A new drayage problem with different customer services and container requirements

Federica Bomboi*
Department of Mathematics and Computer Science,
University of Cagliari, Italy

Jonas Pruente †
Fakultät für Mathematik,
TU Dortmund, Germany

January 13, 2020

Abstract

This paper investigates a drayage problem generalizing a previously proposed, which is motivated by a case study of a real maritime carrier. The aim of this study is to enlarge the supply of the types of service to fulfill additional needs of the customers. A Set Partitioning formulation and two scalable heuristics of route selecting are proposed. The experimentation shows that the proposed enlargement of services can be managed to the optimum or with a small error.

1 Introduction

Drayage is a popular research area in the field of transportation and logistics. It concerns the distribution of empty and loaded containers between intermodal facilities (e.g. ports or railway terminals), export and import customers. This paper is motivated by the real case study of a maritime medium-sized carrier which adopts a fleet of trucks and containers based at a port to provide door-to-door transportation customer services in the landside, Ghezelsoflu et al. [3]. The aim of this study is to generalize the characteristics of the drayage problem stated in Ghezelsoflu et al. [3]. The main advance in their work was testing a new distribution policy for the carrier in which exporters can also be served after importers other than the street-turn methodology currently adopted by the

*Federica Bomboi gratefully acknowledges Sardinia Regional Government for the financial support of her PhD scholarship (P.O.R. Sardegna F.S.E. Operational Programme of the Autonomous Region of Sardinia, European Social Fund 2007-2013 - Axis IV Human Resources, Objective l.3, Line of Activity l.3.1.)

†This work has been supported by the German Research Foundation within the Research Training Group 1855.

carrier. The first additional characteristic considered in this paper is motivated by the reality of some special requests. It occurs that some container loads are not allowed to be carried with other loaded containers at the same time, e.g. containers containing flammable commodities like tobacco and alcohol or commodities with too large weight. Therefore, it is important for the carrier to meet these customer requests. For the sake of clarity, in this paper containers which are carrying these special loads are referred to as "special" and the other containers are called "ordinary". Secondly, in this study the customer is allowed to choose between two ways of service, "stay-with" and "drop & pick". In most of the related literature, containers are left at customer locations by trucks bypassing packing and unpacking works. Since drivers do not wait for containers during loading and unloading operations, they can serve additional customers in the meanwhile. This type of service is known as drop & pick. In the stay-with service, drivers wait for containers during packing and unpacking operations and trucks carry the same container before and after the customer service. In Ghezelsoflu et al. [3], only the stay-with service is adopted. A paper in which stay-with and drop & pick are adopted together is Funke and Kopfer [2]. In their study, the decision of the way of service is not up to the customer like in this paper, but made by the algorithm. However, the method we present is flexible to manage instances in which a subset of customers require a fixed service type and for the others the algorithm can choose their service type. There exist various reasons for the customers to choose one of the two service types. Drop & pick services increase the flexibility of the customers because they can load or unload the container when it can be integrated optimally in their working schedule. Since drop & pick services last less time and can be integrated easier into existing routes, they can be reserved with less advance than stay-with services. Nevertheless, stay-with services need to be considered also because not all customers want to invest in a more expensive equipment that is necessary for drop & pick services. In line with the standards in many countries, the fleet of vehicles consists of trucks carrying one or two 24.5 ft containers. Empty containers need to be provided to and collected from customers in order to meet their transportation requests. Two ways to provide empty containers are regarded: they are picked up in the port or directly carried from importers to exporters according to the street-turn policy, Jula et al. [5], Deidda et al. [1]. For a detailed survey about the literature on drayage problems, we suggest the recent papers Song et al. [8] and Ghezelsoflu et al. [3]. This work can be seen as an important extension of the paper of Ghezelsoflu et al. [3]. In their problem statement, the trucks are able to carry more than one container and all customers are served according to the stay-with service type. They showed some improvements in the current carrier policy solving the problem in which importers do not have the priority to be served before all exporters. The two additional characteristics of this study are the inclusion of drop & pick services and the introduction of special loads. The problem statement formulated by Funke and Kopfer [2] has a lot of similarities to ours because they use trucks which can carry more than one container and they consider both types of services. Despite this, there are some major differences. The most relevant difference is that they assume that

every customer is prepared for a drop & pick service and therefore the cargo transportation company can decide during the optimization process which service they want to use. In our problem, to reach a more realistic setting, the customers are free to decide in advance if they want a stay-with or a drop & pick service, so that all the types of request are fixed before the optimization process. We consider homogeneous containers of 24.5 ft size, instead they use 20 ft and 40 ft containers, which can not be coupled with any other container. Hence, it is not possible to model 40 ft containers as special containers because special containers can be coupled with empty containers. In addition to this, they distinguish also between 20 ft and 40 ft empty containers. Furthermore, they introduce one depot for the storage of empty containers when necessary. In contrast to our drop & pick rules, they assume to end this kind of service on the same day by visiting the customer twice, either with the same truck or another. They solve the problem with a node-arc model for small instances. Ileri et al. [4] consider stay-with and drop & pick orders that are fixed from the beginning. They assume that drivers start and end the working day without a trailer. Additionally, they introduce trailer pools, in which infinitely many empty containers are stored. Drivers can be one of two types-company driver (CD) and third party (TP). This influences the objective function that is minimizing the total cost for the company. The main difference to this paper is that they have only one container per truck. They use a Set Partitioning formulation and solve the relaxed problem with a column generation approach. Also Xue et al. [9] regard both types of service and assume that the customer decides which service to take. They have different additional depots and only one container per truck is allowed. They solve the problem with an algorithm based on window partitions. In conclusion, in best of our knowledge, our problem statement is innovative and not directly comparable with the other problem statements in the literature. It integrates many realistic characteristics from different papers and is therefore worth to be regarded. In particular, the presented Set Partitioning formulation is very flexible, it can handle a lot of different constraints and can be used for various problem statements. Especially for this problem statement, it is more effective than other methods presented in the literature. In section 2, we propose a mathematical model for the problem statement and a Set Partitioning formulation is presented. The section ends with an analysis on how the types of request influence the number of feasible routes and therefore the running time. Two heuristics with scalable parameters to find either faster or more precise solutions are proposed in section 3. In section 4, we compare the two heuristics with the exact solution in an extensive experimental evaluation.

2 Modelling

We start by defining the set of customer service types as follows:

- PE: set of requests in which an empty container must be picked up from an importer during drop & pick service;

- DE: set of requests in which an empty container must be dropped to an exporter during drop & pick service.
- PLO: set of requests in which an ordinary loaded container must be picked up from an exporter during drop & pick service;
- PLS: set of requests in which a special loaded container must be picked up from an exporter during drop & pick service;
- DLO: set of requests in which an ordinary loaded container must be delivered to an importer during drop & pick service;
- DLS: set of requests in which a special loaded container must be delivered to an importer during drop & pick service.
- $PLSWO$: set of requests of exporters with stay-with services and ordinary containers;
- $PLSWS$: set of requests of exporters with stay-with services and special containers;
- $DLSWO$: set of requests of importers with stay-with services and ordinary containers;
- $DLSWS$: set of requests of importers with stay-with services and special containers.

The difference between special and ordinary loads is that a special load can not be carried together with another loaded container. In order to define the feasible sequences of visits among these types of requests, we describe the state of the truck slots after each type of customer request and the types of requests that can be served for each possible state. Let $s_{rj}(i)$ be the state of truck slot j after visiting customer request i in route r. Since trucks carry up to two containers, $j \in \{1, 2\}$. The state of any container slot is one of the following entries: $\{Absent, ClosedO, ClosedS, Empty, InitialO, InitialS\}$, which are defined hereafter:

- $s_{rj}(i) = Absent$ means that no container is placed in slot j;
- $s_{rj}(i) = ClosedO$ means that an ordinary loaded container is put in slot j after serving an export request $i \in PLSWO \cup PLO$;
- $s_{rj}(i) = ClosedS$ means that a special loaded container is put in slot j after serving an export request $i \in PLSWS \cup PLS$;
- $s_{rj}(i) = Empty$ means that the container in slot j is empty;
- $s_{rj}(i) = InitialO$ means that an ordinary container in slot j is loaded to satisfy an importer request in set $DLSWO \cup DLO$;
- $s_{rj}(i) = InitialS$ means that a special container in slot j is loaded to satisfy an importer request in set $DLSWS \cup DLS$.

Type	*Pre slot*	*Forbidden Pre other slot*	*Post slot*
PE	*Absent*	-	*Empty*
DE	*Empty*	-	*Absent*
PLO	*Absent*	$ClosedS/InitialS$	$ClosedO$
PLS	*Absent*	$Closed^*/Initial^*$	$ClosedS$
DLO	$InitialO$	-	*Absent*
DLS	$InitialS$	-	*Absent*
$PLSWO$	*Empty*	$ClosedS/InitialS$	$ClosedO$
$PLSWS$	*Empty*	$Closed^*/Initial^*$	$ClosedS$
$DLSWO$	$InitialO$	-	*Empty*
$DLSWS$	$InitialS$	-	*Empty*

Table 1: Changes in slot states

According to this notation, the possible initializations of the two container slots in a truck leaving the port are:

- $InitialO$ & $InitialO$
- $InitialO$ & *Empty*
- $InitialO$ & *Absent*
- $InitialS$ & *Empty*
- $InitialS$ & *Absent*
- *Empty* & *Empty*
- *Empty* & *Absent*
- *Absent* & *Absent*

In the list of possible truck initializations it is taken into account that the special loaded containers can be carried alone in a truck or coupled only with empty containers. To determine feasible routes for two-containers trucks, both container slots have to be regarded. Table 1 shows in column *Pre slot* the required state of a slot in order to serve the type of request reported in the first column *Type*. For example, the state of a slot must be *Absent* in order to pick an empty container and put the container in that slot. In the column *Forbidden Pre other slot* it is illustrated which status is not allowed for the second slot to accept the type of request reported in the first column. $Closed^*$ stands for $ClosedO$ or $ClosedS$ and $Initial^*$ respectively. Column *Post slot* indicates the state of the used slot after the service of a customer request reported in the corresponding line.

The set of feasible routes is obtained by connecting acceptable sequences of customer requests according to Table 1. Therefore a dynamic program over the number of customers in a route is used. To obtain routes with $k+1$ customers, all routes with k customers are regarded and for every customer it is checked if adding it leads to a feasible route. This has to be done for all possible truck initializations because there exist routes that are only feasible for one of the initializations.

For the presented problem statement we prefer the Set Partitioning model over the node-arc formulation because the problem is highly constrained and with a growing number of constraints the running time of the route generation and of the Set Partition model decreases in general, as opposed to node-arc

formulations that become harder to solve. Given the previous set of feasible routes, we present a Set Partitioning model using the following notation:

R: Set of all feasible routes

C: Set of customer requests

d_r: Cost route $r \in R$;

α_{ir}: Coefficient with value 1 if route $r \in R$ covers $i \in C$, 0 otherwise;

x_r : Binary variable which is 1 if route $r \in R$ is selected, 0 otherwise.

The Set Partitioning formulation of the model is:

$$\begin{aligned} \min \quad & \sum_{r \in R} d_r x_r & & (1) \\ \text{s.t.} \quad & \sum_{r \in R} \alpha_{ir} x_r = 1 & \forall i \in C & (2) \\ & x_r \in \{0,1\} & \forall r \in R & (3) \end{aligned}$$

In (1) the cost of the selected routes are minimized. Constraints (2) ensure that all container requests are met by exactly one route. Finally, (3) defines the domain of the decision variable.

The ability of solving the previous model by a standard mixed-integer programming solver depends on the number of feasible routes. A general formula is derived to determine $|R|$ as function of the types of requests, but it is too large to be reported in the paper. In the basic problem statement it is possible to compute the exact number of routes in less than 1 second. For the problem with time window constraints, 8 hours working day constraints and other special constraints deriving from the heuristics (section 3), we estimate the number of routes by sampling. The knowledge of the number of routes for a given instance is important to decide which algorithm to apply. The maximization of this function shows that, for a fixed number of customers, the number of feasible routes is maximum when half of the customers requests belongs to the set PE and the other half belongs to DE. Indeed, a problem with a huge number of customers and less PE and DE requests could have a lower number of feasible routes than a problem with less customers but more PE and DE requests. Table 2

$\|C\|$	*Type of request*	$\|R\|_{max}$
20	without PE - DE	20440
20	with PE - DE	1,4 billion
100	without PE - DE	12,5 million
100	with PE - DE	50 trillion

Table 2: Example for number of feasible routes

shows the maximum number of the feasible routes for four types of instances in

the case of up to 8 customers per route and illustrates the huge impact of PE and DE customers on the number of feasible routes. In the first column the number of customers $|C|$ for every instance is indicated. The column $|R|_{max}$ represents the maximum number of feasible routes. In the first and third instances, the maximum is reached when half of the customers requests are PLO and the other half are DLO. In the second and fourth instances, the maximum is reached when half of the customers requests are PE and the other half are DE.

3 Heuristics

In this section, two heuristics are proposed to select a subset of routes from the overall set R of feasible routes. This subset of routes will replace R in the model in section 2 and result in a restricted Set Partitioning problem. Therefore, these heuristics can be seen as rules to leave out routes from R that are not likely to appear in the optimal solution. The heuristics are described by the example in Table 3, where only 4 customer requests are considered. They are denoted by $C1$, $C2$, $C3$ and $C4$.

Name	*Type of request*	*Coordinates*
Port	-	$(0, 0)$
C1	PE	$(0, 1)$
C2	DE	$(1, 1)$
C3	DE	$(1, 0)$
C4	PLS	$(2, 0)$

Table 3: Example for the heuristics

- Heuristic 1: since customers of type PE and DE highly increase the total number of routes, the first heuristic requires a limit lim on the number of customers of these types in each route. For instance, if $lim = 2$, only routes with at most 2 customers which have type PE or DE are regarded. Following this idea, one eliminated route is $\{Port, C1, C2, C3, Port\}$ because this route contains three customers with type PE or DE, which is more than the allowed limit lim. A larger value of lim leads to more precision but to a higher running time.

- Heuristic 2: for every customer request $i \in C$, we make a list of the closest $m \leq |C| - 2$ requests. Then, in the routes generation process, a candidate can only be added to the route only if it is in the list of the previous customer in the route. This is motivated by the fact that a high number of long distance routes, which are not likely in the optimal solution, are not regarded. To ensure feasibility, the port can always be added to a route and from the port it is allowed to reach every customer. In our example, see Table 3, if we set $m = 2$, then one eliminated route is $\{Port, C1, C4, Port\}$ because $C4$ is not one of the m closest neighbors of

$C1$. A larger value of m leads to more precision but to a higher running time.

- Combination of Heuristic 1 and Heuristic 2: it applies both additional rules.

4 Experimentation

The presented model is implemented in Java 1.8.0_171 using IBM ILOG CPLEX Studio 12.6.0.0. Tests have been run on an Intel(R) Xeon(R) E7340 processor with 2.4 GHz. The time limit is set to 2 hours and memory limit to 32 GB. Artificial instances are built by a square of 1000x1000 units, where we generate random coordinates of the customers according to the Uniform distribution. The port is placed at $(0, 0)$. The first customers requests are labeled with type PE or DE according to how many PE or DE we want to have in the instances. Next, we set the request type of all other customers with equal probability to one of the remaining request types. The service time is supposed to be 30 minutes for stay-with requests and 15 minutes for drop & pick requests. The time windows range is always 6 hours long and can start from 6:00 am till 2:00 pm. Their initial time is generated by the Uniform distribution. Drivers can start their routes at every time point, therefore, they start exactly at the time point that is necessary to fulfill time windows constraints. We think that letting the drivers have flexible starting time is more realistic. After their first stop, the drivers are not allowed to wait to be in time at the customer location. Each route including the time of the service cannot take longer than 8 hours, which is a reasonable duration for a driver shift. In order to serve a stay-with customer with coordinates $(1000, 1000)$ within 8 hours, 1 minute must be equal to at least 6.29 units. Since this setting cuts too many routes and makes the problem too easy, we double it and set 1 minute to 12.58 units to obtain reasonable settings for the experimentation. According to this data, up to 8 customers can be visited in each route. At first, instances with time windows are considered. Although time windows make more realistic instances, they decrease the number of feasible routes and make the Set Partitioning problem easier to solve. Next, harder problem instances without time windows are tested. The results with time windows are reported in Table 4 and 6 and without time windows in Table 5 and 7 respectively. Every row concerns one group of 10 random instances. The first two columns of every table describe the problem data: the number of customers requests $|C|$, the number of requests of type PE and DE. For example the first row stands for the setting with 20 customers in which one of them asks for a DE request, no customer requires a PE service and the other customers have random request types among the remaining 8 request types. All four tables are based on the same instances. The results are summarized in five groups of columns from Table 4 to table 7. The first group of columns is denoted by *All Routes* and reports the case of complete enumeration of the overall set of feasible routes. The other groups of columns report the

results obtained by four reasonable calibrations of the heuristics proposed in Section 3.

More precisely, the second group of columns is denoted by $lim = 1$ because it refers to the case of a restricted Set Partitioning problem in which each feasible route has one PE or DE request at most. Its associated number of routes is denoted by $|R_1|$. The second group of columns is denoted by $lim = 2$ because it refers to the case of a restricted Set Partitioning problem in which each feasible route has two PE or DE requests at most. Its associated number of routes is denoted by $|R_2|$. The third group of columns is denoted by $m = 0.75|C|$ because it refers to the case of a restricted Set Partitioning problem in which each customer request is connected to 75 percent of the other requests. Its associated number of routes is denoted by $|R_3|$. Finally, the fourth group of columns is denoted by $lim = 2$ & $m = 0.75|C|$ because it refers to the case of a restricted Set Partitioning problem in which each feasible route has two PE or DE requests at most and each customer request is connected to 75 percent of the other requests. Its associated number of routes is denoted by $|R_4|$.

The following results are reported in the case of enumeration of all feasible routes: the average number $|R|$ of routes, the average sum of the route generation time and the running time (it is denoted by time and measured in seconds), the average percentage of the route generation time out of the previous sum (it is denoted by gen. %) and number of solved instances out of ten (it is denoted by s.i.). For each heuristic, we report the average number of routes, the average ratio between the solution obtained by that heuristic and the solution determined with all feasible routes if possible or with the best heuristic otherwise, the average sum of route generation time and the running time (it is denoted by time and measured in seconds) and number of solved instances out of ten (it is denoted by s.i.). The notation uses k for 1000 and — for the cases in which it was not possible to obtain a solution.

In this experimentation 3000 instances were run. In 27 instances the proposed methods were not able to determine a feasible solution: 5 of these instances have time windows, 22 do not. More precisely, 21 instances were not solved because of a lack of memory: in 9 of these instances it was not possible to generate all the routes, in the remaining 12 the lack of memory was observed during the running of the solver. Because of the time limit, 6 instances were not solved: 1 during the generation of the routes, 5 during the solver execution. According to this experimentation, 69.1% of the running time was used by Cplex and only 30.9% for the route generation. The percentage of route generation time was higher for the heuristics because of the additional time spent to check the feasibility of the routes.

In the case of complete enumeration of all routes, the instances with all feasible routes were always solved to the optimality or considered as not solved instances, thus they are a valid benchmark for the quality of solutions determined by the heuristics. The tests show that all the heuristics are much faster as opposed to the case of complete enumeration of all feasible routes, without losing much precision. The comparison between $lim = 1$ and $lim = 2$ shows that the first is faster and does not lead to better results in all instances. This is not surprising

because the first heuristic uses a subset of routes of the second one. In the row with 100 customers and $2PE$ $2DE$ of Table 4, the solution time of 411 seconds for the heuristic with $lim = 1$ is obtained, even if 3632 seconds are spent to solve an outlier instance. The comparison between the heuristic with $lim = 2$ and the heuristic with $m = 0.75|C|$ is more interesting: the last one is in general slower than the first one if there are more than 3 PE or DE customers in the instance and in the other cases is faster. Yet, this result is somehow expected because an increase in PE or DE customer requests does not affect the heuristic with $lim = 2$ crucially. Nevertheless, both heuristics lead to optimal and near-optimal solutions. Since the last heuristic incorporates the two heuristics, the solutions cannot be better than those obtained by the previous two heuristics, but the time is clearly lower of the time of both. Nevertheless, the solution quality keeps being good. Table 5 and 7 show the results for the same instances without time windows. For all settings, the number of routes increases, but the same comments made in the case of time windows still hold. Finally, during the 3000 tests we tested the formula mentioned in Section 2. The main goal of this formula is to predict the running time and the memory of a solution method for a given instance before running it by estimating the number of feasible routes for that instance. The number of routes estimated by the formula is compared to the number of generated routes. The average quotient between these two numbers was 1.000 in all tests. The average time of the estimation over all tests was 3.582 seconds and, in the case of instances with 10 million or more routes, the average time was 47.169 seconds. The maximum time the formula needed was 88 seconds. Since the correlation coefficient between the running time and the estimated number of routes was 0.813 (0.945 if we delete the outlier instance) and 0.991 between the memory and the estimated number of routes, we conclude that the formula is very effective. For the correlation coefficient about time, we considered only instances with at least 1 second running time because the other instances were so fast that they did not need a prediction and, because of the measurement in seconds, they result in distorted statistics.

		All Routes				$lim = 1$				$lim = 2$			
$\|C\|$		—R—	time	gen. %	s. i.	$\|R_1\|$	ratio	time	s. i.	$\|R_2\|$	ratio	time	s. i.
20	1DE 0PE	1 k	0	—	10	1 k	1.000	0	10	1 k	1.000	0	10
	1DE 1PE	4 k	0	—	10	2 k	0.999	0	10	4 k	1.000	0	10
	2DE 1PE	10 k	0	—	10	2 k	0.999	0	10	6 k	1.000	0	10
	2DE 2PE	24 k	0	—	10	2 k	0.981	0	10	8 k	0.999	0	10
	3DE 2PE	45 k	0	—	10	1 k	0.974	0	10	8 k	1.000	0	10
	3DE 3PE	113 k	0	—	10	2 k	0.943	0	10	12 k	0.997	0	10
40	1DE 0PE	16 k	0	—	10	16 k	1.000	0	10	16 k	1.000	0	10
	1DE 1PE	50 k	0	—	10	27 k	1.000	0	10	50 k	1.000	0	10
	2DE 1PE	89 k	0	—	10	26 k	0.999	0	10	67 k	1.000	0	10
	2DE 2PE	247 k	3	33.3	10	34 k	0.998	0	10	111 k	1.000	1	10
	3DE 2PE	535 k	7	14.3	10	31 k	0.998	0	10	131 k	1.000	2	10
	3DE 3PE	1752 k	29	20.7	10	41 k	0.996	0	10	214 k	1.000	3	10
60	1DE 0PE	56 k	0	—	10	56 k	1.000	0	10	56 k	1.000	1	10
	1DE 1PE	178 k	3	33.3	10	99 k	1.000	2	10	178 k	1.000	4	10
	2DE 1PE	383 k	7	28.6	10	114 k	1.000	2	10	286 k	1.000	6	10
	2DE 2PE	1245 k	26	23.1	10	148 k	1.000	3	10	419 k	1.000	12	10
	3DE 2PE	2336 k	50	24.0	10	179 k	0.999	4	10	713 k	1.000	16	10
	3DE 3PE	3962 k	87	21.8	10	149 k	0.998	3	10	709 k	1.000	17	10
80	1DE 0PE	172 k	4	25.0	10	172 k	1.000	6	10	172 k	1.000	6	10
	1DE 1PE	648 k	15	20.0	10	342 k	1.000	12	10	648 k	1.000	19	10
	2DE 1PE	1348 k	34	23.5	10	396 k	1.000	12	10	994 k	1.000	29	10
	2DE 2PE	2173 k	58	22.4	10	340 k	1.000	11	10	1074 k	1.000	34	10
	3DE 2PE	4437 k	124	22.6	10	486 k	1.000	15	10	1702 k	1.000	51	10
	3DE 3PE	10990 k	272	22.8	9	512 k	0.999	18	10	2400 k	1.000	78	10
100	1DE 0PE	552 k	20	15.0	10	552 k	1.000	26	10	552 k	1.000	26	10
	1DE 1PE	998 k	53	11.3	10	673 k	1.000	40	10	998 k	1.000	62	10
	2DE 1PE	2542 k	78	26.9	10	898 k	1.000	36	10	1994 k	1.000	75	10
	2DE 2PE	6494 k	285	18.2	10	1133 k	1.000	411	10	3555 k	1.000	151	10
	3DE 2PE	11182 k	424	25.0	10	1059 k	1.000	42	10	3929 k	1.000	144	10
	3DE 3PE	22381 k	726	30.6	6	1335 k	1.000	57	10	6078 k	1.000	249	10

Table 4: Results for exact solution and heuristic 1 with time windows

References

[1] L. Deidda, M. Di Francesco, A. Olivo, P. Zuddas, *Implementing the street-turn strategy by an optimization model*, Maritime Policy & Management (2008), 35:503-516.

[2] J. Funke, H. Kopfer, *A model for a multi-size inland container transportation problem.* Transportation Research Part E 89 (2016), 70-85.

[3] A. Ghezelsoflu, M. Di Francesco, A. Frangioni, P. Zuddas *A Set-Covering formulation for a drayage problem with single and double container loads*, Journal of Industrial Engineering International (2018), https://doi.org/10.1007/s40092-018-0256-8.

[4] Y. Ileri, M. Bazaraa, T. Gifford, G. Nemhauser, J. Sokol, E. Wikum, *An Optimization Approach for Planning Daily Drayage Operations*, CEJOR (2006), 14:141-156.

[5] H. Jula, A. Chassiakos, P. Ioannou, *Port dynamic empty container reuse*, Transportation Research Part E: Logistics and Transportation Review (2006), 42:43-60.

[6] M. Lai, T. G. Crainic, M. Di Francesco, P. Zuddas, *An heuristic search for the routing of heterogeneous trucks with single and double container loads*, Transportation Research Part E: Logistics and Transportation Review (2013), 56:108-118.

		All Routes				$lim = 1$				$lim = 2$			
$\|C\|$		—R—	time	gen. %	s. i.	$\|R_1\|$	ratio	time	s. i.	$\|R_2\|$	ratio	time	s. i.
20	1DE 0PE	2 k	0	—	10	2 k	1.000	0	10	2 k	1.000	0	10
	1DE 1PE	5 k	0	—	10	2 k	0.998	0	10	5 k	1.000	0	10
	2DE 1PE	15 k	0	—	10	3 k	0.999	0	10	9 k	1.000	0	10
	2DE 2PE	41 k	0	—	10	3 k	0.980	0	10	12 k	1.000	0	10
	3DE 2PE	74 k	0	—	10	2 k	0.978	0	10	11 k	0.999	0	10
	3DE 3PE	308 k	2	0.0	10	3 k	0.945	0	10	20 k	0.996	0	10
40	1DE 0PE	21 k	0	—	10	21 k	1.000	0	10	21 k	1.000	0	10
	1DE 1PE	84 k	1	0.0	10	40 k	1.000	0	10	84 k	1.000	1	10
	2DE 1PE	135 k	1	0.0	10	35 k	0.999	0	10	98 k	1.000	1	10
	2DE 2PE	518 k	7	14.3	10	49 k	0.996	0	10	186 k	1.000	2	10
	3DE 2PE	1150 k	18	22.2	10	45 k	0.998	0	10	214 k	1.000	3	10
	3DE 3PE	3081 k	53	20.8	10	52 k	0.997	0	10	306 k	1.000	4	10
60	1DE 0PE	86 k	1	0.0	10	86 k	1.000	1	10	86 k	1.000	1	10
	1DE 1PE	296 k	5	20.0	10	149 k	1.000	3	10	296 k	1.000	6	10
	2DE 1PE	565 k	10	20.0	10	150 k	1.000	3	10	405 k	1.000	9	10
	2DE 2PE	2249 k	48	25.0	10	225 k	1.000	5	10	846 k	1.000	20	10
	3DE 2PE	4811 k	106	22.6	10	262 k	0.999	6	10	1153 k	1.000	28	10
	3DE 3PE	8638 k	229	24.9	10	216 k	0.999	4	10	1164 k	1.000	27	10
80	1DE 0PE	235 k	6	33.3	10	235 k	1.000	7	10	235 k	1.000	8	10
	1DE 1PE	912 k	21	23.8	10	453 k	1.000	14	10	912 k	1.000	26	10
	2DE 1PE	2059 k	53	24.5	10	537 k	1.000	18	10	1462 k	1.000	46	10
	2DE 2PE	4501 k	115	22.6	10	500 k	1.000	18	10	1825 k	1.000	70	10
	3DE 2PE	10770 k	345	27.2	10	690 k	1.000	22	10	2929 k	1.000	92	10
	3DE 3PE	20237 k	616	29.7	7	748 k	0.999	28	10	4016 k	1.000	132	10
100	1DE 0PE	796 k	36	16.7	10	796 k	1.000	45	10	796 k	1.000	46	10
	1DE 1PE	1931 k	62	24.2	10	988 k	1.000	39	10	1931 k	1.000	68	10
	2DE 1PE	5256 k	161	28.0	10	1369 k	1.000	54	10	3631 k	1.000	141	10
	2DE 2PE	13711 k	511	22.7	9	1722 k	1.000	148	10	6129 k	1.000	257	10
	3DE 2PE	24554 k	926	29.2	7	1751 k	1.000	73	10	7640 k	1.000	320	10
	3DE 3PE	34487 k	1135	29.3	1	2124 k	1.000	100	10	10824 k	1.000	512	10

Table 5: Results for exact solution and heuristic 1 without time windows

[7] M. Lai, M. Battarra, M. Di Francesco, P. Zuddas, *An adaptive guidance meta-heuristic for the vehicle routing problem with splits and clustered backhauls*, Journal of the Operational Research Society (2015), 66:1222-1236.

[8] Y. Song, J. Zhang, Z. Liang, C. Ye, *An exact algorithm for the container drayage problem under a separation mode*, Transportation Research Part E 106 (2017) 231-254.

[9] Z. Xue, C. Zhang, W. H. Lin, L. Miao, P. Yang, *A tabu search heuristic for the local container drayage problem under a new operation mode*, Transportation Research Part E: Logistics and Transportation Review (2014), 62:136-150.

		$m = 0.75\|C\|$				$lim = 2$ & $m = 0.75\|C\|$			
$\|C\|$		$\|R_3\|$	ratio	time	s. i.	$\|R_4\|$	ratio	time	s. i.
20	1DE 0PE	0 k	0.998	0	10	0 k	0.998	0	10
	1DE 1PE	2 k	0.999	0	10	2 k	0.999	0	10
	2DE 1PE	5 k	0.997	0	10	3 k	0.997	0	10
	2DE 2PE	10 k	0.997	0	10	3 k	0.996	0	10
	3DE 2PE	14 k	0.999	0	10	3 k	0.999	0	10
	3DE 3PE	41 k	0.998	0	10	5 k	0.993	0	10
40	1DE 0PE	8 k	0.999	0	10	8 k	0.999	0	10
	1DE 1PE	20 k	1.000	0	10	20 k	1.000	0	10
	2DE 1PE	29 k	1.000	0	10	24 k	1.000	0	10
	2DE 2PE	102 k	0.999	1	10	46 k	0.999	0	10
	3DE 2PE	269 k	0.999	4	10	59 k	0.999	0	10
	3DE 3PE	710 k	0.999	12	10	83 k	0.999	1	10
60	1DE 0PE	25 k	1.000	2	10	25 k	1.000	0	10
	1DE 1PE	69 k	1.000	2	10	69 k	1.000	2	10
	2DE 1PE	150 k	1.000	4	10	114 k	1.000	3	10
	2DE 2PE	492 k	0.999	12	10	210 k	0.999	5	10
	3DE 2PE	897 k	1.000	22	10	284 k	1.000	7	10
	3DE 3PE	1653 k	1.000	39	10	279 k	1.000	8	10
80	1DE 0PE	76 k	1.000	5	10	76 k	1.000	5	10
	1DE 1PE	261 k	1.000	9	10	261 k	1.000	10	10
	2DE 1PE	473 k	1.000	15	10	353 k	1.000	12	10
	2DE 2PE	770 k	1.000	27	10	393 k	1.000	14	10
	3DE 2PE	1390 k	1.000	45	10	582 k	1.000	20	10
	3DE 3PE	4054 k	1.000	147	10	846 k	1.000	33	10
100	1DE 0PE	248 k	1.000	15	10	248 k	1.000	15	10
	1DE 1PE	342 k	1.000	24	10	342 k	1.000	24	10
	2DE 1PE	987 k	1.000	41	10	777 k	1.000	33	10
	2DE 2PE	2199 k	1.000	112	10	1298 k	1.000	106	10
	3DE 2PE	4409 k	1.000	179	10	1510 k	1.000	62	10
	3DE 3PE	8924 k	1.000	447	10	1900 k	1.000	84	10

Table 6: Results for heuristic 2 and combination of heuristics with time windows

		$m = 0.75\|C\|$				$lim = 2$ & $m = 0.75\|C\|$			
$\|C\|$		$\|R_3\|$	ratio	time	s. i.	$\|R_4\|$	ratio	time	s. i.
20	1DE 0PE	1 k	1.000	0	10	1 k	1.000	0	10
	1DE 1PE	3 k	0.999	0	10	3 k	0.999	0	10
	2DE 1PE	8 k	0.999	0	10	5 k	0.999	0	10
	2DE 2PE	15 k	0.998	0	10	5 k	0.998	0	10
	3DE 2PE	24 k	1.000	0	10	5 k	0.998	0	10
	3DE 3PE	112 k	0.999	0	10	9 k	0.995	0	10
40	1DE 0PE	10 k	0.999	0	10	10 k	0.999	0	10
	1DE 1PE	33 k	1.000	0	10	33 k	1.000	0	10
	2DE 1PE	41 k	1.000	0	10	33 k	1.000	0	10
	2DE 2PE	201 k	1.000	3	10	74 k	0.999	1	10
	3DE 2PE	533 k	1.000	8	10	92 k	1.000	1	10
	3DE 3PE	1232 k	1.000	22	10	121 k	1.000	2	10
60	1DE 0PE	38 k	1.000	0	10	38 k	1.000	0	10
	1DE 1PE	110 k	1.000	3	10	110 k	1.000	3	10
	2DE 1PE	211 k	1.000	5	10	157 k	1.000	4	10
	2DE 2PE	918 k	1.000	23	10	337 k	1.000	9	10
	3DE 2PE	1876 k	1.000	48	10	441 k	1.000	12	10
	3DE 3PE	3755 k	1.000	95	10	450 k	1.000	11	10
80	1DE 0PE	100 k	1.000	4	10	100 k	1.000	4	10
	1DE 1PE	356 k	1.000	11	10	356 k	1.000	11	10
	2DE 1PE	754 k	1.000	26	10	535 k	1.000	20	10
	2DE 2PE	1665 k	1.000	54	10	670 k	1.000	23	10
	3DE 2PE	3462 k	1.000	121	10	998 k	1.000	36	10
	3DE 3PE	8314 k	1.000	215	9	1346 k	1.000	51	10
100	1DE 0PE	358 k	1.000	25	10	358 k	1.000	25	10
	1DE 1PE	650 k	1.000	33	10	650 k	1.000	33	10
	2DE 1PE	2116 k	1.000	86	10	1423 k	1.000	59	10
	2DE 2PE	4702 k	1.000	199	10	2169 k	1.000	94	10
	3DE 2PE	11347 k	1.000	398	9	2911 k	1.000	131	10
	3DE 3PE	19108 k	1.000	496	6	3383 k	1.000	165	10

Table 7: Results for heuristic 2 and combination of heuristics without time windows