WORKING PAPER SERIES

The Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes

Tino Henke/M. Grazia Speranza/Gerhard Wäscher

Working Paper No. 6/2014

Impressum (§ 5 TMG)

Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
Tino Henke, M. Grazia Speranza, Gerhard Wäscher
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Postfach 4120
39016 Magdeburg
Germany

http://www.fww.ovgu.de/femm

Bezug über den Herausgeber
ISSN 1615-4274

The Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes

Tino Henke
Department of Management Science, Otto-von-Guericke-University Magdeburg, 39106 Magdeburg, Germany
tino.henke@ovgu.de

M. Grazia Speranza
Department of Quantitative Methods, University of Brescia, 25122 Brescia, Italy
speranza@eco.unibs.it

Gerhard Wäscher
Department of Management Science, Otto-von-Guericke-University Magdeburg, 39106 Magdeburg, Germany
gerhard.waescher@ovgu.de

Abstract

In this paper, a capacitated vehicle routing problem is discussed which occurs in the context of glass waste collection. Supplies of several different product types (glass of different colors) are available at customer locations. The supplies have to be picked up at their locations and moved to a central depot at minimum cost. Different product types may be transported on the same vehicle, however, while being transported they must not be mixed. Technically this is enabled by a specific device, which allows for separating the capacity of each vehicle individually into a limited number of compartments where each compartment can accommodate one or several supplies of the same product type. For this problem, a model formulation and a variable neighborhood search algorithm for its solution are presented. The performance of the proposed heuristic is evaluated by means of extensive numerical experiments. Furthermore, the economic benefits of introducing compartments on the vehicles are investigated.

Keywords: vehicle routing, multiple compartments, glass waste collection, variable neighborhood search, heuristics

1 Introduction

The vehicle routing problem, which will be discussed in this paper, is a variant of the classic capacitated vehicle routing problem (CVRP; for surveys see Golden et al., 2008, Laporte, 2009, or Toth and Vigo, 2002) and occurs in the context of glass waste collection in Germany. Glass waste has to be recycled by law and is used as a raw material for the production of new glass products. It has to be taken to recycling stations by the consumers where it is disposed into different containers according to the color of the waste (usually colorless, green and brown glass). Colors are kept separated because the production of new glass products is less cost-intensive if the glass waste is not too inhomogeneous with respect to its color. Trucks, which are located at a depot of a recycling company, pick up the glass waste from the recycling stations. Since they possess a relatively large loading capacity, they can call at several recycling stations before they have to return to the depot. Recent truck models are equipped with a special device which allows for introducing bulkheads in predefined positions of the loading space such that it can be split into different compartments and, thus, enabling transportation of glass waste with different colors on the same truck without mixing the colors on the tour. This gives rise to the question how the tours of the trucks should be designed given the availability of a device of this kind.

The problem under discussion can be classified as a multi-compartment vehicle routing problem (MCVRP). However, it is different from the ones previously discussed in the literature with respect to the following properties:

- The size of each compartment is not fixed in advance but can be determined individually for each vehicle/each tour.
- The size of the compartments can only be varied discretely, i.e. the walls separating the compartments from each other can only be introduced in specific, predefined positions.
- The number of compartments, into which the capacity of a vehicle is divided, can be identical to the number of product types (glass waste types) but can also be smaller.

Consequently, not only the vehicle tours have to be determined, but it has also to be decided for each vehicle/tour (i) into how many compartments the vehicle capacity should be divided, (ii) what the size of each compartment should be, and (iii) which product type should be assigned to each compartment.

The problem is NP-hard, since it is a generalization of the CVRP (see, for example, Toth and Vigo, 2002). By application of a mathematical model-based exact solution approach, we were only able to solve problem instances with a limited size to optimality. Therefore, a heuristic, namely a variable neighborhood search (VNS), has been developed and will be presented. According to the best of our knowledge, this is the first method which has been proposed for this problem so far. We will further analyze what the economic benefits are which stem from the introduction of flexibly sizable compartments.

The remainder of this paper is organized as follows. Section 2 presents a formal definition and a mathematical formulation of the problem. The relevant literature related to the MCVRP is discussed in Section 3. In Section 4, the proposed variable neighborhood search algorithm is introduced. Extensive numerical experiments have been performed in order to evaluate the mathematical model and the VNS. The design of these experiments and the corresponding results are presented in Section 5. Finally, the main findings are summarized and an outlook on future research is given in Section 6.

2 Problem Description and Formulation

The multi-compartment vehicle routing problem with flexible compartment sizes (MCVRP-FCS) can be formulated as follows: Let an undirected, weighted graph $G = (V, E)$ be given which consists of a vertex set $V = \{0,1, \dots, \mathrm{n}\}$, representing the location of the depot ($\{0\}$) and the locations of n customers ($\{1, \dots, \mathrm{n}\}$), and an edge set $E = \{(\mathrm{i},\mathrm{j}): \mathrm{i},\mathrm{j} \in V, \mathrm{i} < \mathrm{j}\}$, representing the edges which can be travelled between the different locations. To each of these edges, a non-negative cost c_{ij}, $(\mathrm{i},\mathrm{j}) \in E$, is assigned.

Further, let a set P of product types be given. At each vertex (except for the depot) exists a non-negative supply $\mathrm{s}_{\mathrm{ip}} (\mathrm{i} \in V\backslash\{0\}, \mathrm{p} \in P)$ of each of the product types. The supplies have to be collected at their locations and transported to the depot without the product types being mixed. A location may be visited several times in order to pick up different product types. However, if being picked up, each supply has to be loaded in total. In other words, a split collection of a single supply is not permitted.

For the purpose of transportation, a set K of homogeneous vehicles is available, each equipped with a total capacity Q. Individually for each vehicle $\mathrm{k} \in K$, the total capacity Q can be divided into a limited number $\widehat{\mathrm{m}}$ of compartments, $\widehat{\mathrm{m}} \leq |P|$, which allows for loading products of different types on a single vehicle while keeping them separated during transportation. The size of the compartments can be varied discretely in equal step sizes, i.e. each compartment size, but also the total vehicle capacity Q, is a integer multiple of a basic compartment unit size $\mathrm{q}^{\mathrm{unit}}$. Let the set of these multiples be denoted by $M = \{0, 1, 2, \dots, \mathrm{m}^{\max}\}$ where $\mathrm{m}^{\max} = \mathrm{Q}/\mathrm{q}^{\mathrm{unit}}$. Then $\mathrm{q}_{\mathrm{m}} = \frac{1}{\mathrm{m}^{\max}} \cdot \mathrm{m}, \mathrm{m} \in M$, denotes a compartment size relative to the total capacity Q which consists of m ($\mathrm{m} \in M$) multiples of the basic compartment unit size $\mathrm{q}^{\mathrm{unit}}$.

What has to be determined is a set of vehicle tours, an assignment of product types to the vehicles and the sizes of the corresponding compartments such that all supplies are collected, that the capacity of none of the used vehicles is exceeded, and that the total cost of all edges to be travelled is minimized.

This problem involves the following partial decisions to be made simultaneously:

- assignment of product types to each of the vehicles
  (this decision determines which product types can be collected by each vehicles);
- determination of the size of each compartment
  (this decision fixes for each vehicle how its total capacity is split into compartments);
- assignment of supplies to each of the vehicles
  (this decision implicitly includes an assignment of locations to vehicles);
- sequencing of the locations for each of the vehicles
  (this decision determines for each vehicle in which sequence the assigned locations are to be visited).

We note that every vehicle routing problem involves decisions of the last two types, while the first and the second one define the uniqueness of the MCVRP-FCS.

In order to formulate a mathematical model for the MCVRP-FCS, we introduce the following four types of variables:

$$u_{ipk} = \begin{cases} 1, \text{ if supply of product type p at location i is collected by vehicle k,} \\ 0, \text{ otherwise,} \end{cases} \qquad i \in V\backslash\{0\}, p \in P, k \in K;$$

$$x_{ijk} = \begin{cases} 2, \text{ if i=0 and edge (i,j) is used twice by vehicle k,} \\ 1, \text{ if edge (i,j) is used once by vehicle k,} \\ 0, \text{ otherwise,} \end{cases} \qquad i, j \in V, i < j, k \in K;$$

$$y_{pkm} = \begin{cases} 1, \text{ if size } q_m \text{ is selected for product type p in vehicle k,} \\ 0, \text{ otherwise,} \end{cases} \qquad p \in P, k \in K, m \in M;$$

$$z_{ik} = \begin{cases} 1, \text{ if location i is visited by vehicle k,} \\ 0, \text{ otherwise,} \end{cases} \qquad i \in V, k \in K.$$

The objective function and the constraints of the model can then be formulated as follows:

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

$$\sum_{k\in K} u_{ipk} = 1 \qquad \forall i \in V\backslash\{0\}, p \in P,\ s_{ip} > 0 \tag{2}$$

$$u_{ipk} \leq z_{ik} \qquad \forall i \in V\backslash\{0\}, p \in P, k \in K \tag{3}$$

$$z_{ik} \leq z_{0k} \qquad \forall i \in V\backslash\{0\}, k \in K \tag{4}$$

$$\sum_{\substack{j\in V:\\ j>0}} \sum_{k\in K} x_{0jk} \leq 2 \cdot |K| \tag{5}$$

$$\sum_{\substack{j\in V:\\ i<j}} x_{ijk} + \sum_{\substack{j\in V:\\ j<i}} x_{jik} = 2 \cdot z_{ik} \qquad \forall i \in V, k \in K \tag{6}$$

$$\sum_{p\in P} \sum_{m\in M} y_{pkm} \leq \hat{m} \qquad \forall k \in K \tag{7}$$

$$\sum_{p\in P} \sum_{m\in M} q_m \cdot y_{pkm} \leq 1 \qquad \forall k \in K \tag{8}$$

$$\sum_{i\in V\backslash\{0\}} s_{ip} \cdot u_{ipk} \leq Q \cdot \sum_{m\in M} q_m \cdot y_{pkm} \qquad \forall p \in P, k \in K \tag{9}$$

$$\sum_{i\in S} \sum_{\substack{j\in S:\\ i<j}} x_{ijk} \leq |S| - 1 \qquad \forall k \in K,\ S \subseteq V\backslash\{0\}, |S| > 2 \tag{10}$$

$$u_{ipk} \in \{0,1\} \qquad \forall i \in V\backslash\{0\}, p \in P, k \in K \tag{11}$$

$$x_{0jk} \in \{0,1,2\} \qquad \forall j \in V\backslash\{0\},\ k \in K \tag{12}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in V\backslash\{0\}, j \in V\backslash\{0\}, i < j, k \in K \tag{13}$$

$$y_{pkm} \in \{0,1\} \qquad \forall p \in P, k \in K, m \in M \tag{14}$$

$$z_{ik} \in \{0,1\} \qquad \forall i \in V, k \in K \tag{15}$$

The objective function (1) defines the total cost of all tours, which has to be minimized. Constraints (2) guarantee that each positive supply is assigned to exactly one vehicle, while constraints (3) ensure that a supply may only be satisfied by a vehicle if the corresponding customer location is visited by this vehicle. Constraints (4) make sure that the depot is included in each tour. According to constraints (5), the number of tours is limited by the number of available vehicles. Constraints (6) represent the vehicle flow constraints. Constraints (7) ensure that the number of product types assigned to a vehicle may not exceed the maximum number of compartments. Constraints (8) assert for each vehicle that the sum of the compartment sizes must be smaller than or equal to the vehicle capacity. In addition to that, constraints (9) guarantee for each compartment that its capacity is not exceeded by the assigned supplies. Constraints (10) represent the subtour-elimination constraints. Finally, constraints (11) – (15) characterize the variable domains.

3 Literature Review

Some variants of the MCVRP have been discussed in the literature already. El Fallahi et al. (2008) describe a MCVRP arising in the context of distribution of animal food to farms where the sizes of the compartments and the corresponding assignment of product types to the compartments are fixed in advance. They propose a memetic algorithm and a tabu search algorithm for solving this problem. Muyldermans and Pang (2010) consider a similar problem and develop a guided local search algorithm. Furthermore, they identify problem characteristics under which transportation of several product types on vehicles with multiple (a priori fixed) compartments outperforms separate transportation of the different product types on vehicles with a single compartment. Chajakis and Guignard (2003) present a similar problem in the context of deliveries to convenience stores. For their problem, they develop heuristics based on Lagrangian Relaxations. In contrast to these contributions, in this paper a MCVRP with flexible compartment sizes and no a priori assignment of product types to compartments is considered.

Avella et al. (2004) and Brown and Graves (1981) describe applications of the MCVRP in the context of petrol replenishment. In these problems, the compartment sizes are fixed in advance, while the assignment of product types to compartments is not. In contrast to the MCVRP-FCS, only one supply can be assigned to each compartment, because petrol tanks must be emptied completely during the visit of one customer. Morever, Cornillier et al. (2008) extend the problem to multiple periods with inventory components.

Derigs et al. (2011) introduce and examine two variants of the MCVRP: The first variant is, in general, similar to the problem described by El Fallhi et al. (2008), Muyldermans and Pang (2010), and Chajakis and Guignard (2003), whereas the second variant is characterized by continuous flexible compartment sizes, while the assignment of product types to compartments is not fixed. However, in contrast to the MCVRP-FCS, each customer may have multiple demands for the same product type. The case in which the number of compartments may be smaller than the number of product types is not explicitly considered. The authors introduce a general mathematical model, i.e. one which covers both problem variants, and a large neighborhood search algorithm which solves their variants of the MCVRP.

A problem similar to the MCVRP-FCS is presented by Caramia and Guerriero (2010). They present a case study-oriented MCVRP with heterogeneous vehicles in milk collection with fixed compartment sizes but no a priori assignment of product types to vehicles. Also, the number of product types (four in the case study) may be smaller than the number of compartments per vehicle (three or more compartments). In contrast to the MCVRP-FCS, they also consider trailers which can be added to a vehicle. For their problem, they present a model-oriented heuristic procedure which assigns demands to vehicles in a first step and determines specific routes for these vehicles in a second step.

Finally, Mendoza et al. (2010, 2011) discuss the multi-compartment vehicle routing problem with stochastic demands for which they propose several construction procedures and a memetic algorithm.

Apart from the mentioned applications, multi-compartment routing problems also occur in maritime transportation (Fagerholt and Christiansen, 2000). Only recently, Archetti et al. (2013) presented theoretical comparisons between multi-commodity and single-commodity vehicle routing problems, where they provide theoretical and empirical insights on relationships between the split delivery vehicle routing problem, the multi-compartment vehicle routing problem, and a combined split delivery and multi-compartment vehicle routing problem.

Our paper attends to additional aspects which – according to the best of our knowledge – have not been considered in detail in the literature before. These aspects include the case of discretely flexible compartment sizes and the case in which the number of compartments per vehicle is smaller than the number of products types. Especially the second aspect defines a new, unique problem structure for which we suggest a heuristic solution approach.

4 Variable Neighborhood Search

4.1 Overview

In order to determine good solutions for large problem instances of the MCVRP-FCS, a variable neighborhood search approach with multiple starts (MS-VNS) has been developed. The concept of VNS was first proposed by Mladenović and Hansen (1997) and Hansen and Mladenović (2001). VNS is a local search-based metaheuristic which explores the solution space by means of multiple neighborhood structures.

The general VNS framework can be described as follows: Given a set of different neighborhood structures which are sequenced in a specific order, VNS starts from an initial solution x (also first *incumbent solution*) and randomly selects a neighbor x' of x according to the first neighborhood structure (*shaking*). The neighbor is subsequently improved by application of a local search algorithm, providing a solution x'' (*improvement phase #1*). If this solution does not satisfy a given acceptance criterion, another neighbor x''' of the incumbent x solution is selected randomly, this time, however, from the neighborhood defined by the next neighborhood structure in the sequence, and the above-described process is repeated. Otherwise, if x″ satisfies the acceptance criterion, it is updated as the new incumbent solution, i.e. x := x″, and the index of the neighborhood structure is set back to 1, i.e. for the next pass of the improvement phase #1, a neighbor x' of the new incumbent solution of x is selected from the neighborhood defined by the first neighborhood structure. Should no new incumbent solution be determined over a run through all neighborhood structures, the procedure restarts from the first neighborhood structure in the sequence. The procedure continues until a predefined termination criterion is attained. In the following, the execution of a shaking step and the subsequent improvement phase will be referred to as one *iteration*.

Usually the sequence of the neighborhood structures is chosen in a way that the size of the neighborhoods increases as the VNS proceeds. As a consequence, initially the procedure explores neighbors from the incumbent solution, which can be obtained by small changes, while it only moves on to explore larger changes if the search for better solutions was unsuccessful in such "close" neighborhoods.

As we have mentioned before, an important aspect of the MCVRP-FCS are the four different types of partial decisions which have to be made (see Section 2). The usage of a single neighborhood structure may revise several of these decisions simultaneously. However, the combination of decisions is often identical. By the usage of more than one neighborhood structure, VNS is able to revise different combinations of problem decisions, which permits a more diverse exploration of the solution space.

One of the disadvantages of this algorithm is that it often tends to get stuck in a local optimum. In order to overcome this drawback, we implemented our VNS in connection with a multi-start approach. The VNS restarts several times from different initial solutions, which are generated by a randomized construction procedure. The number of starts (in the following called *loops*) is also limited by a termination criterion (*ms termination criterion*). Furthermore, each time a new best solution has been found, it is attempted to improve this solution by a second improvement phase (*improvement phase #2*).

A pseudo-code of our VNS procedure is presented in Fig. 4.1. Z^{best} denotes the objective function value of the best solution found over all loops and Z(x) denotes the objective function value of an arbitrary solution x. Details of the randomized construction procedure, the solution space, the neighborhood structures and improvement phases, and on the acceptance and termination criteria will be explained in detail in the following subsections.

```
input: problem data, number of neighborhood structures $k^{max}$;

$Z^{best}$:= ∞;
do
  generate an initial solution x with objective function value Z(x) randomly;
  set neighborhood structure index k to k:= 1;
  do
    select a neighbor x’ from neighborhood structure k of x randomly (shaking);
    apply local search to x’ and determine a solution x’’ (improvement phase #1);
    if Z(x’’) satisfies the acceptance criterion then
      x:= x’’; k:= 1;
      if Z(x) < $Z^{best}$ then
        $x^{best}$:= x;
        apply local search procedure to $x^{best}$ (improvement phase #2);
        $Z^{best}$:= Z($x^{best}$);
      endif
    else
      k:= k + 1;
      if k = $k^{max}$ + 1 then
        k:= 1;
      endif
    endif
  until vns termination criterion is satisfied
until ms termination criterion is satisfied
output: $x^{best}$;
```

Figure 4.1: Pseudo-code of a VNS for the MCVRP-FCS

4.2 A Randomized Construction Procedure for the Generation of Initial Solutions

Each initial solution is generated by means of the following randomized construction procedure. At the beginning, all (positive) supplies are sequenced randomly. According to this sequence, one supply after another is assigned to a vehicle. Starting from the vehicle with the lowest index, it is checked whether the supply can (still) be accommodated by the respective vehicle. This is the case if (i) a compartment has already been opened for the respective product type (because a supply of this product type has already been assigned to the vehicle) or it still can be opened for this product type (because there is still enough vehicle capacity available and the maximum number of compartments is not exceeded) and (ii) the size of the compartment is at least as large as the supply. If the conditions (i) and (ii) hold, then the supply is assigned to the vehicle; otherwise the next vehicle will be checked. The procedure stops when all supplies have been assigned. We remark that the assignment of supplies to vehicles also constitutes an assignment of locations to vehicles. The latter is used as a basis for the determination of an initial set of routes by application of the Lin-Kernighan Heuristic (Lin and Kernighan, 1973).

We further note that the described procedure of assigning supplies to vehicles may result in an assignment, which contains more vehicles than actually available. In this case, a large penalty is added to the objective function for each additional vehicle. Consequently, the algorithm is directed to finding solutions with a feasible number of vehicles during the improvement phases. Once a solution with a feasible number of vehicles has been found, then only solutions feasible with respect to the number of available vehicles will be generated.

4.3 Solution Space

Throughout the search of the solution space, not only feasible solutions but also solutions which are infeasible with regard to the vehicle capacity can be accepted as incumbent solutions. In order to deal with these infeasibilities, a penalty term is added to the total cost of a solution within the objective function (Gendreau et al., 1994). This penalty term is dependent on the extent according to which the vehicle capacities are violated (excess capacity utilization) and a penalty factor α. For the determination of the excess capacity utilization Q'_k of a vehicle k, the selected compartment sizes are considered instead of the actual supplies transported in these compartments:

$$Q'_k = \max(0; \sum_{p \in P} \sum_{m \in M} q_m \cdot y_{pkm} - Q)$$

The modified objective function value $\tilde{Z}_0$ is then obtained by adding the weighted sum of the total excess capacity utilization of all vehicles to the original objective function value Z_0:

$$\tilde{Z}_0 = Z_0 + \alpha \cdot \sum_{k \in K} Q'_k$$

The value of the penalty factor α is adjusted dynamically. At the beginning of each loop, the factor is set to an identical initial value. If no feasible solution was determined for a certain number of consecutive iterations, the factor is increased. If a feasible solution has eventually been found, the factor is reset to its initial value. In this manner, infeasibilities get penalized only lightly during the early stages of the application of the algorithm. However, the longer the solutions remain infeasible, the stronger the violation gets. Consequently, the algorithm is directed to move to a feasible region of the solution space.

4.4 Neighborhood Structures

The crucial part of the design of a VNS approach consists in defining the neighborhood structures from which the solutions are to be selected randomly in the shaking step. Ten different neighborhood structures have been implemented in order to deal with the various decisions, which have to be made when solving the MCVRP-FCS. They can be distinguished into supply-related, location-related, and product type-related neighborhood structures, focusing on changes of supply-vehicle-assignments, location-vehicle-assignments, and product-vehicle-assignments, respectively.

Two supply-related neighborhood structures are used, of which the first one is based on supply shifts, while the second one is based on supply swaps. A supply shift selects a single supply randomly, deletes it from its current vehicle and inserts it into another, randomly selected, vehicle. A supply swap selects two individual supplies randomly from two different vehicles and exchanges both supplies.

Four different location-related neighborhood structures have been implemented, based on complete location shifts, partial location shifts, location splits, and location swaps. For a complete location shift, one location-vehicle-assignment from the incumbent solution is randomly selected. Then, all supplies related to this specific location, which are currently assigned to the vehicle, are moved collectively to another vehicle. In a partial location shift, only a randomly determined subset of these supplies is shifted. A location split operator works in a similar way as a complete location shift operator except for the difference that each supply may be shifted to a different vehicle. In a location swap, two location-vehicle-assignments are selected randomly and the corresponding supplies of the locations are exchanged between the vehicles. Pretests have also shown that it is sensible to change more than one location-vehicle-assignment simultaneously, e.g. for a complete location shift not only one location but two locations are shifted to other vehicles.

Four different options for the product type-related neighborhood structures have been implemented. Analogously to the location-related neighborhood structures, they are established by complete product shifts, partial product shifts, product splits, and product swaps. In contrast to location-related operators, product-vehicle-assignments instead of location-vehicle-assignments are selected randomly. As for a product shift, the supplies of the selected product type are moved either completely (complete shift) or partially (partial shift) to another vehicle. For the product split, the supplies may be assigned to different vehicles. In a product swap, two product-vehicle-assignments are selected randomly and the corresponding total supplies are exchanged between the two vehicles.

In the VNS algorithm for the MCVRP-FCS, the described neighborhood structures have been implemented in the following sequence: demand shift, demand swap, partial location shift, complete location shift, location swap, location split, partial product shift, complete product shift, product swap, and product split. Only such moves are considered which neither violate the maximum number of product types per vehicle nor the number of available vehicles.

4.5 Improvement Phases

During the improvement phase immediately following the shaking step (improvement phase #1), all routes of a solution which have been changed by the shaking step are improved by a local search procedure. This procedure uses the 3-edge neighborhood structure proposed by Lin (1965), where the neighborhood is established by an operator, according to which three edges are eliminated from a route and three new edges are inserted. The search applies the first improvement principle, i.e. the first route is accepted from a neighborhood which has smaller route costs than the current one. The improvement phase terminates when no further improved route can be identified in the neighborhood of the currently considered route.

In the improvement phase, which follows the identification of a new best solution (improvement phase #2), the well-known Lin-Kernighan Heuristic (Lin and Kernighan, 1973) is applied to each of the routes in the solution. For our implementation we used the code provided by Helsgaun (2000).

4.6 Acceptance Criterion

VNS algorithms tend to get stuck in local optima (cf. Hansen and Mladenović 2001). In order to avoid this drawback, we do not only use several starts of the heuristic but also an acceptance criterion which allows for accepting solutions with objective function values worse than those of the best solution found in a loop. The acceptance criterion used for this algorithm is based on threshold accepting (Dueck and Scheuer, 1990) with a dynamically self-adjusting threshold factor.

The dynamic self-adjustment principle is similar to the adjustment procedure for the penalty factor described in Section 4.2. The factor starts with an initial value of 1. After a certain number of consecutive iterations in which no new incumbent solution was accepted, this factor is increased by a certain value. If a solution is eventually accepted as incumbent solution, the factor is reset to 1 again. In this way, the threshold increases slowly if the algorithm is not able to find a better solution in the region of the solution space which is currently explored, enabling the algorithm to overcome local optima.

4.7 Termination Criteria

A loop terminates after a certain number of iterations without finding a new best solution. The algorithm terminates after a certain number of loops without finding a new globally best solution.

5 Numerical Experiments

5.1 Overview

The above-presented model of the MCVRP-FCS was implemented – with a separation procedure for the subtour elimination constraints – in Microsoft Visual C++ 2012 using CPLEX 12.5. For the MS-VNS algorithm, Microsoft Visual C++ 2012 has been used.

Three sets of numerical experiments were performed in order to evaluate the two approaches. In the first set, the exact solution approach and the VNS algorithm have been applied to relatively small, randomly generated problem instances. These experiments serve two purposes. Firstly, they provide an impression of the limits of the problem size up to which the MCVRP-FCS can be solved optimally by means of a standard LP solver in reasonable computing time. Secondly, by comparing the results from the MS-VNS approach to the results obtained by the exact approach, the solution quality of the MS-VNS approach can be assessed.

In a second set of experiments, large, randomly generated instances of the MCVRP-FCS were considered, to which the MS-VNS algorithm was applied. These experiments have been performed in order to study the behavior of the algorithm in greater detail. They were meant to provide insights into how different problem parameters affect solution quality and computing times and, in particular, which parameters make the problem difficult to solve.

In the third set of experiments, the MS-VNS algorithm was applied to problem instances from practice. These experiments were meant to assess the benefits from introducing vehicles with compartments of flexible size in comparison to the utilization of vehicles with a single compartment only.

All experiments were performed on a 3.2 GHz and 8GB RAM personal computer.

5.2 Problem Generation

As we have explained above, the MCVRP-FCS is different from other multi-compartment vehicle routing problems described in the literature so far. Therefore, we could not use benchmark instances from the literature in our experiments. Instead, we randomly generated new problem instances.

In order to establish different classes of problem instances, the parameters of the MCVRP-FCS have been dealt with in the following way. The number of locations n, the number of product types $|P|$, and the number of compartments $\widehat{\mathrm{m}}$ have been taken as controllable parameters. The respective values chosen for the experiments are given below. Concerning the number $\bar{\mathrm{s}}$ of product types of which supplies are available at each location (number of supplies), three options were considered: small, medium, and large. A small number of supplies corresponds to a situation in which each location provides supplies for at least one product type and at most for one third of the product types ($\bar{\mathrm{s}} = 1$, small number of supplies). A medium number of supplies represents a situation where supplies of at least more than one third and at most two thirds of the product types are available at each location ($\bar{\mathrm{s}} = 2$, medium number of supplies). Finally, for a large number of supplies, supplies are available at each location of at least more than two thirds of the product types up to $|P|$ (i.e. all) product types ($\bar{\mathrm{s}} = 3$, large number of supplies).

The actual supplies for all product types at all locations s_{ip} were generated by means of the following procedure. At first, an initial total supply ($\mathrm{S}^{\mathrm{Total}}$) for all product types across all locations was randomly generated from the integer interval $[125 \cdot \mathrm{n}; 175 \cdot \mathrm{n}]$. This total supply was then split into individual supplies at the locations in several steps. Firstly, based on the respective value of the parameter $\bar{\mathrm{s}}$, a number for the supplied product types ($\bar{\mathrm{s}}_{\mathrm{i}}$, $\mathrm{i} \in V\backslash\{0\}$) was generated randomly for each location i from the corresponding parameter interval for $\bar{\mathrm{s}}$. In order to determine which specific product types at a specific location have an actual supply, i.e. $\mathrm{s}_{\mathrm{ip}} > 0$, a random number out of the interval $[0{,}0.5]$ for the first third of product types, $[0{,}0.3]$ for the second third of product types, and $[0{,}0.2]$ for the last third of product types was drawn. For each location i, these random numbers were sorted in decreasing order, whereas the first $\bar{\mathrm{s}}_{\mathrm{i}}$ product types in this sequence were selected to have a supply greater than zero. Then, the total supply quantities for each product type ($\mathrm{S}_{\mathrm{p}}^{\mathrm{Total}}$, $\mathrm{p} \in P$) were determined in such a manner that 50% of the total supply $\mathrm{S}^{\mathrm{Total}}$ was assigned to the first third of the product types, 30% of the total supply to the second third, and 20% of the total supply to the last third of the product types. Finally, the actual supplies ($\mathrm{s}_{\mathrm{ip}}, \mathrm{i} \in V\backslash\{0\}, \mathrm{p} \in P, \mathrm{s}_{\mathrm{ip}} > 0$) were generated by drawing a random number for each supply, which were subsequently normalized according to the total supply of the corresponding product type $\mathrm{S}_{\mathrm{p}}^{\mathrm{Total}}$. The biased distribution of product type supplies was used in order to generate realistic instances, since in the glass collection problem the different glass types are more and less frequently disposed.

Finally, the vehicle capacity was set to 1,000 for each instance and the number of vehicles was determined by solving a corresponding bin-packing problem exactly in order to guarantee a feasible solution to the problem instance.

All instances can be found online at http://www.mansci.ovgu.de/mansci/en/Research/Materials/2014-p-394.html.

5.3 Experiments with Small, Randomly-Generated Instances

The following parameter values were used for the generation of instances for the first set of experiments:

- number of customer locations n = 10;
- number of products types $|P| \in \{3, 6, 9\}$;
- number of compartments $\widehat{\mathrm{m}} \in \{2, 3\}$ for $|P| = 3$, $\widehat{\mathrm{m}} \in \{2, 4, 6\}$ for $|\mathrm{P}| = 6$, $\widehat{\mathrm{m}} \in \{2, 4, 7, 9\}$ for $|P| = 9$;
- number of supplies $\bar{\mathrm{s}} \in \{1, 2, 3\}$.

The number of locations considered in this set of experiments may appear to be small at first sight, but it has to be emphasized that, apart from the number of locations, also the total number of supplies determines the problem size. Since the number of different supplies at each location and the maximum number of compartments per vehicle make the MCVRP-FCS special, we have chosen to explore these parameters in the first place at the expense of considering not more than 10 customer locations. This confinement was necessary in order to limit the respective optimization models to sizes which were still manageable by CPLEX.

The parameter settings resulted in 27 problem classes, for each of which 50 instances have been generated. To all 1,350 instances, the exact approach and the MS-VNS algorithm have been applied. 500n$|P|$ iterations without improvement (vns termination criterion; cf. Fig. 4.1) and n$|P|$/6 loops without improvement (ms termination criterion) were used as termination criteria for the MS-VNS algorithm.

For each problem class, Table 5.1 presents the number of instances, which have been solved to a proven optimum by the exact approach (#opt.exact) and by the MS-VNS algorithm (#opt.vns), the average objective function value (total cost) per problem instance obtained by the exact approach (tc.exact) and by the MS-VNS algorithm (tc.vns), and computing times per problem instance in seconds for the exact approach (cpu.exact) and the MS-VNS algorithm (cpu.vns). For the exact approach, also the maximal computing time (cpu.exactmax) needed for an instance is reported for each problem class. Furthermore, the average per-instance deviation (aver.error) of the objective function value obtained by the MS-VNS algorithm from the objective function value obtained by the exact approach, and the corresponding maximum deviation (max.error) across all instances in a class are presented.

parameters			cplex				vns			error	
$\|P\|$	$\hat{m}$	$\bar{s}$	#opt.exact	tc.exact	cpu.exact	cpu.exactmax	#opt.vns	tc.vns	cpu.vns	aver.error	max.error
3	2	1 (small)	50	366.9	5.43	52.61	50	366.9	2.31	0.00%	0.00%
		2 (medium)	50	465.3	11.43	194.87	50	465.35	2.91	0.00%	0.00%
		3 (large)	50	596.2	101.33	4686.80	50	596.16	2.86	0.00%	0.00%
	3	1 (small)	50	349.1	2.60	25.84	50	349.08	2.53	0.00%	0.00%
		2 (medium)	50	337.1	2.66	33.65	50	337.15	3.43	0.00%	0.00%
		3 (large)	50	336.6	5.00	94.45	50	336.64	3.72	0.00%	0.00%
6	2	1 (small)	50	550.0	5.91	28.53	50	549.98	12.10	0.00%	0.00%
		2 (medium)	50	767.5	4.83	21.26	50	767.48	11.46	0.00%	0.00%
		3 (large)	50	919.8	1.81	3.96	50	919.78	13.00	0.00%	0.00%
	4	1 (small)	50	367.0	5.04	47.89	50	366.98	12.47	0.00%	0.00%
		2 (medium)	50	492.1	27.79	120.60	50	492.11	14.98	0.00%	0.00%
		3 (large)	50	583.8	326.15	2895.40	49	584.46	15.06	0.15%	7.39%
	6	1 (small)	50	352.0	4.05	72.42	50	351.98	12.05	0.00%	0.00%
		2 (medium)	50	358.0	5.94	27.14	50	357.98	14.89	0.00%	0.00%
		3 (large)	50	357.4	8.96	62.29	50	357.42	15.63	0.00%	0.00%
9	2	1 (small)	50	724.2	502.44	13866.00	50	724.17	46.12	0.00%	0.00%
		2 (medium)	50	1114.3	1921.59	12333.90	50	1114.25	50.46	0.00%	0.00%
		3 (large)	50	1419.1	5400.85	76049.40	50	1419.09	55.79	0.00%	0.00%
	4	1 (small)	50	463.0	12.87	76.28	50	463.05	41.21	0.00%	0.00%
		2 (medium)	50	647.6	267.69	4653.76	50	647.57	48.00	0.00%	0.00%
		3 (large)	50	815.3	1324.58	4712.93	50	815.27	51.07	0.00%	0.00%
	7	1 (small)	50	360.2	5.25	60.90	50	360.16	37.83	0.00%	0.00%
		2 (medium)	50	456.3	93.21	2239.98	44	457.60	47.31	0.26%	6.95%
		3 (large)	50	564.5	1550.47	10854.80	49	564.60	54.68	0.02%	0.85%
	9	1 (small)	50	355.0	3.94	18.95	50	354.98	39.04	0.00%	0.00%
		2 (medium)	50	392.5	10.53	74.96	49	392.70	43.38	0.05%	2.35%
		3 (large)	50	383.4	32.94	459.02	47	383.72	46.99	0.10%	4.04%
minimum			50	336.6	1.81	2.84	1.81	3.96	2.31	0.00%	0.00%
average			50	551.6	386.37	4438.79	431.31	4954.39	25.97	0.02%	0.80%
maximum			50	1419.1	5400.85	76049.40	5400.85	76049.40	55.79	0.26%	7.39%

Table 5.1: Results for the small instances

The exact approach managed to solve all of the 1,350 instances with a maximal computing time of 21 hours. It appears that the MCVR-FCS becomes more difficult to solve (i.e. the respective computing times increase) with an increasing number of product types, with an increasing number of supplies and with a decreasing number of compartments. Especially the latter observation demonstrates that the specific (partial) decision of this problem, namely the assignment of product types to vehicles, has a significant impact on the complexity of the problem and makes it different from other vehicle routing problems. We further note that – with respect to the computing times – the exact approach presented here cannot be expected to be competitive for large, real-world instances.

Regarding the solution quality, it can be observed that the MS-VNS algorithm performs very well on the instances from these problem classes. From the 1,350 instances, the MS-VNS algorithm solves 1,338 (99.1%) optimally. The average deviation from the optimal objective function value (across all instances) only amounted to 0.02%. This is particularly remarkable since the average computing time per problem instance needed by the MS-VNS algorithm (25.97s) represents only 6.0% of the average computing time needed by the exact approach (431.31s). Regarding the maximal computing times, this observation is even more considerable, since the maximal computing time for the exact approach is 21 hours whereas the maximal computing time for the MS-VNS is only 113 seconds.

As for the computing times of the MS-VNS algorithm, it can be observed that they increase with the number of product types and the number of demands. In contrast to the exact approach, the impact of the maximum number of compartments on the computing time is inconsistent. For instances with three or six product types, no pattern is observable; for instances with nine product types, the computing times decrease with increasing number of available compartments.

5.4 Experiments with Large, Randomly-Generated Instances

For the second set of experiments, a set of large instances was generated. These instances consider 50 locations and up to 9 product types, which results in a maximum of 450 demands. For the number of product types, the number of compartments, and the number of demands, the same combinations of parameters as in the first set of experiments were used. For each combination, one instance was randomly generated. Based on these experiments, the solution quality of the MS-VNS was evaluated.

Since we were neither able to use benchmarks from the literature nor to solve any large instance to optimality, we adopted the following approach. Each instance was solved for 360 minutes by the MS-VNS. The best solutions obtained after 10, 20, 30, 40, 50, and 60, 120, 180, 240, 300, and 360 minutes were recorded and the respective objective function values were compared to the best solution found after 360 minutes. Again, $500n|P|$ iterations without improvement per loop were used.

Table 5.2 shows for each instance the total costs of the solutions found (tc) during the first hour of computation in steps of 10 minutes and the corresponding deviations to the best solution found (dev). Moreover, Table 5.3 lists similar results for the six hours of computation in steps of 60 minutes.

$\|P\|$	$\hat{m}$	$\bar{s}$	best solution found after											
			10 minutes		20 minutes		30 minutes		40 minutes		50 minutes		60 minutes	
			tc	dev	tc	dev	tc	dev	tc	dev	tc	dev	tc	dev
3	2	small	1117.79	3.96%	1117.79	3.96%	1101.82	2.47%	1097.24	2.05%	1097.24	2.05%	1080.68	0.51%
		medium	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%
		large	1529.73	4.84%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%
	3	small	1347.26	6.96%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%
		medium	1792.21	10.89%	1792.21	10.89%	1667.96	3.20%	1667.96	3.20%	1667.96	3.20%	1667.96	3.20%
		large	1830.68	2.75%	1830.68	2.75%	1830.68	2.75%	1830.68	2.75%	1830.68	2.75%	1830.68	2.75%
6	2	small	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%
		medium	1970.85	8.07%	1970.85	8.07%	1844.73	1.15%	1844.73	1.15%	1844.73	1.15%	1844.73	1.15%
		large	2879.94	20.16%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%
	4	small	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%
		medium	1627.56	6.42%	1627.56	6.42%	1627.56	6.42%	1627.56	6.42%	1627.56	6.42%	1589.01	3.90%
		large	2486.69	8.06%	2469.11	7.30%	2469.11	7.30%	2434.65	5.80%	2372.35	3.09%	2372.35	3.09%
	6	small	1648.06	3.48%	1647.35	3.43%	1631.44	2.44%	1631.44	2.44%	1631.44	2.44%	1631.44	2.44%
		medium	2069.92	3.48%	2069.92	3.48%	2069.92	3.48%	2069.92	3.48%	2069.92	3.48%	2052.98	2.63%
		large	2472.03	6.59%	2472.03	6.59%	2472.03	6.59%	2428.83	4.73%	2387.18	2.93%	2387.18	2.93%
9	2	small	1607.41	0.71%	1607.41	0.71%	1597.10	0.06%	1597.10	0.06%	1597.10	0.06%	1597.10	0.06%
		medium	2726.08	2.95%	2702.56	2.06%	2702.56	2.06%	2702.56	2.06%	2702.56	2.06%	2702.56	2.06%
		large	3343.61	7.10%	3268.01	4.68%	3268.01	4.68%	3268.01	4.68%	3268.01	4.68%	3268.01	4.68%
	4	small	1480.58	8.33%	1480.58	8.33%	1463.77	7.10%	1463.77	7.10%	1463.47	7.07%	1393.81	1.98%
		medium	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%
		large	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%
	7	small	1601.54	13.64%	1534.57	8.89%	1509.99	7.14%	1490.15	5.73%	1490.15	5.73%	1490.15	5.73%
		medium	2517.56	7.32%	2517.56	7.32%	2494.68	6.34%	2494.68	6.34%	2494.68	6.34%	2494.68	6.34%
		large	3830.43	6.56%	3711.65	3.25%	3711.65	3.25%	3711.65	3.25%	3612.92	0.51%	3612.92	0.51%
	9	small	1843.51	6.36%	1843.51	6.36%	1789.81	3.26%	1789.81	3.26%	1789.81	3.26%	1789.81	3.26%
		medium	2816.5	7.21%	2816.50	7.21%	2793.74	6.35%	2793.74	6.35%	2793.74	6.35%	2793.74	6.35%
		large	4117.89	5.83%	4117.89	5.83%	4023.72	3.41%	4023.72	3.41%	4023.72	3.41%	3942.29	1.32%
minimum				0.00%		0.00%		0.00%		0.00%		0.00%		0.00%
average				5.62%		3.98%		2.94%		2.75%		2.48%		2.03%
maximum				20.16%		10.89%		7.30%		7.10%		7.07%		6.35%

Table 5.2: Results for the large instances, 1 hour of computing time

$\lvert P\rvert$	$\hat{m}$	$\bar{s}$	best solution found after											
			60 minutes		120 minutes		180 minutes		240 minutes		300 minutes		360 minutes	
			tc	dev	tc	dev	tc	dev	tc	dev	tc	dev	tc	dev
3	2	small	1080.68	0.51%	1080.68	0.51%	1075.76	0.05%	1075.21	0.00%	1075.21	0.00%	1075.21	0.00%
		medium	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%	1136.31	0.00%
		large	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%	1459.09	0.00%
	3	small	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%	1259.55	0.00%
		medium	1667.96	3.20%	1616.20	0.00%	1616.20	0.00%	1616.20	0.00%	1616.20	0.00%	1616.20	0.00%
		large	1830.68	2.75%	1781.74	0.00%	1781.74	0.00%	1781.74	0.00%	1781.74	0.00%	1781.74	0.00%
6	2	small	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%	1323.69	0.00%
		medium	1844.73	1.15%	1844.73	1.15%	1844.73	1.15%	1844.73	1.15%	1844.73	1.15%	1823.69	0.00%
		large	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%	2396.85	0.00%
	4	small	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%	1112.26	0.00%
		medium	1589.01	3.90%	1589.01	3.90%	1571.95	2.78%	1571.95	2.78%	1529.42	0.00%	1529.42	0.00%
		large	2372.35	3.09%	2354.77	2.33%	2301.20	0.00%	2301.20	0.00%	2301.20	0.00%	2301.20	0.00%
	6	small	1631.44	2.44%	1609.81	1.08%	1592.65	0.00%	1592.65	0.00%	1592.65	0.00%	1592.65	0.00%
		medium	2052.98	2.63%	2052.98	2.63%	2039.76	1.97%	2012.06	0.59%	2000.30	0.00%	2000.30	0.00%
		large	2387.18	2.93%	2368.14	2.11%	2368.14	2.11%	2368.14	2.11%	2319.24	0.00%	2319.24	0.00%
9	2	small	1597.100	0.06%	1596.12	0.00%	1596.12	0.00%	1596.12	0.00%	1596.12	0.00%	1596.12	0.00%
		medium	2702.56	2.06%	2675.37	1.03%	2675.37	1.03%	2661.55	0.51%	2648.00	0.00%	2648.00	0.00%
		large	3268.01	4.68%	3268.01	4.68%	3248.53	4.06%	3121.83	0.00%	3121.83	0.00%	3121.83	0.00%
	4	small	1393.81	1.98%	1393.81	1.98%	1393.81	1.98%	1393.81	1.98%	1393.81	1.98%	1366.79	0.00%
		medium	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%	1541.96	0.00%
		large	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%	1977.32	0.00%
	7	small	1490.15	5.73%	1472.07	4.45%	1409.34	0.00%	1409.34	0.00%	1409.34	0.00%	1409.34	0.00%
		medium	2494.68	6.34%	2464.19	5.04%	2370.69	1.06%	2370.69	1.06%	2345.86	0.00%	2345.86	0.00%
		large	3612.92	0.51%	3612.92	0.51%	3612.92	0.51%	3612.92	0.51%	3594.65	0.00%	3594.65	0.00%
	9	small	1789.81	3.26%	1765.17	1.84%	1759.60	1.52%	1741.41	0.47%	1741.41	0.47%	1733.29	0.00%
		medium	2793.74	6.35%	2709.40	3.13%	2709.40	3.13%	2709.40	3.13%	2705.09	2.97%	2627.05	0.00%
		large	3942.29	1.32%	3942.29	1.32%	3890.97	0.00%	3890.97	0.00%	3890.97	0.00%	3890.97	0.00%
minimum				0.00%		0.00%		0.00%		0.00%		0.00%		0.00%
average				2.03%		1.40%		0.79%		0.53%		0.24%		0.00%
maximum				6.35%		5.04%		4.06%		3.13%		2.97%		0.00%

Table 5.3: Results for the large instances, 6 hours of computing time

Table 5.2 shows that, on average, solutions found after 10 minutes have an objective function value which deviates 5.62% from the best solution found after 360 minutes. After 60 minutes of running time, this deviation is decreased to 2.03% on average. Furthermore, one can observe a decrease in the change of the deviations: Whereas the solutions were improved by 1.63% and 1.04% on average during the second and third 10-minute intervals, respectively, this change decreased to 0.19%, 0.27%, and 0.45% for the remaining three 10-minute intervals. Similar results are shown by Table 5.3: During the second and third 60-minute interval, the average deviation is reduced by 0.64% and 0.61%, respectively. In contrast to this development, the decrease during the last three intervals is only 0.25%, 0.29%, and 0.24%. Figure 5.1 illustrates these findings, showing a decreasing average improvement with increasing computing time. Therefore, we cautiously conclude that the objective function values converge. Regarding different problem parameters, no significant impact on the solution quality can be observed. Therefore, the MS-VNS seems to perform with similar quality for all problem settings.

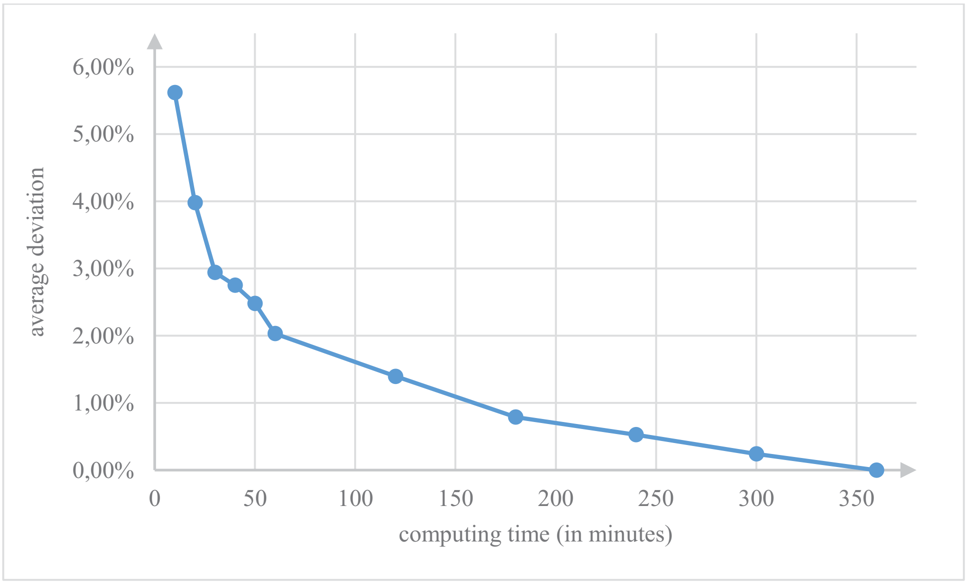

Figure 5.1: Average deviation plotted against computing time

5.5 Experiments with Instances from Practice

In the third set of experiments, we investigated the benefits of introducing vehicles with multiple compartments over vehicles with a single compartment only. For these experiments, we generated instances based on data from practice, which we obtained from a recycling company based in the City of Magdeburg, Germany. The company is responsible for picking up glass waste from containers, serving 127 containers at 44 different locations. Up to three different containers are available at each location, provided for the collection of colorless, green and brown glass. The trucks, which are used for picking up the glass waste at the container locations, possess a single flexible wall that can be introduced at predefined positions such that the total capacity of the truck can be divided into at most two compartments. The position of the wall can be rearranged for each vehicle tour.

The company provided us with a list which – over a period of several weeks – documented for each day which location was visited, which product type (glass color) was picked up, and what the respective supplies were. From this data we were able to estimate the so-called fill rates, i.e. the amount of glass by which a container is filled per day.

Over a planning horizon of 25 days we then deterministically simulated the development of the supplies for each container at each location, assuming that a container is always emptied on the last day before its capacity would be exceeded. The data (i.e. the supplies for each container at each location) that has been obtained by the simulation for each day has then been considered as the basic data for the definition of a specific routing problem instance. These 25 days have been selected for our numerical experiments, giving rise to 25 different problem instances. The distances between the locations and between the depot and the locations were determined by means of a commercial navigation system and taken as the costs of the corresponding edges.

The procedure resulted in instances where the number of container locations varies between 26 and 34 and the number of demands between 34 and 54. The number of product types is three for all instances. The MS-VNS approach was applied to the 25 instances allowing either one or up to two compartments for each vehicle. For both scenarios, all algorithm settings were identical. As termination criteria, $500\mathrm{n}|P|$ iterations without improvement and $\mathrm{n}|P|/6$ loops without improvement were used. The average computing times amounted to 261 seconds for the CVRP-scenario and 452 seconds for the MCVRP-scenario.

Table 5.4 presents the characteristics of each problem instance and the corresponding results from the experiment, namely the number of locations (n) and the total number of supplies ($\overline{\mathrm{s}}$), the total costs of the solutions in case only one compartment can be used on each vehicle (tc.cvrp), the total costs of the solutions if up to two (flexible-sized) compartments can be introduced (tc.mcvrp) and the improvement (impr; in percent) which can be obtained by the latter. Finally, the corresponding number of tours for vehicles with one compartment (tours.cvrp) and with up to two compartments (tours.mcvrp) are shown.

instance	parameters		total costs			number of tours	
	n	$\overline{s}$	tc.cvrp	tc.mcvrp	impr	tours.cvrp	tours.mcvrp
1	32	54	123.83	83.375	32.7%	3	2
2	26	38	117.59	78.054	33.6%	3	2
3	33	44	130.18	91.188	30.0%	3	2
4	33	46	136.24	95.543	29.9%	3	2
5	31	48	149.20	79.975	46.4%	4	2
6	29	34	115.05	77.243	32.9%	3	2
7	34	54	163.42	94.532	42.2%	4	2
8	28	40	132.67	92.671	30.1%	3	2
9	30	44	117.10	78.699	32.8%	3	2
10	31	44	122.70	82.282	32.9%	3	2
11	34	48	160.17	90.26	43.6%	4	2
12	30	36	125.60	85.696	31.8%	3	2
13	32	54	123.83	83.475	32.6%	3	2
14	26	38	117.59	78.054	33.6%	3	2
15	33	44	130.18	91.188	30.0%	3	2
16	33	46	136.24	95.543	29.9%	3	2
17	31	48	149.20	79.975	46.4%	4	2
18	29	34	115.05	77.243	32.9%	3	2
19	34	54	163.42	94.532	42.2%	4	2
20	28	40	132.67	92.671	30.1%	3	2
21	30	44	117.10	78.699	32.8%	3	2
22	31	44	122.70	82.282	32.9%	3	2
23	34	48	160.17	90.26	43.6%	4	2
24	30	36	125.60	85.696	31.8%	3	2
25	32	54	123.83	83.475	32.6%	3	2
minimum	26	34	115.1	77.2	29.9%	3	2
average	31.0	44.6	132.5	85.7	34.8%	3.2	2.0
maximum	34	54	163.4	95.5	46.4%	4	2

Table 5.4: Results for the real-world instances

The results of this set of experiments demonstrate the economic benefits of using vehicles with multiple, flexible-size compartments over vehicles with single compartments. For these problem instances, the costs of the tours could be reduced by 34.8 % on average. Furthermore, the number of tours (or vehicles) necessary on average to pick up all supplies was reduced by one, from 3.2 tours to 2.0 tours.

6 Conclusions

In this paper, a variant of the multi-compartment vehicle routing problem was introduced and studied. This problem is different from the ones previously discussed in the literature. In particular, it includes compartments of flexible sizes, allowing for the number of compartments being smaller than the number of product types, which have to be transported separately. Because of this aspect, an additional question arises concerning the assignment of product types to vehicles.

The main contributions of this paper include (1) a presentation of this new variant of the MCVRP which is inspired by a real-world application; (2) the development of an exact and a heuristic solution procedure for this problem; and (3) extensive numerical experiments investigating the problem itself as well as the performance of both suggested algorithms.

Concerning the exact approach, the results of the experiments have shown that only problem instances of limited size can be solved to optimality in reasonable computing time. Detailed analysis of the results from these experiments provided insights into the impact of different problem parameters: The problem becomes – not unexpectedly – harder to solve with an increasing number of customer locations, product types and supplies. However, it was also shown that the maximum number of compartments, which determines the number of product types per vehicles, has a significant impact on the complexity of the problem. The smaller the number of available compartments is, the harder the problem becomes to solve. This result particularly emphasizes the relevance of the additional assignment decision encountered in this problem, namely the assignment of product types to be transported by each vehicle.

Concerning the heuristic approach, we were able to show that it performs well – producing optimal or near-optimal solutions for small problem instances. For large instances with up to 450 demands, the heuristic finds presumably good quality solutions in reasonable time.

In a third experiment, based on instances from practice, the benefits of using vehicles with multiple compartments of flexible sizes over using vehicles with a single compartment were investigated. The results for this specific application have shown that – if multiple, flexible-size compartments can be introduced – the total costs of the tours necessary for collecting all supplies can be reduced drastically. These results indicate the necessity of dealing with and developing effective solutions approaches for multi-compartment vehicle routing problems.

References

Archetti, C.; Campbell, A.; Speranza, M.G. (2013): Multi-commodity vs. single-commodity routing. In: Transportation Science, article in press.

Avella, P.; Boccia, M.; Sforza, A. (2004): Solving a fuel delivery problem by heuristic and exact approaches. In: European Journal of Operational Research 152, 170-179.

Brown, G.G.; Graves, G.W. (1981): Real-time dispatch of petroleum tank trucks. In: Management Science 27, 19-32.

Caramia, M.; Guerriero, F. (2010): A milk collection problem with incompatibility constraints. In: Interfaces 40, 130-143.

Chajakis, E.D.; Guignard, M. (2003): Scheduling deliveries in vehicles with multiple compartments. In: Journal of Global Optimization 26, 43-78.

Cornillier, F.; Boctor, F.F.; Laporte, G.; Renaud, J. (2008): A heuristic for the multi-period petrol station replenishment problem. In: European Journal of Operational Research 191, 295-305.

Derigs, U.; Gottlieb, J.; Kalkoff, J.; Piesche, M.; Rothlauf, F.; Vogel, U. (2011): Vehicle routing with compartments: applications, modelling and heuristics. In. OR Spektrum 33, 885-914.

Dueck, G.; Scheuer, T. (1990): Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. In: Journal of Computational Physics 90, 161-175.

El Fallahi, A.; Prins, C.; Wolfer Calvo, R. (2008): A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. In: Computers & Operations Research 35, 1725-1741.

Fagerholt, K.; Christiansen, M. (2000): A combined ship scheduling and allocation problem. In: Journal of the Operational Research Society 51, 834-842.

Gendreau, M.; Hertz, A.; Laporte, G. (1994): A tabu search heuristic for the vehicle routing problem. Management Science 40, 1276–1290.

Golden, B.L.; Raghavan, S.; Wasil, E.A. (2008): The vehicle routing problem: latest advances and new challenges. New York: Springer.

Hansen, P.; Mladenović, N. (2001): Variable neighborhood search: principles and applications. In: European Journal of Operational Research 130, 449-467

Helsgaun, K. (2000): An effective implementation of the Lin-Kernighan traveling salesman heuristic. In: European Journal of Operational Research 126, 106-130.

Laporte, G. (2009): Fifty years of vehicle routing. In: Transportation Science 43, 408-416.

Lin, S. (1965): Computer solutions to the traveling salesman problem. Bell Systems Technical Journal 44, 2245-2269.

Lin, S.; Kernighan, B.W. (1973): An effective heuristic algorithm for the traveling-salesman problem. In: Operations Research 21, 498-516.

Mendoza, J.E.; Castanier, B.; Guéret, C; Medaglia, A.L.; Velasco, N. (2010): A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. In: Computers & Operations Research 37, 1886-1898.

Mendoza, J.E.; Castanier, B.; Guéret, C.; Medaglia, A.L.; Velasco, N. (2011): Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. In: Transportation Science 45, 346-363.

Mladenović, N.; Hansen, P. (1997): Variable neighborhood search. In: Computers & Operations Research 24, 207-226.

Muyldermans, L.; Pang, G. (2010): On the benefits of co-collection: experiments with a multi-compartment vehicle routing problem. In: Europan Journal of Operational Research 206, 93-103.

Toth, P.; Vigo, D. (2002): The vehicle routing problem. Philadelphia: Society for Industrial and Applied Mathematics.

Otto von Guericke University Magdeburg
Faculty of Economics and Management
P.O. Box 4120 | 39016 Magdeburg | Germany

Tel.: +49 (0) 3 91 / 67-1 85 84
Fax: +49 (0) 3 91 / 67-1 21 20

www.fww.ovgu.de/femm

ISSN 1615-4274