation takes less than a second.

The results for the IMM-TAC instances revealed that operation time and volume occupation tend to improve as reliability index increases. An improving trend can be observed in both ready time and volume occupation (Fig. 2). Regarding operation time, this trend can be explained due to a higher risk of waiting in an environment with lower

Table 1
Summary of the statistics of solving each of the sub-problems for each of the instances based on [25] (IMM-TAC). For each instance, we report the fraction of the container (in percentage) that was filled (on average), the fraction of proven optimal solutions found within the time limit and the number of trivial solution which the solver could not improve.

Id	Occupation	Optimals (%)	Trivial solutions	Id	Occupation	Optimals (%)	Trivial solutions
1	94.92	80.74	0	25	73.23	4.69	0
2	74.64	1.47	0	26	68.66	18.96	0
3	77.22	2.44	0	27	64.90	3.91	0
4	99.65	98.53	0	28	76.08	0.59	0
5	95.64	30.69	0	29	74.78	0.29	0
6	91.07	28.35	0	30	78.54	0.10	0
7	99.79	95.89	0	31	72.62	0.29	0
8	78.50	6.74	0	32	72.35	9.78	0
9	84.90	0.88	0	33	65.43	37.54	6
10	68.38	89.74	0	34	71.64	6.65	0
11	75.34	0.29	0	35	64.48	28.15	0
12	87.89	98.04	0	36	82.83	1.08	0
13	96.15	32.55	0	37	83.75	5.28	0
14	100.00	100.00	0	38	85.91	1.47	0
15	90.83	26.88	0	39	79.95	0.59	0
16	98.52	85.34	0	40	81.16	1.08	0
17	80.06	1.66	0	41	91.43	1.76	0
18	36.81	99.90	0	42	73.47	7.62	0
19	47.54	95.41	0	43	62.05	23.95	1
20	68.03	46.43	0	44	63.10	13.20	0
21	83.11	1.17	0	45	63.78	25.61	0
22	75.71	1.08	0	46	58.91	52.59	0
23	75.03	45.55	1	47	65.82	11.63	0
24	80.95	2.64	0				
				Avg.	77.35	28.28	0.17

solve instances of the models [41].

The number of proven optimal solutions is higher than 70% on average. The pattern we could observe in the experiments is that these proofs of optimality are more frequent in smaller (less boxes) subproblems, which is expected. Moreover, the number of trivial solutions reported is higher than the results for IMM-TAC showed. In a sense, IMM-TAC instances could be considered *easier* to solve when compared to the ones derived from [9,20] (BR-TAC).

4.3. Results for the case with uncertainty

Having solved each of the sub-problems that could happen when the shipments (set of boxes) do not follow their defined schedule, we can use the dynamic programming approach to solve a stochastic version of the problem. In this case, the uncertainty is related to set of boxes which

reliability index. The same reasoning applies to the volume occupation, the higher the risk of waiting the lower the occupation achieved. For the decision maker, it suggests that the more reliable the logistic environment the better the expected vehicles occupation and the operation time, which is expected. Moreover, a linear trend specially for the operation time in relation to the reliability index was revealed. It can help managers in devising business plans to account for costs of operation, delivery times and value of possible penalties for delayed deliveries in seasonal variations of the reliability index. This, besides allowing a more efficient management of the facility, can potentially increase the service level for final customers. For the BR-TAC instances, a similar behavior can be observed for $\mu = 2$ and $\mu = 4$ (see Fig. 3). However, the operation time for $\mu = 1$ increased as the reliability index increased. This can happen in cases for which the μ parameter favors the operation time objective too much, in other words, the loading

Table 2
Summary of the statistics of solving each of the 1023 sub-problems for each of the instances based on [9,20] (BR-TAC). For each instance, we report the fraction of the container that was filled (in percentage), the number of proven optimal solutions found within the time limit and the number of trivial solution which the solver could not improve. The first number in the *id* column represents from which set of the original benchmarks the instance was generated. Higher number correspond to more types of boxes.

Id	Occupation	Optimals (%)	Trivial solutions	Id	Occupation	Optimals (%)	Trivial solutions
1.1	47.74	79.96	0	6.1	46.95	75.46	1
1.2	46.30	71.07	3	6.2	45.57	68.23	22
1.3	47.10	71.85	1	6.3	46.39	72.63	6
1.4	32.06	41.54	225	6.4	46.02	70.77	11
1.5	47.88	79.18	4	6.5	45.28	69.99	15
2.1	48.12	85.83	0	7.1	46.90	81.33	0
2.2	47.50	78.30	0	7.2	46.86	77.91	1
2.3	41.63	55.52	64	7.3	47.12	77.13	1
2.4	33.79	49.17	210	7.4	45.51	67.94	20
2.5	48.08	81.72	0	7.5	47.28	76.05	1
3.1	47.92	82.21	0	8.1	46.20	70.97	11
3.2	47.70	79.67	0	8.2	47.00	76.25	4
3.3	46.30	69.99	11	8.3	46.45	73.80	3
3.4	37.37	53.37	147	8.4	45.75	70.28	12
3.5	47.88	79.86	0	8.5	44.96	67.55	26
4.1	46.60	80.94	0	9.1	45.67	69.01	15
4.2	47.00	77.42	1	9.2	47.17	77.22	0
4.3	46.91	72.92	3	9.3	46.92	75.37	1
4.4	42.94	64.13	59	9.4	46.43	71.46	3
4.5	46.79	73.51	2	9.5	46.01	70.28	12
5.1	47.35	82.01	0	10.1	45.88	74.19	14
5.2	46.03	71.55	7	10.2	47.02	77.22	1
5.3	47.07	74.19	2	10.3	46.30	73.51	9
5.4	46.61	72.14	7	10.4	47.17	76.93	0
5.5	46.97	75.56	1	10.5	46.52	71.55	0
				Avg.	46.70	75.66	18.80

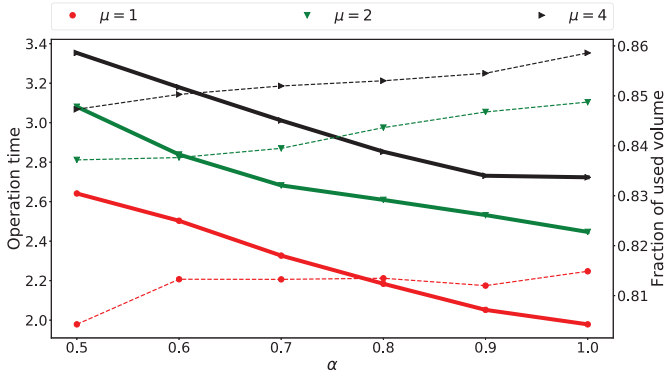


Fig. 2. Profile of loaded volume (dashed line) and operation time (full line) related to the reliability index (α) for $\mu \in \{1, 2, 4\}$ for the IMM-TAC instances. The values were averaged across all instances.

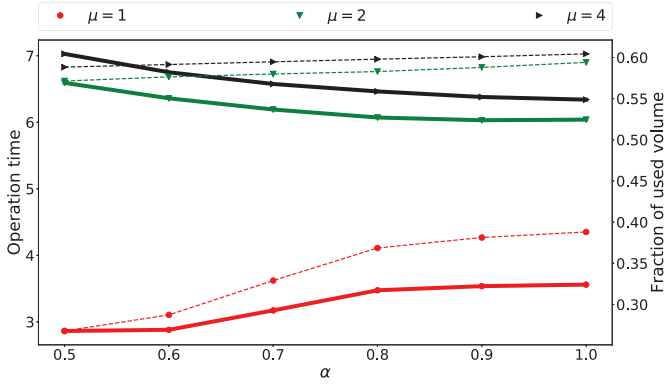


Fig. 3. Profile of loaded volume (dashed line) and operation time (full line) related to the reliability index (α) for $\mu \in \{1, 2, 4\}$ for the BR-TAC instances. The values were averaged across all instances.

process finishes too early.

5. Conclusions and future research

High occupation rates of trucks/container are essential for effective road distribution systems. However, achieving these rates can be challenging due to the pressure for lower and lower delivery times. We define an *Output Maximization Container Loading Problem with Time Availability Constraints* which accommodates these two opposing goals: high filling rates and low delivery times. Moreover, we propose a dynamic programming framework which is suited to handle the problem, including uncertainty on the availability of boxes in time. This framework separates the geometric and temporal aspects of the decisions, enabling the use of all available solution methodologies for tackling different practical constraints in the container loading process, including some which we did not consider in this research (e.g. vertical stability and weight distribution). Future work may extend the core decision problem considered here in order to include additional decisions in the context of cross-docking strategies. In particular, some natural extensions would be the integration of scheduling decisions for the arrival of shipments and the consideration of multiple docks. Other avenue for research is the consideration of multiple destinations, with the definition of shipping routes in the second layer of the distribution network.

Acknowledgments

We thank State of São Paulo Research Foundation (FAPESP), process numbers, #2018/07240-0 #2017/01097-9, #2015/15024-8, CEPID #2013/07375-0 and National Council for Scientific and Technological Development (CNPq), grant #308761/2018-9 from Brazil.

We also thank Professor Eduardo Fontoura Costa from the Instituto de Ciências Matemática Matemáticas e de Computação (ICMC) of the University of São Paulo for the initial discussions regarding the dynamic programming algorithm.

Finally, we thank all the reviewers for their valuable feedback during the review process.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.orp.2019.100126

References

[1] Alonso M, Alvarez-Valdes R, Iori M, Parreño F. Mathematical models for multi container loading problems with practical constraints. Comput Ind Eng 2019;127:722–33.
[2] Amini A, Tavakkoli-Moghaddam R. A bi-objective truck scheduling problem in a cross-docking center with probability of breakdown for trucks. Comput Ind Eng 2016;96:180–91.
[3] Araya I, Guerrero K, Nuñez E. VCS: A new heuristic function for selecting boxes in the single container loading problem. Comput Oper Res 2017;82:27–35.
[4] Assadi MT, Bagheri M. Differential evolution and population-based simulated annealing for truck scheduling problem in multiple door cross-docking systems. Comput Ind Eng 2016;96:149–61.
[5] Bartlett M, Frisch AM, Hamadi Y, Miguel I, Tarim SA, Unsworth C. The temporal knapsack problem and its solution. International conference on integration of artificial intelligence (AI) and operations research (OR) techniques in constraint programming. 2005. p. 34–48.
[6] Bazgosha A, Ranjbar M, Jamili N. Scheduling of loading and unloading operations in a multi stations transshipment terminal with release date and inventory constraints. Comput Ind Eng 2017;106:20–31.
[7] Bellanger A, Hanafi S, Wilbaut C. Three-stage hybrid-flowshop model for cross-docking. Comput Oper Res 2013;40:1109–21.
[8] Belle JV, Valckenaers P, Cattrysse D. Cross-docking: state of the art. Omega 2012;40:827–46.
[9] Bischoff EE, Ratcliff MSW. Issues in the development of approaches to container loading. Omega 1995;23:377–90.
[10] Boloori Arabani A, Ghomi S, Zandieh M. A multi-criteria cross-docking scheduling with just-in-time approach. Int J Adv Manuf Technol 2010;49:741–56.
[11] Boloori Arabani A, Ghomi S, Zandieh M. Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage. Expert Syst Appl 2011;38:1964–79.
[12] Boloori Arabani A, Zandieh M, Ghomi S. Multi-objective genetic-based algorithms for a cross-docking scheduling problem. Appl Soft Comput J 2011;11:4954–70.
[13] Bortfeldt A. A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. Comput Oper Res 2012;39:2248–57.
[14] Bortfeldt A, Wäscher G. Constraints in container loading – A state-of-the-art review. Eur J Oper Res 2013;229:1–20.
[15] Ceschia S, Schaerf A. Local search for a multi-drop multi-container loading problem. J Heuristics 2013;19:275–94.
[16] Chen CS, Lee SM, Shen QS. An analytical model for the container loading problem. Eur J Oper Res 1995;80:68–76.
[17] Chen F, Song K. Minimizing makespan in two-stage hybrid cross docking scheduling problem. Comput Oper Res 2009;36:2066–73.
[18] Chmielewski A, Naujoks B, Janas M, Clausen U. Optimizing the door assignment in LTL-Terminals. Transp Sci 2009;43:198–210.
[19] Cota PM, Gimenez BMR, Araújo DPM, Nogueira TH, de Souza MC, Ravetti MG. Time-indexed formulation and polynomial time heuristic for a multi-dock truck scheduling problem in a cross-docking centre. Comput Ind Eng 2016;95:135–43.
[20] Davies AP, Bischoff EE. Weight distribution considerations in container loading. Eur J Oper Res 1999;114:509–27.
[21] De Cauwer M, Mehta D, O'Sullivan B. The temporal bin packing problem: An application to workload management in data centres. Proceedings - 2016 IEEE 28th international conference on tools with artificial intelligence, ICTAI 2016. 2017. p. 157–64.
[22] Dell'Amico M., Furini F., Iori M.. A branch-and-price algorithm for the temporal bin packing problem. arXiv:190204925 2019;.
[23] Gelareh S, Monemi RN, Semet F, Goncalves G. A branch-and-cut algorithm for the truck dock assignment problem with operational time constraints. Eur J Oper Res 2016;249:1144–52.
[24] Gonçalves JF, Resende MGC. A parallel multi-population biased random-key genetic algorithm for a container loading problem. Comput Oper Res 2012;39:179–90.
[25] Ivancic N, Mathur K, Mohanty BB. An integer programming based heuristic approach to the three-dimensional packing problem. J Manuf Oper Manag 1989;2:268–89.
[26] Jamrus T, Chien C-F. Extended priority-based hybrid genetic algorithm for the less-than-container loading problem. Comput Ind Eng 2016;96:227–36.
[27] Junqueira L, Morabito R, Yamashita SD. Three-dimensional container loading models with cargo stability and load bearing constraints. Comput Oper Res 2012;39:74–85.
[28] Keshtzari M, Naderi B, Mehdizadeh E. An improved mathematical model and a hybrid metaheuristic for truck scheduling in cross-dock problems. Comput Ind Eng 2016;91:197–204.
[29] Kumar PR, Varaiya P. Stochastic systems: estimation, identification, and adaptive control. Society for Industrial and Applied Mathematics; 2015.
[30] Ladier AL, Alpan G. Cross-docking operations: current research versus industry practice. Omega 2016;62:145–62.
[31] Ladier A-L, Alpan G. Crossdock truck scheduling with time windows: earliness, tardiness and storage policies. J Intell Manuf 2018;29:569–83.
[32] Liao TW, Egbelu PJ, Chang PC. Two hybrid differential evolution algorithms for optimal inbound and outbound truck sequencing in cross docking operations. Appl Soft Comput J 2012;12:3683–97.
[33] Lim A, Ma H, Qiu C, Zhu W. The single container loading problem with axle weight constraints. Int J Prod Econ 2013;144:358–69.
[34] Maknoon M, Soumis F, Baptiste P. An integer programming approach to scheduling the transshipment of products at cross-docks in less-than-truckload industries. Comput Oper Res 2017;82:167–79.
[35] Martello S, Pisinger D, Vigo D. The three-dimensional bin packing problem. Oper Res 2000;48:256–67.
[36] Peskir G, Shiryaev A. Optimal stopping and free-boundary problems. Springer; 2006.
[37] Rahmanzadeh Tootkaleh S, Fatemi Ghomi SMT, Sheikh Sajadieh M. Cross dock scheduling with fixed outbound trucks departure times under substitution condition. Comput Ind Eng 2016;92:50–6.
[38] Ramos AG, Silva E, Oliveira JF. A new load balance methodology for container loading problem in road transportation. Eur J Oper Res 2018;266:1140–52.
[39] Shakeri M, Low MYH, Turner SJ, Lee EW. A robust two-phase heuristic algorithm for the truck scheduling problem in a resource-constrained crossdock. Comput Oper Res 2012;39:2564–77.
[40] Sheng L, Hongxia Z, Xisong D, Changjian C. A heuristic algorithm for container loading of pallets with infill boxes. Eur J Oper Res 2016;252:728–36.
[41] Silva EF, Toffolo TAM, Wauters T. Exact methods for three-dimensional cutting and packing: a comparative study concerning single container problems. Comput Oper Res 2019;109:12–27.
[42] Takahara S, Miyamoto S. An Evolutionary Approach for the Multiple Container Loading Problem. Hybrid intelligent systems, fifth international conference on. IEEE; 2005. p. 1–6.
[43] Tian T, Zhu W, Lim A, Wei L. The multiple container loading problem with preference. Eur J Oper Res 2016;248:1–11.
[44] Vahdani B, Zandieh M. Scheduling trucks in cross-docking systems: a robust meta-heuristics approach. Comput Ind Eng 2010;58:12–24.
[45] Van Belle J, Valckenaers P, Vanden Berghe G, Cattrysse D. A tabu search approach to the truck scheduling problem with multiple docks and time windows. Comput Ind Eng 2013;66:818–26.
[46] Wei L, Zhu W, Lim A. A goal-driven prototype column generation strategy for the multiple container loading cost minimization problem. Eur J Oper Res 2015;241:39–49.
[47] Wisittipanich W, Hengmeechai P. Truck scheduling in multi-door cross docking terminal by modified particle swarm optimization. Comput Ind Eng 2017;113:793–802.
[48] Yu W, Egbelu PJ. Scheduling of inbound and outbound trucks in cross docking systems with temporary storage. Eur J Oper Res 2008;184:377–96.
[49] Zhao X, Bennell JA, Bektaş T, Dowsland K. A comparative review of 3D container loading algorithms. Int Trans Oper Res 2016;23:287–320.
[50] Zheng J-N, Chien C-F, Gen M. Multi-objective multi-population biased random-key genetic algorithm for the 3-D container loading problem. Comput Ind Eng 2015;89:80–7.