




Article

The Double Traveling Salesman Problem with Multiple Stacks and a Choice of Container Types

Lars Magnus Hvattum ^{1,*}, Gregorio Tirado ^{2,3,*} and Ángel Felipe ⁴

¹ Faculty of Logistics, Molde University College, 6410 Molde, Norway

² Department of Financial and Actuarial Economics & Statistics, Complutense University of Madrid, 28223 Madrid, Spain

³ Interdisciplinary Mathematics Institute, Complutense University of Madrid, 28040 Madrid, Spain

⁴ Department of Statistics and Operational Research, Complutense University of Madrid, 28040 Madrid, Spain; felipe@mat.ucm.es

* Correspondence: hvattum@himolde.no (L.M.H.); gregoriotd@ucm.es (G.T.)

Received: 26 May 2020; Accepted: 11 June 2020; Published: 16 June 2020



Abstract: The double traveling salesman problem with multiple stacks involves the transportation of goods between two regions. In one region, a vehicle carrying a container visits customers, where pallets of goods are loaded into the container. The container is then shipped to a different region, where another vehicle visits another set of customers where the pallets are unloaded. Pallets are loaded in several rows inside the container, where each row follows the last-in-first-out principle. The standard test instances for the double traveling salesman problem with multiple stacks implies the use of a 45-foot pallet wide container to carry EUR-1 pallets. This paper investigates the effect on transportation costs if an open side container could be used when transporting the pallets. Computational experiments show savings in transportation costs of up to 20%. Moreover, by using a container loaded from the side, rather than from the rear, the defining attributes of the double traveling salesman problem seem to be lost.

Keywords: intermodal transportation; vehicle routing; loading; variable neighborhood search

1. Introduction

Petersen and Madsen [1] introduced an optimization problem referred to as the double traveling salesman problem with multiple stacks, based on a prospective customer of a company producing computer software systems for transportation companies. The problem is set in two different regions. A vehicle carrying a container must visit customers to make pickups in one region. The container is then transported to a different region, whereupon a different vehicle carries the container while visiting customers to make deliveries. It is not possible to repack the container en route, and an opening in one end of the container provides the only access to its contents.

The items transported are standardized pallets and each container can fit a given number of R rows with a limited number of L pallets each. This information is crucial, due to the inability to repack the container. Each row of pallets forms a stack that must be loaded and unloaded based on a first-in-last-out principle. This provides a set of difficult linking constraints that must be taken into consideration when routing the vehicle in both the pickup region and the delivery region. For a given total capacity of $R * L$, the loading constraints are most severe when R is low and L is high, whereas having a high value of R provides more flexibility.

Since its introduction, the double traveling salesman problem with multiple stacks has received significant attention from researchers, examining both exact and heuristic solution methods, as well as performing studies on computational complexity and polyhedral analysis. However, it seems that the

initial assumptions of the underlying problem structure have not been examined. In this work, one of the hidden assumptions of the problem is questioned: the type of container available to carry out the transportation. By replacing the type of container used in the transportation, the defining characteristics of the problem seem to be lost. This suggests that a technical solution of modifying the container technology used would have cancelled out the need for advancing the frontier of operations research.

Petersen and Madsen [1] provided test instances where the container could contain three rows with 11 pallets each. Standardized Euro Pallets (EPAL, or EUR-1) have dimensions of 1200 by 800 millimetres. This provides a good match with the internal dimensions of a 45-foot pallet wide container, which has an internal width of 2400 millimetres and an internal length of about 13,600 millimetres, with the exact dimensions varying between different manufacturers. Additional test instances were presented with three rows of 22 pallets each, based on situations where the height of each pallet is less than half of the height of the container, and where a pallet can be put on top of another pallet. However, the same loading constraints could also arise from the transportation of another standardized pallet, the EUR-6 pallet, which has dimensions of 600 by 800 millimetres.

The basic original instances have $R = 3$ rows of pallets, each row of length $L = 11$, providing an overall capacity of $R * L = 3 * 11 = 33$ pallets. However, there is an alternative configuration given the dimensions of the container. Placing the EUR-1 pallets with their long side towards the short side of the container, only two rows of pallets will fit. However, there is enough capacity for either 16 or 17 pallets in each row, depending on the exact length of the container. Therefore, the longest editions of the 45-foot containers have enough space for $2 * 17 = 34$ pallets in total. When pallets are vertically stackable, the pattern of three rows provides a capacity of $3 * 22 = 66$ pallets, whereas the pattern of two rows provides a maximum capacity of $2 * 34 = 68$. Transporting EUR-6 pallets provides another possibility. Instead of using $R = 3$ rows of length $L = 22$ and a total capacity of $3 * 22 = 66$, it is possible to use $R = 4$ rows of length $L = 17$ for a total capacity of $4 * 17 = 68$.

The above is true for containers loaded from the back, that is, using one of the short sides. Open side versions also exist for many container sizes, where goods can be loaded from one of the long sides of the container. The existence of such capabilities would open up a wide range of options in the context of the double traveling salesman problem with multiple stacks. For any of the previously mentioned loading options, the number of rows R and their length L can be swapped, producing instances with many rows of relatively low length. Table 1 provides examples of values for R and L that may appear, depending on the type of pallet and container used. Figure 1 illustrates two of the combinations.

Table 1. Selected values for the number of stacks R and their capacity L based on different pallets and container technologies. The first three rows correspond to values that are covered by existing instances.

R	L	Configuration of Pallets and Container
3	11	EUR-1 pallets, 45 ft container loaded from the back
3	22	EUR-6 pallets, 45 ft container loaded from the back
3	44	Vertically stacked EUR-6 pallets, 45 ft container loaded from the back
2	17	EUR-1 pallets, 45 ft container loaded from the back
2	34	Vertically stacked EUR-1 pallets, 45 ft container loaded from the back
4	17	EUR-6 pallets, 45 ft container loaded from the back
4	34	Vertically stacked EUR-6 pallets, 45 ft container loaded from the back
11	3	EUR-1 pallets, 45 ft container loaded from the side
11	6	Vertically stacked EUR-1 pallets, 45 ft container loaded from the side
17	2	EUR-1 pallets, 45 ft container loaded from the side
17	4	EUR-6 pallets, 45 ft container loaded from the side
17	8	Vertically stacked EUR-6 pallets, 45 ft container loaded from the side
22	3	EUR-6 pallets, 45 ft container loaded from the side
22	6	Vertically stacked EUR-6 pallets, 45 ft container loaded from the side

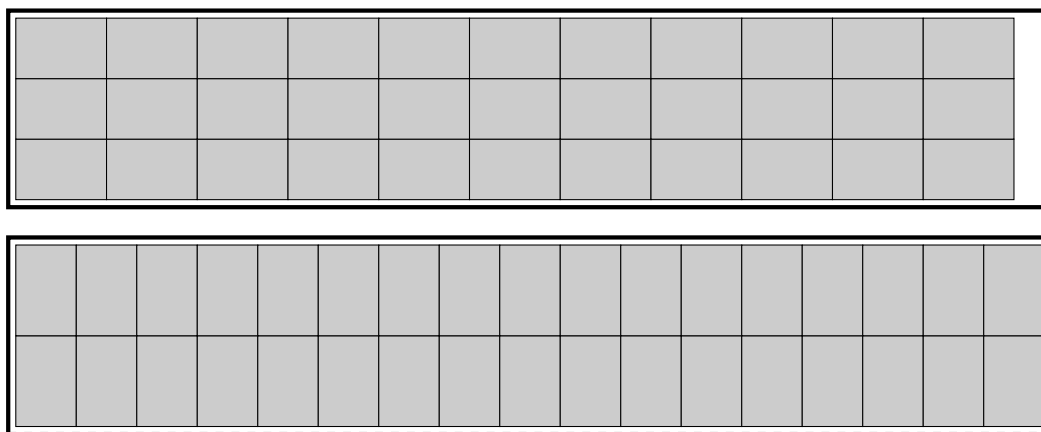


Figure 1. Top: a 45-foot container loaded from the back with three rows of 11 pallets each, resulting in a problem with $R = 3$ and $L = 11$. Bottom: a 45-foot container loaded from the side with two rows of 17 pallets each, resulting in a problem with $R = 17$ and $L = 2$.

The rest of this paper is structured, as follows. Section 2 reviews the related literature, while Section 3 presents the mathematical formulation of the double traveling salesman problem with multiple stacks and the solution methods used in our analysis. Section 4 contains a computational study, with the aim of determining the importance of considering different container types in the considered problem. The paper is concluded with Section 5, where some conclusions and additional thoughts about underlying assumptions of the problem are presented.

2. Literature

The double traveling salesman problem with multiple stacks was first introduced by Petersen and Madsen [1], who presented several heuristic algorithms for the problem, based on iterated local search, tabu search, simulated annealing, and large neighborhood search. Variable neighborhood search for the problem was examined by Felipe et al. [2], with improvements in [3], whereas Urrutia et al. [4] presented a dynamic programming based local search method.

Several mathematical formulations of the double traveling salesman problem with multiple stacks with corresponding solution strategies were presented by Petersen et al. [5]. A specialized algorithm was given by [6] and improved by Lusby and Larsen [6], based on combining separate traveling salesman problems for the pickup region and the delivery region. Branch-and-cut was used by Alba Martínez et al. [7], whereas Carrabs et al. [8] presented a branch-and-bound algorithm for the problem with only two stacks. Two stacks, with infinite capacity each, was also solved by Barbato et al. [9] using a set covering approach, and in [10] using a branch-and-cut algorithm.

Some of the theoretical properties of the problem were investigated by Casazza et al. [11], leading to a simple heuristic method. Given a route for the pickup region and a route for the delivery region, Toulouse and Calvo [12] showed that it can be decided in polynomial time whether or not a feasible stacking exists. Furthermore, Toulouse and Calvo [12] also showed that, given a stacking, optimal routes conditional on the stacking can be determined in polynomial time. Bonomo et al. [13] discussed similar complexity results.

After the introduction of the double traveling salesman problem with multiple stacks, many variants have also been considered in the literature: Iori and Riera-Ledesma [14] presented a generalization with multiple vehicles, and gave three mathematical formulations with corresponding exact solution methods. Heuristics based on iterated local search, simulated annealing, and variable neighborhood descent were proposed for this problem by da Silveira et al. [15]. Another simulated annealing implementation was provided by Chagas et al. [16] and a variable neighborhood search by Chagas et al. [17].

Another variation arises when considering a pickup and delivery traveling salesman problem with multiple stacks, where the nodes to visit are not necessarily split into two separate regions. A large

neighborhood search was proposed for this generalization by Côté et al. [18], whereas both Pereira and Urrutia [19] and Sampaio and Urrutia [20] presented branch-and-cut algorithms.

The pickup and delivery problem with time windows and multiple stacks was investigated by [21]. Other types of loading constraints have been considered in the literature as well. Doerner et al. [22] considered the transportation of wood products, while using both a tabu search and an ant colony optimization method. Gendreau et al. [23] used tabu search and Iori et al. [24] a branch-and-cut algorithm for vehicle routing problems with two-dimensional loading constraints. Fuellerer et al. [25] used ant colony optimization for a vehicle routing problem with three-dimensional loading constraints, and Chagas et al. [26] considered partial last-in-first-out loading constraints. Iori [27] presented a survey on combined routing and loading problems and is recommended for further details about related problems.

Even though a wide variety of problems with loading and capacity constraints leading to stacks of items have been approached in the literature through different methodologies, as far as the authors are aware, none of them performed an analysis of the practical consequences that the choice of container types and packing patterns used for transportation may have on the final costs. This is the research gap that this paper tries to fill, focusing on the case of the original double traveling salesman problem with multiple stacks.

3. Background

In this section, we first provide a mathematical model to formally define the studied problem. Next, we describe the solution methods used in the analysis.

3.1. Mathematical Model

The double traveling salesman problem with multiple stacks was modelled for the first time by Petersen and Madsen [1]. Let $G^1 = (V^1, A^1)$, $G^2 = (V^2, A^2)$ be two complete graphs representing, respectively, the pickup and delivery networks of the problem. For every $\omega \in \{1, 2\}$, edge $(i, j) \in A^\omega$ of G^ω has a certain weight c_{ij}^ω , representing the travel cost between nodes i and j . Let n be the number of orders and the node sets $V^\omega = \{v_0^\omega, v_1^\omega, \dots, v_n^\omega\}$, $\omega \in \{1, 2\}$, where v_0^ω is the depot and $v_1^\omega, \dots, v_n^\omega$ represent the n orders. The configuration of the container used for transportation is given by the number of available stacks, being denoted by R , and their maximum capacity, denoted by L .

In addition, we have set $V_*^\omega = V^\omega \setminus \{v_0^\omega\}$, containing all nodes, but the depot in each graph $\omega \in \{1, 2\}$, and set $P = \{1, \dots, R\}$, representing the R available stacks. The set of orders is $D = \{1, \dots, n\}$, in a way that the item associated to each order $i \in D$ must be picked up at $v_i^1 \in V_*^1$ of G^1 , loaded into a certain stack $p \in P$, and delivered at $v_i^2 \in V_*^2$ of G^2 .

The problem is modelled as a binary program through three sets of binary variables. The routes of the solution are given by variables $\{x_{ij}^\omega\}$, which determine directly the values of variables $\{y_{ij}^\omega\}$, that indicate precedence between pickups and deliveries. The assignment of orders to stacks is given by variables $\{z_{ip}\}$. They are defined in what follows, where $\omega \in \{1, 2\}$.

$$\begin{aligned}
 x_{ij}^\omega &= \begin{cases} 1 & \text{if } j \text{ is visited immediately after } i \text{ in network } \omega \\ 0 & \text{otherwise} \end{cases} & \forall i, j \in V^\omega \\
 y_{ij}^\omega &= \begin{cases} 1 & \text{if } j \text{ is visited after } i \text{ in network } \omega \\ 0 & \text{otherwise} \end{cases} & \forall i, j \in V_*^\omega \\
 z_{ip} &= \begin{cases} 1 & \text{if order } i \text{ is assigned to stack } p \\ 0 & \text{otherwise} \end{cases} & \forall i \in D, \forall p \in P
 \end{aligned}$$

The model is thus given by Equations (1)–(10).

$$\min f = \sum_{\substack{i, j \in V^\omega \\ \omega \in \{1, 2\}}} c_{ij}^\omega \cdot x_{ij}^\omega \tag{1}$$

$$\sum_{i \in V^\omega} x_{ij}^\omega = 1 \quad \forall j \in V^\omega, \forall \omega \tag{2}$$

$$\sum_{j \in V^\omega} x_{ij}^\omega = 1 \quad \forall i \in V^\omega, \forall \omega \tag{3}$$

$$y_{ij}^\omega + y_{ji}^\omega = 1 \quad \forall i, j \in V_*^\omega, i \neq j, \forall \omega \tag{4}$$

$$y_{ik}^\omega + y_{kj}^\omega \leq y_{ij}^\omega + 1 \quad \forall i, j, k \in V_*^\omega, \forall \omega \tag{5}$$

$$x_{ij}^\omega \leq y_{ij}^\omega \quad \forall i, j \in V_*^\omega, \forall \omega \tag{6}$$

$$y_{ij}^1 + z_{ip} + z_{jp} \leq 3 - y_{ij}^2 \quad \forall i, j \in V_*^\omega, \forall p \in P \tag{7}$$

$$\sum_{p \in P} z_{ip} = 1 \quad \forall i \in D \tag{8}$$

$$\sum_{i \in D} z_{ip} \leq L \quad \forall p \in P \tag{9}$$

$$x, y, z \in \{0, 1\} \tag{10}$$

The objective function (1), to be minimized, is the sum of all pickup and delivery costs. The flow conservation constraints are given in (2) and (3), while the right definition of $\{y_{ij}^\omega\}$ variables is ensured by constraints (4)–(6) and the LIFO order to be followed in each stack is imposed by Equation (7). Finally, constraints (8) indicate that each order must be assigned to one, and only one, stack and constraints (9) make sure that the maximum capacity of the stacks is not exceeded.

3.2. Solution Method

Metaheuristics are problem-independent algorithmic frameworks that describe strategies for developing powerful heuristic optimization methods [28]. Therefore, they can be applied to a wide range of problems, such as optimizing non-linear functions of continuous variables [29] or linear functions on binary variables [30]. For the double traveling salesman problem with multiple stacks, to analyze the choice of container types and packing patters, this paper uses a variable neighborhood search [3].

Variable neighborhood search was introduced by Mladenović and Hansen [31]. It is based on the performance of several consecutive local search procedures by changing the neighborhood structure used to define neighbors every time that the search gets stuck in a local optimum. This is a simple idea that has produced very good results in a wide variety of hard optimization problems. The algorithm we use to solve the double traveling salesman problem with multiple stacks is based on an enhanced variable neighborhood search with some additional elements that are specifically adapted to the problem at hand [3]. Six different neighborhood structures or operators, each of which define a local search procedure, are used: Route Swap (swaps the positions of two consecutive orders in one of the routes of the solution), Complete Swap (swaps the stack positions of two orders that are assigned to different stacks), In-Stack Swap (swaps the stack positions of two orders that are assigned to the same stack), Reinsertion (moves one order to a different position in both routes of the solution and reassigns it to a different stack), r -Route Permutation (r orders that are assigned to different stacks and visited consecutively in one route are permuted), and r -Stack Permutation (r orders that are loaded consecutively into the same stack are permuted). The core of the solution method is a variable neighborhood descent (VND) algorithm in which several local search procedures using these neighborhood structures are concatenated. The VND takes as input an initial feasible solution S and the set $\Delta = \{\Delta_k, k = 1, \dots, n_\Delta\}$ of neighborhood structures to be used for the local search. A pseudocode of the VND algorithm is given in what follows.

1. *Initialization:* Do $k = 1$.
2. *Search start:* Do $\hat{S} = S, improve = false$.
3. *Local Search:* Find the best solution $\bar{S} \in \Delta_k(\hat{S})$ belonging to the k^{th} neighborhood of \hat{S} .
4. If $f(\bar{S}) < f(\hat{S})$, do $\hat{S} = \bar{S}, improve = true$ and go back to step 3.
5. *Change of neighborhood structure:*
 - If *improve* do $k = 1$.
 - Otherwise do $k = k + 1$.
6. *Stopping condition:*
 - If $k \leq n_\Delta$ go back to step 2.
 - If $k > n_\Delta$ and *improve*, do $k = 1$ and go back to step 2. Otherwise, END: the best solution found is S .

VND outputs an improved solution which is a local optimum with respect to all neighborhood structures in Δ . Every time a VND execution finishes, a perturbation phase consisting in the performance of a certain number of random moves is applied, in order to escape from the current local optimum, and the VND is applied again to the perturbed solution. In addition to this, several enhancing features are introduced into this basic algorithm in order to improve its performance: more than one initial solution are generated and a different search is performed from each of them; an additional intensification phase is carried out, starting from the best improved initial solution; different orderings of the neighborhood structures used in the VND are considered, choosing randomly among them according to certain probabilities; if the current solution could not be improved after many iterations or it is too far from the best known solution, the search process is restarted; to avoid undoing perturbation moves, tabu lists for the different involved operators are used; the temporal relaxation of some precedence and capacity constraints, controlled through several infeasibility measures and correction procedures to ensure feasibility, is considered for diversification purposes.

The algorithm is fed initial solutions generated in two ways. The first one consists in solving the particular case with only one stack, whose solutions are always feasible for any loading plan with any container configuration. The advantage of this problem is that it reduces to a standard traveling salesman problem in a network whose arc weights are the sum of the weights of the two networks of the original problem. The second one is by using a simple randomized procedure, properly guided in order to ensure the feasibility of the obtained solutions. This second procedure is important to generate a wide variety of initial solutions.

In addition, the two traveling salesman problems induced by dropping all loading constraints on both the pickup and the delivery region of any instance are solved to optimality independently in order to obtain a lower bound: no container configuration or loading pattern can possibly improve the sum of the costs of the two optimal traveling salesman problem solutions. This has been done by using the TSP function of the optimizer OPTMODEL NETWORK from SAS software. This function implements a variant of the branch-and-cut algorithm by Applegate et al. [32], which is one of the most efficient exact methods available in the literature. Default parameters regarding the use of cutting planes, heuristics, node, and branching variable selection, identified as AUTOMATIC by SAS, were used.

4. Computational Study

Test instances for the double traveling salesman problem with multiple stacks were introduced by Petersen and Madsen [1], consisting of twenty instances with 33 customers, and twenty instances with 66 customers. These instances are used in four different experiments to evaluate how the choice of containers and packing patterns influences the transportation costs. The experimental design is simply to solve each of the forty basic instances based on different container configurations and compare the results, rather than relying on more complex experimental designs [33]. The results are obtained by applying the heuristic described in Section 3.2.

The heuristic was run on an Intel Core i5-3210M CPU 2.50 GHz with 6 GB RAM, with a running time of 10 min. for the instances with 33 customers, and 30 min. for the instances with 66 customers.

A lower bound for each instance has been calculated by solving separately the two independent traveling salesman problems that arise when dropping the precedence constraints that are associated with the last-in-first-out principle imposed on the loading of the container. These lower bounds, obtained by using the SAS software, represent the minimum costs that must be covered, even in the absence of loading constraints.

The first experiment considers a container loaded from the back with EUR-1 pallets. The standard packing pattern is $R = 3$ and $L = 11$, for a total capacity of 33 pallets. Assume that the company needs the full capacity to serve regular customers. Now, if the company has an extra demand for transportation of one pallet from the depot of the pickup region to the depot of the unloading region, one solution is to use a different packing pattern, such as $R = 2$ and $L = 17$, for a total capacity of 34 pallets.

Table 2 provides results showing how the total transportation cost changes for 20 standard test instances for selected packing patterns. The table shows, for each instance, the lower bound from solving traveling salesman problems of each region separately ("LB"), the transportation cost ("Cost"), and the relative deviation between the lower bound and the cost ("Dev"). For the first experiment, the average deviation from the lower bound increases from 14.1% to 28.4% when switching packing patterns to facilitate one extra pallet.

The second experiment tests the effect of introducing an open side container, instead of a container loaded from the back. Suitable open side containers may require additional customization, as compared to the more standard containers loaded from the back. To balance this, one would expect that the transportation costs can be reduced, as an alternative packing pattern with many short rows is possible, yielding much flexibility in the routing decisions. Table 2 shows the results for the most flexible alternative packing pattern with $R = 17$ and $L = 2$. For each of the 20 test instances, the heuristic is able to find a solution that matches the lower bound, providing a reduction in the transportation costs of 12% when compared to the use of a standard container loaded from the back ($R = 3$, $L = 11$). In the case where an additional pallet is transported between the depots, the saving in transportation cost amounts to 22% when using $R = 17$ and $L = 2$ instead of $R = 2$ and $L = 17$.

In the third experiment, the transportation of EUR-6 pallets is considered. Assuming that 66 pallets are to be transported in a regular 45-foot pallet wide container, the standard packing pattern of the test instances could be used, with $R = 3$ and $L = 22$. However, for EUR-6 pallets, an alternative pattern is available, with $R = 4$ and $L = 17$, while using the same type of container. This is expected to lead to lower transportation costs, as the added row provides more flexibility. Table 3 confirms this, as the average deviations to the lower bound are 29.7% and 18.7% for $R = 3$ and $R = 4$, respectively.

The fourth experiment deals with vertically stacked EUR-1 pallets and a demand of 66 pallets to be transported. For a standard container, the standard packing pattern has $R = 3$ stacks with $L = 22$ pallets, where pallets in each row are placed in two layers on top of each other. If an open side container is available, however, a packing pattern with $R = 11$ stacks of length $L = 6$ can be used. Table 3 shows that the open side container again leads to significantly reduced transportation costs, saving almost 21% when compared to the container loaded from the back. However, for this setting, there is still a 2.8% gap from the solutions obtained to the lower bound, meaning that it is still important to consider loading decisions when optimizing the routes.

Table 2. Results for instances with 33 customers and three different container configurations, given as ($R * L$).

Instance	LB	Baseline (3 * 11)		Experiment 1 (2 * 17)		Experiment 2 (17 * 2)	
		Cost	Dev	Cost	Dev	Cost	Dev
R00	911	1063	16.7%	1237	35.8%	911	0.0%
R01	875	1032	17.9%	1152	31.7%	875	0.0%
R02	935	1065	13.9%	1194	27.7%	935	0.0%
R03	961	1100	14.5%	1270	32.2%	961	0.0%
R04	937	1068	14.0%	1186	26.6%	937	0.0%
R05	900	1008	12.0%	1132	25.8%	900	0.0%
R06	998	1110	11.2%	1256	25.9%	998	0.0%
R07	963	1105	14.7%	1250	29.8%	963	0.0%
R08	978	1123	14.8%	1258	28.6%	978	0.0%
R09	976	1095	12.2%	1211	24.1%	976	0.0%
R10	901	1016	12.8%	1150	27.6%	901	0.0%
R11	892	1001	12.2%	1115	25.0%	892	0.0%
R12	984	1111	12.9%	1241	26.1%	984	0.0%
R13	956	1085	13.5%	1200	25.5%	956	0.0%
R14	879	1039	18.2%	1181	34.4%	879	0.0%
R15	985	1142	15.9%	1287	30.7%	985	0.0%
R16	967	1093	13.0%	1227	26.9%	967	0.0%
R17	946	1073	13.4%	1211	28.0%	946	0.0%
R18	1008	1126	11.7%	1269	25.9%	1008	0.0%
R19	938	1091	16.3%	1211	29.1%	938	0.0%
Average			14.1%		28.4%		0.0%

Table 3. Results for instances with 66 customers and three different container configurations, given as ($R * L$).

Instance	LB	Baseline (3 * 22)		Experiment 3 (4 * 17)		Experiment 4 (11 * 6)	
		Cost	Dev	Cost	Dev	Cost	Dev
R00	1237	1653	33.6%	1491	20.5%	1263	2.1%
R01	1257	1641	30.5%	1474	17.3%	1301	3.5%
R02	1295	1666	28.6%	1513	16.8%	1318	1.8%
R03	1290	1640	27.1%	1553	20.4%	1315	1.9%
R04	1295	1626	25.6%	1511	16.7%	1338	3.3%
R05	1204	1545	28.3%	1406	16.8%	1215	0.9%
R06	1294	1702	31.5%	1551	19.9%	1355	4.7%
R07	1307	1669	27.7%	1525	16.7%	1329	1.7%
R08	1297	1649	27.1%	1501	15.7%	1348	3.9%
R09	1276	1617	26.7%	1509	18.3%	1328	4.1%
R10	1339	1734	29.5%	1559	16.4%	1383	3.3%
R11	1268	1624	28.1%	1477	16.5%	1313	3.5%
R12	1295	1671	29.0%	1558	20.3%	1338	3.3%
R13	1275	1635	28.2%	1519	19.1%	1292	1.3%
R14	1245	1655	32.9%	1516	21.8%	1297	4.2%
R15	1228	1623	32.2%	1515	23.4%	1254	2.1%
R16	1356	1758	29.6%	1602	18.1%	1390	2.5%
R17	1274	1711	34.3%	1521	19.4%	1315	3.2%
R18	1328	1761	32.6%	1577	18.8%	1375	3.5%
R19	1256	1651	31.4%	1529	21.7%	1272	1.3%
Average			29.7%		18.7%		2.8%

The experiments show that the choice of container configuration is important. Using a standard container loaded from the rear leads to transportation routes that are much longer and more expensive than what could be achieved with a container loaded from the side. It seems that existing heuristic solution methods can cope very well with the high number of stacks that result from an open

side container. One might conjecture that certain exact methods, such as those developed by Lusby et al. [34], may benefit significantly when solving the problem with open side containers.

5. Concluding Remarks

The double traveling salesman problem with multiple stacks, introduced by Petersen and Madsen [1], has received substantial interest from researchers. The problem was based on real-world transportation requirements, which represented a novel challenge in the way that routing decisions were combined with packing decisions. As the problem provided ample venues for research on theoretical aspects of the problem, as well as on the development of efficient heuristic and exact solution methods, it might be valid to ask whether additional aspects of the real-world application may have been overlooked.

5.1. Main Conclusions

The implied choice of container technology in the double traveling salesman problem with multiple stacks was examined in this paper. Additional test instances can be derived, depending on the packing pattern used, and on whether the container is loaded from the back or from the side. It was demonstrated that the total transportation cost is highly dependent on the container choice, in some cases resulting in savings of more than 20% by allowing for the use of open side containers.

5.2. Limitations

The current study might not have exhausted all relevant variations of container technologies. As further examples, one could consider double door containers, where doors are available on both short sides of the container. This means that the container can be loaded while using one of the doors in the pickup-region and unloaded using either of the doors in the delivery-region, implying that there is a choice between last-in-first-out loading and first-in-first-out loading. Yet, other variants can be derived using open top containers, where goods are loaded and unloaded from the top side of the container, given that suitable equipment for loading and unloading is available at each customer location.

Different loading decisions may require the consideration of load stability, depending on the weight of the goods transported. Consider as an example Figure 2: if each pallet has a significant weight, a load as indicated might lead to dangerous situations on the road, as the content of the container is much heavier on one side. The issue of stability might be more critical when using open side containers, but the increased flexibility might nevertheless be economically attractive, at the expense of dealing with an optimization problem that has additional constraints to ensure stability at each leg of the routes.

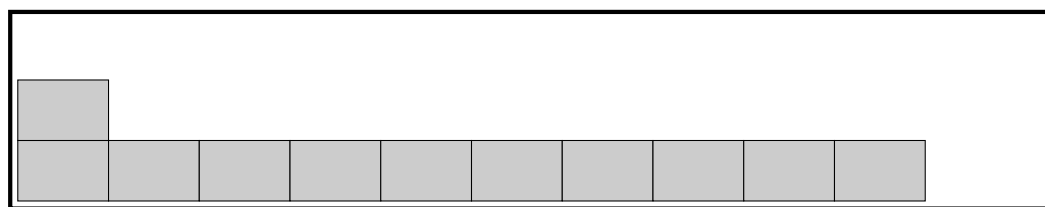


Figure 2. Illustration of stacking with $R = 3$ and $L = 11$ that may violate stability restrictions if heavy goods is transported.

Another issue not tackled in current research is when some customers have more than one pallet to be transported. The customer might require that only one visit is made, but might also require that different pallets are delivered to different locations.

Author Contributions: Conceptualization, L.M.H. and G.T.; methodology, G.T. and Á.F.; software, G.T. and Á.F.; validation, G.T.; formal analysis, G.T. and L.M.H.; investigation, L.M.H., G.T. and Á.F.; resources, G.T. and Á.F.; data curation, G.T.; writing—original draft preparation, L.M.H.; writing—review and editing, G.T. and Á.F.; visualization, L.M.H.; supervision, L.M.H.; project administration, L.M.H. and G.T.; funding acquisition, L.M.H. All authors have read and agreed to the published version of the manuscript.

Funding: The first author was supported by the AXIOM project, partially funded by the Research Council of Norway. The second author was supported by the Government of Spain, grant MTM2015-65803-R, and the local Government of Madrid, grant S2013/ICE-2845 (CASI-CAM-CM).

Acknowledgments: The authors wish to thank the three anonymous reviewers that contributed with useful inputs to help improve the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Petersen, H.; Madsen, O. The double travelling salesman problem with multiple stacks—Formulation and heuristic solution approaches. *Eur. J. Oper. Res.* **2009**, *198*, 139–147. [[CrossRef](#)]
- Felipe, Á.; Ortuño, M.; Tirado, G. The double traveling salesman problem with multiple stacks: A variable neighborhood search approach. *Comput. Oper. Res.* **2009**, *36*, 2983–2993. [[CrossRef](#)]
- Felipe, Á.; Ortuño, M.; Tirado, G. Using intermediate infeasible solutions to approach vehicle routing problems with precedence and loading constraints. *Eur. J. Oper. Res.* **2011**, *211*, 66–75. [[CrossRef](#)]
- Urrutia, S.; Milanés, A.; Løkketangen, A. A dynamic programming based local search approach for the double traveling salesman problem with multiple stacks. *Int. Trans. Oper. Res.* **2015**, *22*, 61–75. [[CrossRef](#)]
- Petersen, H.; Archetti, C.; Speranza, M. Exact solutions to the double travelling salesman problem with multiple stacks. *Networks* **2010**, *56*, 229–243. [[CrossRef](#)]
- Lusby, R.; Larsen, J. Improved exact method for the double TSP with multiple stacks. *Networks* **2011**, *58*, 290–300. [[CrossRef](#)]
- Alba Martínez, M.; Cordeau, J.F.; Dell’Amico, M.; Iori, M. A branch-and-cut algorithm for the double traveling salesman problem with multiple stacks. *INFORMS J. Comput.* **2013**, *25*, 41–55. [[CrossRef](#)]
- Carrabs, F.; Cerulli, R.; Speranza, M.G. A branch-and-bound algorithm for the double TSP with two stacks. *Networks* **2013**, *61*, 58–75. [[CrossRef](#)]
- Barbato, M.; Grappe, R.; Lacroix, M.; Calvo, R.W. A set covering approach for the double traveling salesman problem with multiple stacks. In *International Symposium on Combinatorial Optimization*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 260–272.
- Barbato, M.; Grappe, R.; Lacroix, M.; Calvo, R. Polyhedral results and a branch-and-cut algorithm for the double traveling Salesman problem with multiple stacks. *Discret. Optim.* **2016**, *21*, 25–41. [[CrossRef](#)]
- Casazza, M.; Ceselli, A.; Nunkesser, M. Efficient algorithms for the double traveling salesman problem with multiple stacks. *Comput. Oper. Res.* **2012**, *39*, 1044–1053. [[CrossRef](#)]
- Toulouse, S.; Calvo, R. On the complexity of the multiple stack TSP, kSTSP. In *International Conference on Theory and Applications of Models of Computation*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 360–369.
- Bonomo, F.; Mattia, S.; Oriolo, G. Bounded coloring of co-comparability graphs and the pickup and delivery tour combination problem. *Theor. Comput. Sci.* **2011**, *412*, 6261–6268. [[CrossRef](#)]
- Iori, M.; Riera-Ledesma, J. Exact algorithms for the double vehicle routing problem with multiple stacks. *Comput. Oper. Res.* **2015**, *63*, 83–101. [[CrossRef](#)]
- da Silveira, U.E.F.; Benedito, M.P.L.; dos Santos, A.G. Heuristic approaches to double vehicle routing problem with multiple stacks. In *Proceedings of the 2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, Marrakesh, Morocco, 14–16 December 2015; pp. 231–236.
- Chagas, J.B.; Silveira, U.E.; Benedito, M.P.; Santos, A.G. Simulated annealing metaheuristic for the double vehicle routing problem with multiple stacks. In *Proceedings of the 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Rio de Janeiro, Brazil, 1–4 November 2016; pp. 1311–1316.
- Chagas, J.; Silveira, U.; Santos, A.G.; Souza, M. A variable neighborhood search heuristic algorithm for the double vehicle routing problem with multiple stacks. *Int. Trans. Oper. Res.* **2020**, *27*, 112–137. [[CrossRef](#)]
- Côté, J.F.; Gendreau, M.; Potvin, J.Y. Large neighborhood search for the pickup and delivery traveling salesman problem with multiple stacks. *Networks* **2012**, *60*, 19–30. [[CrossRef](#)]

19. Pereira, A.H.; Urrutia, S. Formulations and algorithms for the pickup and delivery traveling salesman problem with multiple stacks. *Comput. Oper. Res.* **2018**, *93*, 1–14. [[CrossRef](#)]
20. Sampaio, A.H.; Urrutia, S. New formulation and branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks. *Int. Trans. Oper. Res.* **2017**, *24*, 77–98. [[CrossRef](#)]
21. Cherkesly, M.; Desaulniers, G.; Irnich, S.; Laporte, G. Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks. *Eur. J. Oper. Res.* **2016**, *250*, 782–793. [[CrossRef](#)]
22. Doerner, K.F.; Fuellerer, G.; Hartl, R.F.; Gronalt, M.; Iori, M. Metaheuristics for the vehicle routing problem with loading constraints. *Netw. Int. J.* **2007**, *49*, 294–307. [[CrossRef](#)]
23. Gendreau, M.; Iori, M.; Laporte, G.; Martello, S. A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Netw. Int. J.* **2008**, *51*, 4–18. [[CrossRef](#)]
24. Iori, M.; Salazar-González, J.J.; Vigo, D. An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transp. Sci.* **2007**, *41*, 253–264. [[CrossRef](#)]
25. Fuellerer, G.; Doerner, K.F.; Hartl, R.F.; Iori, M. Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *Eur. J. Oper. Res.* **2010**, *201*, 751–759. [[CrossRef](#)]
26. Chagas, J.B.; Toffolo, T.A.; Souza, M.J.; Iori, M. The double traveling salesman problem with partial last-in-first-out loading constraints. *arXiv* **2019**, arXiv:1908.08494.
27. Iori, M. An annotated bibliography of combined routing and loading problems. *Yugosl. J. Oper. Res.* **2016**, *23*, 782–793. [[CrossRef](#)]
28. Sörensen, K.; Glover, F. Metaheuristics. In *Encyclopedia of Operations Research and Management Science*, 3rd ed.; Gass, S., Fu, M., Eds.; Springer: Boston, MA, USA, 2013; pp. 960–970.
29. Jäntschi, L.; Bolboacă, S.; Bălan, M.; Sestras, R.; Diudea, M. Results of Evolution Supervised by Genetic Algorithms. *Not. Sci. Biol.* **2010**, *2*, 12–15. [[CrossRef](#)]
30. Hvattum, L.; Løkketangen, A.; Glover, F. Adaptive Memory Search for Boolean Optimization Problems. *Discret. Appl. Math.* **2004**, *142*, 99–109. [[CrossRef](#)]
31. Mladenović, N.; Hansen, P. Variable neighborhood search. *Comput. Oper. Res.* **1997**, *24*, 1097–1100. [[CrossRef](#)]
32. Applegate, D.L.; Bixby, R.E.; Chvatal, V.; Cook, W.J. *The Traveling Salesman Problem: A Computational Study*; Princeton University Press: Princeton, NJ, USA, 2006.
33. Bolboacă, S.; Jäntschi, L. Design of Experiments: Useful Orthogonal Arrays for Number of Experiments from 4 to 16. *Entropy* **2007**, *9*, 198–232. [[CrossRef](#)]
34. Lusby, R.; Larsen, J.; Ehrgott, M.; Ryan, D. An exact method for the double TSP with multiple stacks. *Int. Trans. Oper. Res.* **2010**, *17*, 637–652. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).