# An adaptive guidance meta-heuristic for the Vehicle Routing Problem with Splits and Clustered Backhauls

Michela Lai[a], Maria Battarra[b], Massimo Di Francesco[a], Paola Zuddas[a]

[a] *Department of Mathematics and Computer Science, University of Cagliari, Italy*
[b] *School of Mathematics, University of Southampton, UK*

**Abstract**

This paper presents a vehicle routing problem, where trucks deliver container loads from a port to import customers and collect container loads from export customers to the same port. In each route, import customers must be serviced before export customers and each customer can be visited more than once. We model the problem using an Integer Linear Programming formulation and propose an Adaptive Guidance metaheuristic. Our extensive computational experiments show that the adaptive guidance algorithm is capable of solving average-sized instances within limited computing time.

*Keywords:* Vehicle Routing Problem with Splits, Backhauls, Drayage, Adaptive Guidance, Meta-heuristics

## 1. Introduction

This paper addresses a vehicle routing problem motivated by the case study of the italian carrier Grendi Trasporti Marittimi, which provides *door-to-door* freight transportation services. The carrier manages a homogeneous fleet of trucks and containers based at the port of Vado Ligure (Italy). Trucks move container loads from the port to import customers and from export customers to the port.

It is important to note that in this problem containers are not picked up or delivered. They are brought to the customers, where they are packed or unpacked and moved away by the same trucks. Therefore, while containers are emptied at importer locations, drivers supervise the unloading operations and wait for empty containers to be returned. Similarly, trucks move empty containers to export customers, drivers supervise packing operations and wait for loaded containers to be returned. The truck and the containers are coupled in the sense that the truck carries the same set of containers throughout the route.

From the customer's point of view, this practice is perceived as a high quality service, because the loading and unloading operations are closely supervised and the integrity of the cargo is monitored. From the carrier's point of view, this policy improves container safety and integrity, because containers are never left unsupervised at customer locations.

More important, the carrier is aware of the fact that leaving containers at customer locations would save drivers the time to supervise loading and unloading operations and they could move to other customers in the meanwhile (Cheung et al., 2008). The profitability of this alternative policy depends on the availability of inland depots close to the customers, but inland depots are not often financially feasible for small carriers.

In this case-study, the container loads of export customers are typically not ready before the afternoon, thus the carrier serves import customers before exporters. Moreover, the containers emptied at importers can be filled at export customers, hence a potential routing cost saving can be obtained.

Since the number of containers loads to be delivered to importers and picked from exporters is possibly different, trucks may be required to leave and enter the port carrying some empty containers. More precisely, if the number of container loads to be delivered is larger than the number of container loads to be picked up, trucks return empty containers back to the port. Otherwise, trucks leave the port carrying empty containers to accommodate the requests of all export customers.

Importers and exporters often demand a number of container loads larger than the truck's capacity. Hence, splitting customer demand may be compulsory and each customer may be visited more than once. Moreover, customer demands can be split among several trucks, even if the demand is lower than the capacity. The objective is to determine a set of routes in which routing costs are minimized, all customers are serviced, importers are visited before exporters, and the capacities of trucks are never exceeded.

According to the problem classification in Parragh et al., 2008, this problem belongs to the class of Vehicle Routing Problems with Clustered Backhauls (VRPCB), because in each route all deliveries must be performed before all pickups. However, in classical VRPCB, each customer must be visited only once, whereas in this problem multiple visits at each customer are allowed. Our problem also belongs to the class of the so-called *one-to-many-to-one* pickup and delivery problems, because all delivery demands are initially located at the port and all pickup demands are destined to the same port (Berbeglia et al., 2007).

This problem is called hereafter Split Vehicle Routing Problem with Clustered Backhauls (SVRPCB) and, as far as we are aware, it has not been addressed in its current form in the literature before. In this paper, linehaul customers are referred as import customers, delivery customers or importers. In the same way, backhaul customers are also called export customers, pickup customers or exporters. Similarly, let importer routes and exporter routes be the routes serving only importers or exporters, respectively.

An Integer Linear Programming (ILP) model is presented to address small-sized problems. In order to solve larger instances, we propose a meta-heuristic which exploits existing algorithms for simpler SVRPCB subproblems and guides them toward the construction of good SVRPCB solutions. More precisely, the meta-heuristic constructs a feasible SVRPCB solution by first decomposing the SVRPCB into two Split Vehicle Routing Problems (SVRP), where the first subproblem involves only importers and the second only exporters. These problems are solved by the Tabu Search (TS) of Archetti et al., 2006. Next, importer and exporter routes are paired and merged by solving an assignment problem. This two-stage constructive heuristic is the building block for the proposed meta-heuristic.

However, the importer routes and exporter routes by the TS could not result in good SVRPCB solutions. Therefore, at each iteration of the proposed algorithm, critical properties of the current SVRPCB solution are detected. Some guidance mechanisms are implemented by perturbing the data of the two SVRP, in order to discourage the TS in creating routes having undesired characteristics.

This paper not only proposes a meta-heuristic algorithm for the SVRPCB, but also aims at investigating the effect of the growth in transportation capacities on the carrier's service. The possibility of employing trucks with larger capacities than a single container is considered. This allows the carrier to estimate the savings in adopting larger vehicles.

The rest of the paper is organized as follows. In Section 2, we review the related literature and in Section 3 we present the ILP formulation. In Section 4, the meta-heuristic based on Adaptive Guidance mechanisms is proposed. In Section 5, the results of our extensive computational experience are presented and a comparison between the performances of a state-of-art solver and the meta-heuristic algorithms is reported. Finally, conclusions and further research directions are summarized in Section 6.

## 2. Literature Review

Several papers address the VRPCB, where all linehauls are visited before backhauls and each customer must be visited exactly once. Exact methods for the VRPCB are proposed by Mingozzi et al., 1999 and Toth and Vigo, 1997. Heuristics have been developed by Anily, 1996, Goetschalckx and Jacobs-Blecha, 1989, Toth and Vigo, 1999, Osman and Wassan, 2002, Brandão, 2006, Ropke and Pisinger, 2006 and Zachariadis and Kiranoudis, 2012. Recently, the unified hybrid genetic search algorithm of Vidal et al., 2012 provided the most competitive results for the VRPCB. We refer to the surveys of Gribkovskaia and Laporte, 2008 and Toth and Vigo, 2002 for the single-vehicle and multiple-vehicle problems, respectively.

What makes the SVRPCB different from the VRPCB is the possibility to serve customers more than once. A recent review on SVRP was presented by Archetti and Speranza, 2012.

Some attributes of the SVRPCB can be found in Mitra, 2005 and Mitra, 2008. These papers consider a homogeneous fleet of vehicles located at a depot to serve delivery and pickup demands of a set of customers. Although splitting is allowed, unlike in the SVRPCB, importers and exporters can be visited in any order. Mitra, 2005 developed a Mixed Integer Linear Programming (MILP) formulation for the problem and presented a route construction heuristic, which improved the best known solutions obtained by the MILP formulation. Mitra, 2008 further investigated this problem designing a parallel clustering technique and route construction heuristic.

In the field of intermodal freight transportation, the distribution of containers by trucks between customers and intermodal terminals is known as “drayage”. According to Macharis and Bontekoning, 2004, drayage involves the distribution of a full container from an intermodal terminal to a receiver and the subsequent collection of an empty container, or the provision of an empty container to a shipper for the subsequent transportation of a full container. This definition accounts for both policies where trucks and containers are separated or coupled, as in the SVRPCB.

The separation of trucks and containers has been investigated by Jula et al., 2005, Chung et al., 2007, Zhang et al., 2011, Zhang et al., 2010, Vidovic et al., 2011, Braekers et al., 2013 and Nossack and Pesch, 2013. The variant where trucks and containers are coupled received less attention, in fact it has been investigated only in papers motivated by specific technical restrictions (i.e., Imai et al., 2007) or regulation policies (Cheung et al., 2008).

From a methodological point of view, the latter variant was investigated by Imai et al., 2007, who formulated their problem as the optimal assignment of trucks to a set of delivery and pickup pairs. They developed a subgradient heuristic based on Lagrangian Relaxation. However, trucks cannot visit more than one importer or one exporter in a single trip, because they can carry one container only. Caris and Janssens, 2009 modeled the container drayage problem as a full truckload pickup and delivery problem with time windows. They constructed an initial solution by a two-phase insertion heuristic and improved it using a local search heuristic based on three neighborhoods. Yet, in their problem setting, each truck carries one container only. Lai et al., 2013 investigated how to deliver and collect container loads by trucks carrying one or two containers. A feasible solution was built using an adaptation of the Clarke and Wright, 1964 algorithm and it was improved using two neighborhoods. Hence, this algorithm cannot be used for trucks carrying more than two containers.

To conclude, a frequent characteristic of papers on drayage is the assumption that trucks carry at most one container (Jula et al., 2005, Namboothiri and Erera, 2008, Zhang et al., 2011, Zhang et al., 2010 and Sterzik and Kopfer, 2013). However, if the weight of the containers is under a set value, the capacity of trucks could be higher than one container. Carrying two or more containers per truck is allowed in many countries (Nagl, 2007). Since larger capacities can increase the efficiency of the distribution, this paper investigates this opportunity and aims at quantifying its benefits. However, it is important to note that this opportunity substantially increases the difficulty of SVRPCB, because the underlying packing problem becomes more difficult to solve.

## 3. Formulation

This section introduces the notation and presents an ILP model for the SVRPCB. Let $p$ be the port, $I$ the set of importers, $E$ the set of exporters and $K$ the set of trucks, each with capacity $Q$-containers. Let $d_i$ be the number of containers used to serve customer $i \in I \cup E$. If $i \in I$, $d_i$ represents the number of containers used to deliver container loads to import customer $i \in I$. If $i \in E$, $d_i$ represents the number of containers used to pick up container loads from export customer $i \in I$.

Given a direct graph $G = (N, A)$, the set $N$ is defined as $N = \{p \cup I \cup E\}$. Since trucks are not allowed to move from exporters to importers, the set $A$ of arcs is defined as $A = A_1 \cup A_2$, where $A_1 = \{(i,j) | i \in p \cup I, j \in N, i \neq j\}$ $A_2 = \{(i,j) | i \in E, j \in p \cup E, i \neq j\}$. Three sets of variables are defined:

**$x_{ij}^k$:** Routing selection variables taking value 1 if arc $(i,j) \in A$ is traversed by truck $k \in K$, 0 otherwise; let $c_{ij} \geq 0$ be the cost of traversing arc $(i,j)$;

**$y_{ij}^k$:** Number of loaded containers carried along arc $(i,j) \in A$ by truck $k \in K$;

**$z_{ij}^k$:** Number of empty containers carried along arc $(i,j) \in A$ by truck $k \in K$.

The problem can be formulated as follows:

$$min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} \ x_{ij}^k \tag{1}$$

$s.t.$

$$\sum_{k \in K} \sum_{l \in N} y_{il}^k = \sum_{k \in K} \sum_{j \in p \cup I} y_{ji}^k - d_i \qquad \forall i \in I \tag{2}$$

$$\sum_{k \in K} \sum_{l \in N} z_{il}^k = \sum_{k \in K} \sum_{j \in p \cup I} z_{ji}^k + d_i \qquad \forall i \in I \tag{3}$$

$$\sum_{l \in N} y_{il}^k \leq \sum_{j \in p \cup I} y_{ji}^k \qquad \forall i \in I, \forall k \in K \tag{4}$$

$$\sum_{l \in N} z_{il}^k \geq \sum_{j \in p \cup I} z_{ji}^k \qquad \forall i \in I, \forall k \in K \tag{5}$$

$$\sum_{k \in K} \sum_{l \in p \cup E} y_{il}^k = \sum_{k \in K} \sum_{j \in N} y_{ji}^k + d_i \qquad \forall i \in E \tag{6}$$

$$\sum_{k \in K} \sum_{l \in p \cup E} z_{il}^k = \sum_{k \in K} \sum_{j \in N} z_{ji}^k - d_i \qquad \forall i \in E \tag{7}$$

$$\sum_{l \in p \cup E} y_{il}^k \geq \sum_{j \in N} y_{ji}^k \qquad \forall i \in E, \forall k \in K \tag{8}$$

$$\sum_{l \in p \cup E} z_{il}^k \leq \sum_{j \in N} z_{ji}^k \qquad \forall i \in E, \forall k \in K \tag{9}$$

$$\sum_{(ji)\in A}(y_{ji}^k+z_{ji}^k)=\sum_{(il)\in A}(y_{il}^k+z_{il}^k) \qquad \forall i\in I\cup E, \forall k\in K \tag{10}$$

$$y_{ij}^k+z_{ij}^k\le Q\ x_{ij}^k \qquad \forall (i,j)\in A, \forall k\in K \tag{11}$$

$$\sum_{j\in N}x_{ji}^k-\sum_{l\in N}x_{il}^k=0 \qquad \forall i\in N, \forall k\in K \tag{12}$$

$$\sum_{j\in N}x_{ij}^k\le 1 \qquad \forall i\in N, \forall k\in K \tag{13}$$

$$\sum_{k\in K}\sum_{i\in I\cup E}z_{ip}^k-\sum_{k\in K}\sum_{i\in I\cup E}z_{pi}^k=\sum_{i\in I}d_i-\sum_{i\in E}d_i \tag{14}$$

$$x_{ij}^k\in\{0,1\} \qquad \forall (i,j)\in A, \forall k\in K \tag{15}$$

$$y_{ij}^k\in\{0,1,\ldots,Q\} \qquad \forall (i,j)\in A, \forall k\in K \tag{16}$$

$$z_{ij}^k\in\{0,1,\ldots,Q\} \qquad \forall (i,j)\in A, \forall k\in K \tag{17}$$

Routing costs are minimized in the objective function (1).

Constraints (2)-(5) concern the distribution of containers to importers. Constraints (2) and (3) are the flow conservation constraints of loaded and empty containers, respectively, at each importer node. Constraints (4) enforce that the number of loaded containers cannot increase after servicing any importer, whereas constraints (5) guarantee that the number of empty containers does not decrease.

Similarly, constraints (6)-(9) concern the distribution of containers to exporters. Constraints (6) and (7) are the flow conservation constraints of loaded and empty containers, respectively, for each exporter. Constraints (8) and (9) enforce that the number of loaded containers cannot decrease after visiting an exporter, whereas the number of empty containers cannot increase.

Constraints (10) guarantee that the number of containers carried by each truck does not change after visiting a customer. Constraints (11) impose that the number of containers on each truck does not exceed the capacity $Q$.

Constraints (12) represent the flow conservation constraints for each truck at each node. Constraints (13) enforces that each truck can reach only one node from the current node. It is important to note that constraints (12) and (13) enforce that the degree of each node must be at most 2. This forces a vehicle to visit the same customer at most once in a route. Moreover, if there is a successor for a node $i$ visited in the route of truck $k$, Constraints (12) impose that there is also a predecessor for the same node and the same truck. Constraints (13) also guarantee that trucks are not used more than once.

Constraints (14) represent the flow conservation of empty containers at the port $p$. Finally, Constraints (15), (16) and (17) define the domain of the decision variables.

The model has been implemented using IBM ILOG CPLEX Optimization Studio 12.5 and solved by ILOG CPLEX 12.2 solver. Since exact methods may not be able to solve realistic-size instances of SVRPCB with high truck capacity, we present a meta-heuristic, which is described in the following section.

## 4. Meta-heuristic algorithm

The proposed meta-heuristic is based on *Adaptive Guidance* (AG) mechanisms, which are simple rules applied to check the quality of the current solution and detect possibly improvements. Then, the input parameters of simpler subproblems are perturbed so as to achieve the desired diversification in the complex problem at hand. Examples of successful implementations of adaptive guidance algorithms are presented in Battarra et al., 2009, Bai et al., 2007, Kramer, 2008 and Olivera and Viera, 2007. Moreover, Hart, 2005 presented a large class of simple rules of behavior, called adaptive heuristics.

Our overall meta-heuristic consists of three phases:

*(i)* SVRP phase decomposes the SVRPCB into two SVRPs, one for importers and one for exporters, each solved by the TS (Glover and Laguna, 1998) proposed by Archetti et al., 2006.

*(ii)* Merging phase merges importer routes and exporter routes determined in SVRP phase by an ILP model based on the saving concept;

*(iii)* AG phase analyses the current solution, detects areas of improvement and adjusts the input parameters of the SVRP phase.

The three phases are repeated sequentially until a stop criterion is satisfied and the best solution found is returned.

Table 1 illustrates the pseudo-code of the meta-heuristic algorithm, in which the following notation is adopted:

**tExe** Execution time;

**it** Number of consecutive iterations performed during the whole execution;

**notImpIt** Number of consecutive iterations performed since an improving solution was found;

**S**$^*$ Best solution found;

**MAXTIME** Maximum execution time;

**MAXIT** Maximum number of consecutive iterations allowed during the whole execution;

**SolImp** Set of importer routes determined in the SVRP phase by the TS solving the SVRP on the set $I$ of importers.

**SolExp** Set of exporter routes determined in the SVRP phase by the TS solving the SVRP on the set $E$ of exporters.

**Sol** Current solution of the meta-heuristic;

**SMatrix** Matrix of all savings that can be obtained by merging importer routes and exporter routes;

**Merge**($Sol$, $SolImp$, $SolExp$, $SMatrix$) Function merging routes determined in the SVRP phase by an ILP model. The input parameters are the current solution $Sol$, the set of importer routes $SolImp$ and exporter routes $SolExp$, and the saving matrix $SMatrix$. The output is the new current solution $Sol$;

**AdaptiveGuidance**($Sol$, $SolImp$, $SolExp$, $it$) Function analyzing the current solution $Sol$ according to different criteria (or guidance mechanisms) and perturbing the costs in the SVRP phase. Since it is not compulsory to perform all mechanisms at each iteration, this function depends on the current number of iterations $it$.

```
procedure MAIN
    Start tExe
    it = 0
    notImpIt = 0
    S* ← ∅
    while tExe ≤ MAXTIME & notImpIt ≤ MAXIT do
        it = it + 1
        notImpIt = notImpIt + 1
        SolImp ← TS(I);                                   ▷ SVRP phase Section 4.1
        SolExp ← TS(E);
        Create the savings matrix SMatrix
        Sol ← ∅
        Sol ← MERGE(Sol, SolImp, SolExp, SMatrix)         ▷ Merging phase
Section 4.2
        if Sol ≤ S* || S* == ∅ then
            S* ← ∅
            S* ← Sol
            notImpIt ← 0
        end if
        ADAPTIVEGUIDANCE(Sol, SolImp, SolExp, it)         ▷ AG phase Section 4.3
    end while
    return S*
end procedure
```

Table 1: The structure of the meta-heuristic.

In the following, the three phases of the algorithm are described in detail.

### *4.1. SplitVRP phase*

The SVRP phase consists of solving two SVRPs: the first involves importers only, whilst the second exporters only. As stated previously, the TS by Archetti et al., 2006 is employed to solve this NP-hard problem. The algorithm consists of three phases: *(i)* the first phase determines the initial feasible solution constructing a giant tour by the GENIUS algorithm (Gendreau et al., 1992) and imposing trucks to return to the depot whenever their load equals the capacity; *(ii)* the second phase consists of a TS based on relocation moves, where a customer is either relocated into another route or copied into an alternative route. In the latter case, its original demand is split between the two routes; *(iii)* the third phase improves the solution found by removing t-split cycles and by re-optimizing each route using the GENIUS algorithm.

### *4.2. Merging phase*

Routes determined in the SVRP phase are merged in the Merging phase according to an ILP model, which is inspired by the Clarke and Wright savings algorithm (Clarke and Wright, 1964). In this algorithm savings are obtained by merging a route servicing importers with a route servicing exporters, instead of leaving them separate. It is important to note that, four possible routes can be generated by merging a selected pair of routes, because the first and the last importer may be linked to the first or the last exporter. To clarify, consider for instance $n$ importers, serviced by route $r_i = \{p,\ i_1, \ldots,\ i_n,\ p\}$, and $m$ exporters serviced by route $r_j = \{p,\ e_1, \ldots,\ e_m,\ p\}$. Moreover, let $c(i_n, e_1)$ be the cost of arc $(i_n, e_1) \in A$, and so on. When the merging of routes $r_i$ and $r_j$ is evaluated, the algorithm computes four different savings based on the extra mileage evaluation:

- $s^1_{ij} = c_{(i_n,\ p)} + c_{(p,\ e_1)} - c_{(i_n,\ e_1)}$, where routes $r_i$ and $r_j$ keep their original direction in the final route, i.e. importers are visited from $i_1$ to $i_n$ and exporters from $e_1$ to $e_m$;

- $s^2_{ij} = c_{(i_n,\ p)} + c_{(p,\ e_1)} + c_{(e_m,\ p)} - c_{(i_n,\ e_m)} - c_{(e_1,\ p)}$, where in the final route $r_i$ has the original direction and $r_j$ the opposite one, i.e. importers are visited from $i_1$ to $i_n$ and exporters from $e_m$ to $e_1$;

- $s^3_{ij} = c_{(p,\ i_1)} + c_{(i_n,\ p)} + c_{(p,\ e_1)} - c_{(p,\ i_n)} - c_{(i_1,\ e_1)}$, where in the final route $r_i$ has the opposite direction and $r_j$ the original one, i.e. importers are visited from $i_n$ to $i_1$ and exporters from $e_1$ to $e_m$;

- $s^4_{ij} = c_{(p,\ i_1)} + c_{(i_n,\ p)} + c_{(p,\ e_1)} + c_{(e_m,\ p)} - c_{(p,\ i_n)} - c_{(i_1,\ e_m)} - c_{(e_1,\ p)}$, where routes $r_i$ and $r_j$ have the opposite direction in the final route, i.e. importers are visited from $i_n$ to $i_1$ and exporters from $e_m$ to $e_1$;.

Each pair of routes is supposed to be merged according to the maximum saving. Therefore, the saving generated by merging routes $r_i$ and $r_j$ is $s_{ij} = max\{s^1_{ij}, s^2_{ij}, s^3_{ij}, s^4_{ij}\}$. Maximum savings are recorded in a matrix, in which the number of rows is equal to $|SolImp|$ and the number of columns is equal to $|SolExp|$.

Routes determined in SVRP phase are merged in the Merging phase by the following assignment problem. For all $i \in SolImp$ and $j \in SolExp$, let $w_{ij}$ be a binary variable, which takes value 1 if routes $r_i$ and $r_j$ are merged, 0 otherwise. The assignment problem can be formulated by the following ILP model:

$$max \sum_{i \in SolImp} \sum_{j \in SolExp} s_{ij}\ w_{ij} \tag{18}$$

$$s.t.$$

$$\sum_{j \in SolExp} w_{ij} \leq 1 \qquad \forall i \in SolImp \tag{19}$$

$$\sum_{i \in SolImp} w_{ij} \leq 1 \qquad \forall j \in SolExp \tag{20}$$

$$w_{ij} \in \{0, 1\} \qquad \forall i \in SolImp, j \in SolExp \tag{21}$$

The overall gain is maximized in the objective function (18), where $s_{ij}$ represents the maximum saving obtained by merging routes $i$ and $j$, as described above.

Constraints (19) and (20) enforce that each route in $SolImp$ can be merged at most with a route in $SolExp$ and vice-versa. We do not consider merging operations nvolving more than an importer route and an exporter route, because it is quite unlikely that feasible SVRPCB solutions would be obtained due to the violation of the capacity constraint.

### *4.3. Adaptive guidance phase*

The AG phase analyses the incumbent SVRPCB solution according to predefined criteria, each of which gives rise to a guidance mechanism. If drawbacks are detected in the solution, the input data of the TS are suitably perturbed by guidance mechanisms, which are implemented by penalizing costs in the SVRP phase. In this section we illustrate how to identify drawbacks in the incumbent solution, define quantitative measures for their evaluation and design suitable penalization mechanisms that would result in the desired diversification effect, without corrupting the original SVRPCB input data.

Our meta-heuristic is guided by the following guidance mechanisms:

(i) $A.G.M.1$ - *Avoiding too many Splits*

Since the TS tends to generate routes where load splitting is allowed, the resulting SVRPCB solutions may be likely poor when the number of visits to customers is unnecessarily high. This guidance mechanism is aimed at correcting this drawback. Given a customer $i$, let $minTrip_i = \lceil d_i/Q \rceil$ be the minimum number of visits required to satisfy its demand, let $visit_i$ be the number of visits to customer $i$ in the current solution and let $exceed_i$ be the difference between $visit_i$ and $minTrip_i$. This guidance mechanism selects the importer and the exporter with the largest positive values of $exceed_i$, if any. A penalization is introduced for all arcs entering and leaving these customers in the next $\gamma$ iterations, in order to guide the TS toward a lower use of arcs connecting these customers and, hence, split the load into a lower number of routes.

(ii) $A.G.M.2$ - *Promising extreme importers*

The first importer and the last one play a crucial role in the SVRPCB, in fact, if they are close to export customers, the Merging phase is far more effective in connecting the importer route to an exporter route. However, the set $SolImp$ of import routes determined in the SVRP phase ignores the location of exporters and, hence, the resulting SVRPCB solutions may be likely poor. This guidance mechanism aims to guide the TS, so that importers with close exporters are forced to be the first ones or the last ones in the new solution of the SVRP phase. In what follows, we refer to extreme importers instead of the first and the last importer in a route.

Given a importer $i \in I$, we denote with $\alpha_i$ the number of times in which $i \in I$ is visited as an extreme node in the incumbent SVRPCB solution minus the number of times in which $i \in I$ is visited as an internal node. In order to diversify the current solution, we are interested in the negative values of $\alpha_i$, because they indicate customers which are frequently visited as internal nodes. Moreover, let $\sigma_i$ be the sum of all distances between the selected importer $i \in I$ and all exporters. Since low values of $\sigma_i$ indicate the high proximity of many exporters to the selected importer, this guidance mechanism selects the importer $i \in I$ having a negative value of $\alpha_i$ and the minimum value of $\sigma_i$, if any. In order to remove customer $i$ from its frequent position of internal node, a penalization is added in the SVRP phase to all arcs entering or leaving importer $i$ for the next $\gamma$ iterations.

(iii) $A.G.M.3$ - *Promising extreme exporters*

This mechanism works as $A.G.M.2$, but it considers extreme exporters instead of importers.

(iv) $A.G.M.4$ - *Avoiding expensive arcs*

This mechanism aims to avoid the use of the most costly arcs in the incumbent SVRPCB solution. This guidance mechanism selects the most expensive arcs connecting pairs of importers and pairs of exporters and penalize them in the SVRP phase for the next $\gamma$ iterations.

*Remarks*

In the proposed meta-heuristic the execution of a single guidance mechanism is iteration-dependent. As a result, at any iteration one may run a guidance mechanism with all other mechanisms, with some of them or one at a time. Hence, it is important to properly calibrate parameters controlling when each guidance mechanism should be performed during the overall execution of the algorithm.

### 4.4. Penalizations

Once the incumbent SVRPCB solution has been analysed according to a guidance mechanism, the selected arc costs are penalized in the SVRP phase for the subsequent $\gamma$ iterations. If arc $(i, j)$ connects two customers, its cost is penalized as

$$c_{ij} = c_{ij} + RandomCoef \ \cdot \ M \tag{22}$$

The value $M$ is set up as the largest entry of the cost matrix and $RandomCoef$ is a coefficient that randomly decreases/increases the penalties during the overall execution of the algorithm, according to the formula:

$$RandomCoef = (Random(1, \ldots, \alpha) + \beta)/100 \tag{23}$$

where $\beta$ is a self-adapting parameter taking initial value 0 and increasing by $\alpha$ after each $\alpha$ iterations. Whenever a better SVRPCB solution is found, $\beta$ is set to 0, in order to refresh penalties.

A larger penalty is added to the cost of arcs connecting customers to the port, in order to minimize the number of trucks in the solution. More formally, if arc $(i, j)$ connects a customer to the port, its cost is penalized as

$$c_{ij} = c_{ij} + RandomCoef \ \cdot \ M + (|N| - 1) \ \cdot \ M \tag{24}$$

where $M$ is the largest entry of the cost matrix and $N$ the set of nodes. Moreover, whenever an improving solution is found, penalties are set to zero for arcs linking the port to the set of importers or exporters serviced by a lower number of routes. This allows for a lower number of routes and, hence, lower routing costs.

Three different methods are proposed for the introduction of penalties. The three methods are:

***(i)*** **Unchecked penalties** Penalties are added to an arc cost, even if a penalization is already applied. To clarify, if a penalty on arc $(i, \ j) \in A$ is added from iteration $it$ to $it + \gamma$ and the arc is selected to be penalized at iteration $it + \delta$ (with $\delta \leq \gamma$), the penalty is applied twice;

***(ii)*** **Unique penalties** Penalties are applied in the next $\gamma$ iterations on an arc only if it is not penalized at the moment. To clarify, if a penalty on arc $(i, \ j) \in A$ is applied from iteration $it$ to $it + \gamma$ and the arc is selected to be penalized at iteration $it + \delta$ (with $\delta \leq \gamma$), the penalty is rejected and the adaptive guidance mechanism is executed again, until an arc not yet penalized is detected or no more penalties become available;

***(iii)*** **Incremental unique penalties** It implements both previous penalization strategies. To clarify, if a penalty is applied on arc $(i, \ j) \in A$ from iteration $it$ to $it + \gamma$ and the arc is selected to be penalized at iteration $it + \delta$ (with $\delta \leq \gamma$), the penalty is accepted and the adaptive guidance mechanism is executed again, until an arc not yet penalized is detected or no more penalties become available. Therefore, if a penalty on an arc $(i, \ j) \in A$ is found at iterations $it$ and $it + \delta$ (with $\delta \leq \gamma$), the penalty is inserted twice and the adaptive guidance mechanism is executed again to look for additional penalties.

## 5. Computational results

This section aims to analyze the performance of the proposed meta-heuristic. Our test set consists of 140 uniformly generated instances with 10, 20, 30, 40, and 50 customers. Since large-sized instances are the most challenging, we generate a larger number of instances of large size (12 instances with 10 customers, 20 with 20 customers, 28 with 30 customers, 36 with 40 customers and 44 with 50 customers).

Instances with the same number of customers have the same customer locations, which are integers uniformly generated between $-1000$ and $+1000$ and the same demands, which are integers generated according to a random uniform distribution in the range 1 to 5.

The ratio between the number of importers and exporters is generated as follows. Denoting by $n$ the number of customers, we generate $n/5+1$ instances. The number of importers in instance $k \neq \{0, n/5\}$ is $5k$, consequently the number of exporters is $n-5k$. However, in order to have at least two importers and two exporters in the instance $k = 0$ and $k = n/5$, the number of import and export customers for such instances is forced to be 2, respectively.

The number of trucks in each instance is fixed and is equal to the minimum number of trucks needed to service all container loads. It is computed as the the bin packing lower bound $\lceil max\{\sum_{i\in I} d_i, \sum_{i\in E} d_i\}/Q\rceil$.

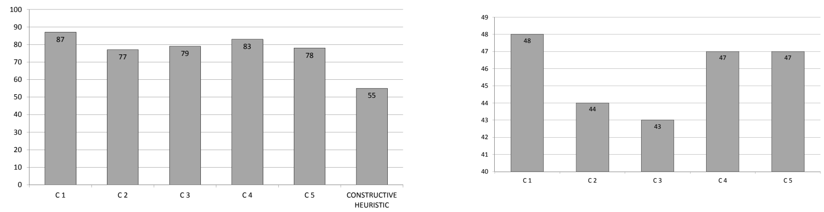
Twenty percent of the instances for each problem-size considers trucks carrying up to 1-container, 2-containers, 4-containers and 6-containers. This choice depends on the current rules adopted in several countries. Whenever the overall load weight is not a constraint, almost all countries allow to carry up to 2 containers, some others up to 4 containers. To our knowledge, in Australia up to 3 40 feet containers per truck are allowed when rail transportation is not available (Nagl, 2007). Nevertheless, these instances allow for experimenting with transportation units smaller than containers. The instances are available upon request.

### 5.1. *Experimental Setting*

The integer programming formulation (1)-(17) had been coded using IBM ILOG CPLEX Optimization Studio 12.5 and solved by the Branch & Cut of ILOG CPLEX 12.2. The meta-heuristic presented in Section 4 was coded in C++, and the integer model (18)-(21) is solved using the Callable Libraries of CPLEX 12.2. Experiments have been performed on a Linux four-CPU server 2.67 GHz 64 GB RAM, with default parameter settings.

Although a major requirement for the carrier is to determine solutions in about 10 minutes, the solver execution has also been set to stop after 3 hours. This choice allows the solver to produce better upper and lower bounds and provide a better term of comparison for assessing the quality of the meta-heuristic.

We set $MAXTIME$ to 600 seconds, as suggested by the carrier, $MAXIT = 10000$, $\gamma = |I|$ for penalties involving importers and $\gamma = |E|$ for penalties involving exporters. Finally, the coefficient $\alpha$ in Equation (23) takes value 10. These settings proved to provide good quality results in our preliminary testing.




(a) *Number of best known solutions returned by the each calibration and the constructive heuristic.*

(b) *Improvements with respect to the constructive heuristic.*

The meta-heuristic depends also on the parameter $\varphi$, which sets the strategy to update penalties: it takes value 1 for "Unchecked penalties", 2 for "Unique penalties" and 3 for "Incremental unique penalties" (see Section 4.4). All penalization strategies are tested and combined with several execution sequences of the guidance mechanisms. The top five calibrations in our preliminary experiments are denoted by $C_1$, $C_2$, $C_3$, $C_4$ and $C_5$, and are described hereafter:

- $C_1$ each adaptive guidance mechanism has probability 33% to be performed at each iteration. Penalties are updated according to the strategy "Unchecked penalties";
- $C_2$ all adaptive guidance mechanisms are performed at each iteration. Penalties are updated according to the strategy "Unchecked penalties";
- $C_3$ the $AGM1$ is the only adaptive guidance mechanism and it is performed at each iteration. Penalties are updated according to the strategy "Unchecked penalties";
- $C_4$ each adaptive guidance mechanism has probability 33% to be performed at each iteration. Penalties are updated according to the strategy "Incremental unique penalties";
- $C_5$ each adaptive guidance mechanism has probability 25% to be performed at each iteration. Penalties are updated according to the strategy "Incremental unique penalties".

In order to select the best calibration among them, all generated instances are solved with each setting of the meta-heuristic. Since 33 instances out of 140 are proven to be optimal by Cplex, we consider the remaining 107 instances and we compute how many times the best solution is found by each setting of the meta-heuristic and by the constructive heuristic. Results are represented in Figure 1a.

Figure 1a shows that calibration $C_1$ seems to be the most effective, in fact it determines the best solution for 87 times out of 107 instances. Figure 1a also shows that in 55 instances the constructive heuristic returns the best solution and no improvement is obtained by any proposed guidance mechanism.

Figure 1b shows how many times each setting of the metaheuristic improves the solution of the SVRPCB determined by the constructive heuristic. For example, calibration $C_1$ improves the initial feasible solution of the SVRPCB for 48 times and calibration $C_2$ for 44 times.

As Figures 1a and 1b show, $C_1$ seems to be the most promising calibration. Therefore, the results obtained by this calibration are discussed hereafter.

### 5.2. Effectiveness of the adaptive guidance mechanisms

This section illustrates the improvements produced by the adaptive guidance mechanisms running under the calibration $C_1$ with respect to the constructive heuristic solution.

In Table 2 each row describes a class of several instances. $Q$ denotes the transportation capacity of the homogeneous fleet of trucks and Instances the number of instances in the considered class. Table 2 reports in the column $CONSTRUCTIVE \quad HEURISTIC$ Average t(s), which is the average time to determine the first feasible solution by the constructive heuristic. The column $ADAPTIVE \quad GUIDANCE$ indicates the average time in seconds to find the best feasible solution for the meta-heuristic (Average t(s)) and the average gap between the solution of the meta-heuristic and the solution of the constructive heuristic (Average % Gap). Negative values of this gap indicate the average improvement produced by guidance mechanisms on the class of instances considered in that row.

Table 2 shows that interesting improvement opportunities can be obtained by the guidance mechanisms. Moreover, they seem to be more effective as the truck capacity increases.

### 5.3. Comparison with Cplex

This section compares solutions provided by the meta-heuristic with those obtained by state-of-art solver *CPLEX*. Computational results are reported in Table 3 following the additional notation:

- *Average % Gap 10 min*: Average percentage gaps with respect to best solutions provided by CPLEX in 10 minutes. When the solutions of the meta-heuristic are better than the best CPLEX upper bounds, or the meta-heuristic provides the optimal solutions, gaps are reported in bold.
- *Average % Gap 3 h*: Average percentage gaps with respect to best solutions provided by CPLEX in 3 hours. When solutions of the meta-heuristic are better than CPLEX upper bounds, or the meta-heuristic provides optimal solutions, gaps are reported in bold.
- *n.a.*: No available gap with respect to CPLEX within its time limit, because CPLEX did not find any feasible solution.

*CPLEX 10 min and 3h*

| Q | Instances | CONSTRUCTIVE HEURISTIC Average t(s) | ADAPTIVE GUIDANCE Avergage t(s) | ADAPTIVE GUIDANCE Average % Gap |
|---|---|---|---|---|
| | | 10 CUSTOMERS | | |
| 1 | 3 | 0.23 | 0.23 | 0.00 |
| 2 | 3 | 0.18 | 0.18 | 0.00 |
| 4 | 3 | 5.19 | 5.19 | 0.00 |
| 6 | 3 | 4.22 | 74.04 | 0.00 |
| | | 20 CUSTOMERS | | |
| 1 | 5 | 1.97 | 1.97 | 0.00 |
| 2 | 5 | 1.11 | 1.11 | 0.00 |
| 4 | 5 | 6.96 | 173.79 | **-0.57** |
| 6 | 5 | 8.70 | 96.09 | **-1.70** |
| | | 30 CUSTOMERS | | |
| 1 | 7 | 7.57 | 7.57 | 0.00 |
| 2 | 7 | 8.63 | 41.68 | **-0.36** |
| 4 | 7 | 13.21 | 408.39 | **-1.98** |
| 6 | 7 | 13.77 | 73.67 | **-1.19** |
| | | 40 CUSTOMERS | | |
| 1 | 9 | 23.50 | 23.50 | 0.00 |
| 2 | 9 | 23.33 | 188.15 | **-0.31** |
| 4 | 9 | 12.39 | 195.06 | **-1.02** |
| 6 | 9 | 17.26 | 228.82 | **-1.92** |
| | | 50 CUSTOMERS | | |
| 1 | 11 | 31.69 | 31.69 | 0.00 |
| 2 | 11 | 23.28 | 48.19 | **-0.04** |
| 4 | 11 | 16.31 | 131.61 | **-0.28** |
| 6 | 11 | 19.38 | 198.04 | **-0.26** |

Table 2: Adaptive guidance effectiveness

- *Optimality / Feasibility*: The first number indicates the number of optimal solutions obtained in that class of instances; the second number indicates the number of feasible solutions for which the optimality cannot be demonstrated;
- *Average Opt. Gap*: The optimality gap between upper and lower bounds determined by CPLEX in 10 minutes and 3h, respectively;
- *n.s.*: No feasible solution determined by CPLEX within the time limit.

It is important to note that each row of Table 3 represents average percentage gaps over a class of instances.

Table 3 shows that the meta-heuristic provides exact solutions in instances where the transportation capacity is 1 container. CPLEX outperforms the meta-heuristic in few small instances; when $n = 10$ and $Q = 6$, there are exact solutions at most 1.62% better than those determined by the meta-heuristic.

In the instances with 20 customers, CPLEX outperforms the meta-heuristic only when it is executed for 3h: the gaps are 0.12% and 0.41% for $Q$ equal 2 and 6, respectively. Nevertheless, the solutions obtained by CPLEX in 10 minutes are up to 12.19% worse on average than those of the meta-heuristic.

In case of instances with 30 customers, the meta-heuristic outperforms systematically CPLEX, both when it is executed for 10min and 3h. The average

| | | ADAPTIVE GUIDANCE | | | CPLEX 10 min | | CPLEX 3 h | |
|---|---|---|---|---|---|---|---|---|
| Q | Instances | Average t(s) | Average % Gap 10 min | Average % Gap 3 h | Optimality / Feasibility | Average %Opt. Gap | Optimality / Feasibility | Average % Opt. Gap |
| | | | | 10 CUSTOMERS | | | | |
| 1 | 3 | 0.23 | 0.00 | 0.00 | 3 / 0 | 0.00 | 3 / 0 | 0.00 |
| 2 | 3 | 0.18 | 0.00 | 0.00 | 1 / 2 | 3.18 | 1 / 2 | 2.33 |
| 4 | 3 | 5.19 | 0.00 | 0.00 | 1 / 2 | 1.45 | 3 / 0 | 0.00 |
| 6 | 3 | 74.04 | 1.62 | 1.62 | 3 / 0 | 0.00 | 3 / 0 | 0.00 |
| | | | | 20 CUSTOMERS | | | | |
| 1 | 5 | 1.97 | 0.00 | 0.00 | 5 / 0 | 0.00 | 5 / 0 | 0.00 |
| 2 | 5 | 1.11 | **-0.49** | 0.12 | 0 / 5 | 5.20 | 0 / 5 | 4.46 |
| 4 | 5 | 173.79 | **-4.73** | **-0.30** | 0 / 5 | 16.27 | 0 / 5 | 10.97 |
| 6 | 5 | 96.09 | **-12.19** | 0.41 | 0 / 5 | 21.22 | 0 / 5 | 7.63 |
| | | | | 30 CUSTOMERS | | | | |
| 1 | 7 | 7.57 | 0.00 | 0.00 | 7 / 0 | 0.00 | 7 / 0 | 0.00 |
| 2 | 7 | 41.68 | **-10.91** | **-1.42** | 0 / 5 | 15.70 | 0 / 7 | 6.14 |
| 4 | 7 | 408.39 | **-29.10** | **-15.28** | 0 / 7 | 37.51 | 0 / 7 | 25.33 |
| 6 | 7 | 73.67 | **-42.78** | **-24.20** | 0 / 5 | 48.86 | 0 / 7 | 33.62 |
| | | | | 40 CUSTOMERS | | | | |
| 1 | 9 | 23.50 | 0.00 | 0.00 | 2 / 0 | 0.00 | 9 / 0 | 0.00 |
| 2 | 9 | 188.15 | n.a. | n.a. | 0 / 0 | n.s. | 0 / 0 | n.s. |
| 4 | 9 | 195.06 | **-31.51** | **-24.82** | 0 / 4 | 36.94 | 0 / 5 | 30.28 |
| 6 | 9 | 228.82 | **-54.78** | **-49.95** | 0 / 1 | 62.71 | 0 / 6 | 57.68 |
| | | | | 50 CUSTOMERS | | | | |
| 1 | 11 | 31.69 | n.a. | 0.00 | 0 / 0 | n.s. | 9 / 0 | 0.00 |
| 2 | 11 | 48.19 | n.a. | n.a. | 0 / 0 | n.s. | 0 / 0 | n.s. |
| 4 | 11 | 131.61 | n.a. | n.a. | 0 / 0 | n.s. | 0 / 0 | n.s. |
| 6 | 11 | 198.04 | n.a. | n.a. | 0 / 0 | n.s. | 0 / 0 | n.s. |

Table 3: Comparison with the exact algorithm.

gaps are up to 42.78% and 24.20% for 10min and 3h, when $Q = 6$. A similar trend of improvement can be observed when $n = 40$, even if there are few instances where CPLEX was capable of generating an upper bound and, thus, the direct comparison of the methods is less significant. When $n = 50$, CPLEX cannot determine any upper bound in 10 minutes and returns only 9 upper bounds out of 44 instances in 3h.

Tests show that the meta-heuristic improves most of the upper bounds produced by the exact algorithm, when the instance size is larger than 20-30 customers. Moreover, CPLEX is not able to provide feasible solutions for 77 out of 140 instances within a time limit of 10 minutes, and 51 out of 140 instances within a time limit of 3 hours. From the point of view of the execution time, the meta-heuristic provides all feasible solutions in less than 10 minutes.

Finally, Figure 1 analyses how larger capacities remarkably decrease the routing cost of the distribution. As Figure 1 shows, whenever the trucks have a larger capacity, the distribution is performed at a lower cost:

- If we consider the instances with capacity $Q = 2$ with respect to the instances with capacity $Q = 1$, the routing cost decreases by 47.05% in the case of 20 customers, up to 58.22% in the case of 10 customers.

- If we consider the instances with capacity $Q = 4$ with respect to the instances with capacity $Q = 2$, the routing cost decreases by 38.72% in the case of 10 customers, up to 46.06% in the case of 40 customers.

- If we consider the instances with capacity $Q = 6$ with respect to the instances with vehicles $Q = 4$, the routing cost decreases by 20.01% in the case of 10 customers, up to 26.94% in the case of 20 customers.

Note that the marginal improvement due to the vehicles with capacity $Q = 6$ with respect to trucks with capacity $Q = 4$ is relatively small, but if we consider the instances with vehicles capacity $Q = 6$, with respect to the instances with vehicles capacity $Q = 1$, the routing cost decrease by 77.99% in the case of 20 customers and up to 79.52% in the case of 10 customers.

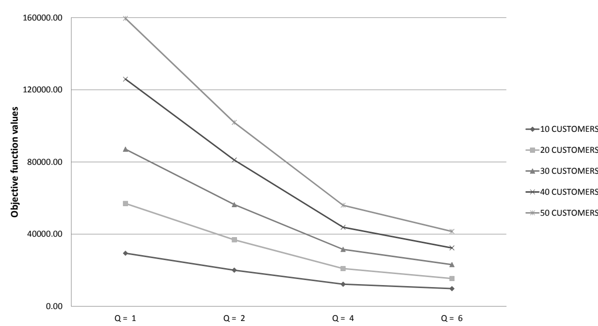



Figure 1: Efficiency of the distribution with larger transportation capacities

## 6. Conclusions

This paper addressed the SVRPCB, which is rich vehicle routing problem originating from a real world application. Although there are many papers on VRPCB and SVRP, to our knowledge, their integration was seldom investigated. In the specific field of container transportation, this is an interesting variant of drayage problems, due to the coupling between containers and trucks, each of which can carry more than one container. In this paper we have presented a mathematical model for the SVRPCB.

The proposed solution method is a meta-heuristic based on adaptive guidance mechanisms. It determines feasible solutions for SVRPCB by a constructive heuristic decomposing the problem into two simpler SVRPs, each solved by a TS, and exactly merging routes by an assignment problem. However, these feasible solutions may be inefficient, since too many splits may be performed, highly expensive arcs may be used and the first or the last importer and/or exporter in any route may not be appropriate.

The proposed meta-heuristic aims to improve these solutions by detecting predefined drawbacks and guiding the TS in the SVRPs, in order to produce the desired diversification in SVRPCB solutions. More precisely, four guidance mechanisms are implemented by perturbing in the subsequent iterations the costs of the SVRPs, in order reduce splits, use less expensive arcs and change the first and/or the last customer in current routes.

Our experimentation indicates that some guidance mechanisms are more effective than others, but usually they are all able to improve initial feasible solutions. In our experimentation the most effective guidance mechanism is obtained when all proposed guidance mechanisms are randomly combined and arcs already perturbed can be penalized further. Moreover, the meta-heuristic is much more effective than a state-of-art solver in solving artificial instances with 20 and 30 customers, yielding considerable savings in terms of travelled distances. Therefore, the meta-heuristic represents a promising instrument to improve the decision-making process and provides a quantitative estimation of savings obtainable by increasing transportation capacities.

To conclude, the adaptive guidance mechanism is a general approach, which is based on the iterative analysis of current solutions and perturbation of simpler subproblems by problem-specific adaptive guidance mechanisms. It is important to note that this approach can exploit existing heuristics for subproblems of the problem at hand. Hence, one may easily adapt modules of code already in use, minimizing the inconvenience of adopting a new software. Easy pieces of code are also easier to maintain and possibly adapt to incorporate more advanced problem features. Further research will be carried out to implement guidance mechanisms on rich vehicle routing problems.

## 7. Acknowledgements

This work was partially supported by Grendi Trasporti Marittimi. The authors are also grateful to Gunes Erdogan for his careful comments on previous versions of the paper and Teodor Gabriel Crainic for the enlightening talks on this topic.

## References:

Anily, S., 1996. The vehicle-routing problem with delivery and back-haul options. Naval Research Logistics 43 (3), 415–434.

Archetti, C., Speranza, M. G., 2012. Vehicle routing problems with split deliveries. International Transactions in Operational Research 19 (1-2), 3–22.

Archetti, C., Speranza, M. G., Hertz, A., 2006. A tabu search algorithm for the split delivery vehicle routing problem. Transportation science 40 (1), 64–73.

Bai, R., Burke, E. K., Gendreau, M., Kendall, G., 2007. A simulated annealing hyper-heuristic: Adaptive heuristic selection for different vehicle routing problems. In: The 3rd Multidisciplinary International Conference on Scheduling: Theory and Applications, MISTA 2007, Paris, France.

Battarra, M., Monaci, M., Vigo, D., 2009. An adaptive guidance approach for the heuristic solution of a minimum multiple trip vehicle routing problem. Computers & Operations Research 36 (11), 3041–3050.

Berbeglia, G., Cordeau, J., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: a classification scheme and survey. Top 15 (1), 1–31.

Braekers, K., Caris, A., Janssens, G., 2013. Integrated planning of loaded and empty container movements. OR Spectrum 35 (2), 457–478.

Brandão, J., 2006. A new tabu search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research 173, 540–555.

Caris, A., Janssens, G., 2009. A local search heuristic for the pre- and end-haulage of intermodal container terminals. Computers and Operations Research 36 (10), 2763–2772.

Cheung, R., Shi, N., Powell, W., Simao, H., 2008. An attributedecision model for cross-border drayage problem. TransportationnResearch Part E: Logistics and Transportation Review 44 (2), 217–234.

Chung, K., Ko, C., Shin, J., Hwang, H., Kim, K., 2007. Development of mathematical models for the container road transportation in korean trucking industries. Computers & Industrial Engineering 53 (2), 252–262.

Clarke, G., Wright, J. W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12 (4), 568–581.

Gendreau, M., Hertz, A., Laporte, G., 1992. New insertion and postoptimization procedures for the traveling salesman problem. Operations Research 40, 1086–1094.

Glover, F., Laguna, M., 1998. Tabu Search. Springer.

Goetschalckx, M., Jacobs-Blecha, C., 1989. The vehicle routing problem with backhauls. European Journal of Operational Research 42 (1), 39–51.

Gribkovskaia, I., Laporte, G., 2008. One-to-many-to-one single vehicle pickup and delivery problems. In: Golden, B., Raghavan, S., Wasil, E., Sharda, R., Voss, S. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Vol. 43. Springer, Berlin, pp. 359–377.

Hart, S., 2005. Adaptive heuristics. Econometrica 73 (5), 1401–1430.

Imai, A., Nishimura, E., Current, J., 2007. A lagrangian relaxation-based heuristic for the vehicle routing with full container load. European journal of Operational Research 176 (1), 87–105.

Jula, H., Dessouky, M., Ioannou, P., A., C., 2005. Container movement by trucks in metropolitan networks: modeling and optimization. TransportationnResearch Part E: Logistics and Transportation Review 41 (3), 235–259.

Kramer, O., 2008. Self-adaptive heuristics for evolutionary computation. Springer.

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

Macharis, C., Bontekoning, Y., 2004. Opportunities for or in intermodal freight transportresearch: A review. European journal of Operational Research 153 (2), 400–416.

Mingozzi, A., Giorgi, S., R., B., 1999. An exact method for the vehicle routing problem with backhauls. Transportation Science 33 (3), 315–329.

Mitra, S., 2005. An algorithm for the generalized vehicle routing problem with backhauling. Asia-Pacific journal of Operational Research 22 (2), 153–169.

Mitra, S., 2008. A parallel clustering technique for the vehicle routing problem with split deliveries and pickups. Journal of the Operational Research Society 59 (11), 1532–1546.

Nagl, P., 2007. Longer combination vehicles (lcv) for asia and pacific region: Some economic implications.
URL http://www.unescap.org/pdd/publications/workingpaper/wp_07_02.pdf

Namboothiri, R., Erera, A., 2008. Planning local container drayage operations given a port access appointment system. Transportation Research Part E: Logistics and Transportation Review 44 (2), 185–202.

Nossack, J., Pesch, E., 2013. A truck scheduling problem arising in intermodal container transportation. European Journal of Operational Research 230 (3), 666–680.

Olivera, A., Viera, O., 2007. Adaptive memory programming for the vehicle routing problem with multiple trips. Computers & Operations Research 34 (1), 2847.

Osman, I., Wassan, N., 2002. A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. Journal of Scheduling 5, 263–285.

Parragh, S. N., Doerner, K. F., Hartl, R. F., 2008. A survey on pickup and delivery problems. part i: Transportation between customers and depot. Journal fur Betriebswirtschaft 58 (1), 21–51.

Ropke, S., Pisinger, D., 2006. A unified heuristic for a large class of vehicle routing problems with backhauls. European Journal of Operational Research 171 (3), 750–775.

Sterzik, S., Kopfer, H., 2013. A tabu search heuristic for the inland container transportation problem. Computers & Operations Research 40 (4), 953–962.

Toth, P., Vigo, D., 1997. An exact algorithm for the vehicle routing problem with backhauls. Transportation Science 31 (4), 372–385.

Toth, P., Vigo, D., 1999. A heuristic algorithm for the symmetric and asymmetric vehicle routing problem with backhauls. European Journal of Operational Research 113 (3), 528–543.

Toth, P., Vigo, D., 2002. VRP with backhauls. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, PA, pp. 195 – 224.

Vidal, T., Crainic, T., Gendreau, M., Prins, C., 2012. A unified solution framework for multi-attribute vehicle routing problems. Tech. Rep. CIRRELT 2012-23, University of Montreal, Canada.

Vidovic, M., Radivojevic, G., Rakovic, B., 2011. Vehicle routing in containers pickup up and delivery processes. Procedia-Social and Behavioral Sciences 20, 335–343.

Zachariadis, E., Kiranoudis, C., 2012. An effective local search approach for the vehicle routing problem with backhauls. Expert Systems with Applications 39, 3174–3184.

Zhang, R., Yun, W., Kopfer, H., 2010. Heuristic-based truck scheduling for inland container transportation. OR Spectrum 32 (3), 787–808.

Zhang, R., Yun, W., Moon, I., 2011. Modeling and optimization of a container drayage problem with resource constraints. International journal of Production Economics 133 (1), 351–359.