

AN EXACT APPROACH FOR THE MULTI-COMPARTMENT VEHICLE ROUTING PROBLEM

Marcos Costa Roboredo

Departamento de Engenharia de Produção, Universidade Federal Fluminense - UFF
Rua Passo da Pátria, 156, Campus Praia Vermelha, São Domingos, Niterói, RJ, 24210-240
mcroboredo@id.uff.br

Carlos Leonardo Ramos Póvoa

Universidade Estadual do Norte Fluminense UENF
Av. Alberto Lamego, 2000 - Pq. Califórnia, Campos dos Goytacazes - RJ, 28013-602
clrp@uenf.br

André Soares Velasco, Frederico Galaxe Paes

Instituto Federal de Educação, Ciência e Tecnologia Fluminense IFF
Av. Souza Mota, 350 - Parque Fundão, Campos dos Goytacazes - RJ, 28060-010
asvelasco@ifff.edu.br, fpaes@ifff.edu.br

RESUMO

O Problema de Roteamento de Veículos com Múltiplos Compartimentos (MCVRP) é uma generalização do clássico problema de roteamento de veículos capacitado onde múltiplos compartimentos são usados nos veículos para transportar diferentes produtos que não podem ser misturados. Neste trabalho, nós lidamos com uma variante clássica do MCVRP onde existe um único tipo de veículo que possui um compartimento exclusivo para cada tipo de produto, os tamanhos dos compartimentos são fixos e todas as demandas dos clientes são atendidas por um único veículo. Para a variante estudada, nós propomos pela primeira vez um algoritmo exato, que é um algoritmo branch-and-cut-and-price implementado através do framework VRPSolver. Além disso, nós apresentamos 52 novas instâncias com até 134 clientes. Os resultados mostram que o nosso método resolve 49 das 52 instâncias na otimalidade onde 35 instâncias foram resolvidas no nó raiz.

PALAVRAS CHAVE. Problema de roteamento de veículos, Múltiplos compartimentos, Branch-and-cut-and-price.

ABSTRACT

The Multi-Compartment Vehicle Routing Problem (MCVRP) is a generalization of the classical capacitated vehicle routing problem where multiple compartments are used in the vehicles to transport different types of products that cannot be mixed. In this paper, we deal with a classical MCVRP variant where there is only one type of vehicle that has one exclusive compartment for each type of product, the compartment sizes are fixed and all customer demands are totally served by a single vehicle. For this variant, we present for the first time an exact algorithm that is a branch-and-cut-and-price one coded within the VRPSolver framework. Besides we present new 52 benchmark instances with up to 134 customers. The results show that our method optimally solves 49 instances where 35 of them were solved at the root node.

KEYWORDS. Vehicle routing problem, Multi-Compartment, Branch-and-cut-and-price.

1. Introduction

The Multi-Compartment Vehicle Routing Problem (MCVRP) is a generalization of the widely known Capacitated Vehicle Routing Problem (CVRP). The CVRP only allows the transport of one type of product, or multiple products that can be mixed in a single compartment. In MCVRP, multiple compartments are used to transport different types of products that cannot be mixed. Typical applications include transporting fuels (diesel, gasoline, etc.) to gas stations, transporting food at different storage temperatures and collecting waste of different types (glass, cardboard, plastic). For more details about classifications, models and variants on MCVRPs, we refer to Ostermeier et al. [2020].

In this paper, we deal with a MCVRP variant that considers a set of homogeneous vehicles, a set of products and a set of customers. Each vehicle has exactly one compartment reserved for each type of product. The size of each compartment is fixed and previously known. Each customer can have a different demand for each type of product. The problem objective is to design routes for the vehicles with minimum total cost satisfying the following constraints: all demands of a given customer is served by a single vehicle in a single route and the total demand of a given product served by a given vehicle does not exceed the vehicle compartment size reserved for this product. We refer that variant just as MCVRP.

To the best of our knowledge, there are just heuristic algorithms for the MCVRP as, for example, the algorithms proposed by Fallahi et al. [2008], Mendoza et al. [2010], Abdulkader et al. [2015], Silvestrin e Ritt [2017] and Póvoa et al. [2021]. Naturally, those previous algorithms are applied to large instances (most of them have at least 100 customers). In this context, this paper presents the following contributions for the MCVRP. First, we propose an exact branch-and-cut-and-price (BCP) algorithm that is coded within VRPSolver framework [Pessoa et al., 2020]. Moreover, we propose a set of new 52 benchmark instances whose the number of customers varies from 36 to 134.

In order to show the robustness of our method, we apply it to the 52 proposed instances where the results show that our method optimally solve 49 of 52 instances. Moreover, 35 instances were optimally solved at the root node.

The remainder of this paper is organized as follows. Section 2 presents a literature review about MCVRP and its variants. Section 3 formally defines the MCVRP. Section 4 presents the proposed exact VRPSolver model for the MCVRP. Instances and the results are discussed in Section 5. Finally, Section 6 presents our conclusions and suggestions for future researches.

2. Literature review

The MCVRP appears for the first time at Brown e Graves [1981] and in Brown et al. [1987] where it is described as an application to the fuel distribution problem. Ostermeier et al. [2020] identified a total of 84 publications related to MCVRP and created a typology of the different variations of the problem as well as their applications. The typology developed considers the attributes related to the compartments and the order fulfillment requirements. In relation to compartments, the authors identified three types of attributes: a) flexibility of compartment sizes: fixed (FixS) or flexible (FlexS). b) Assignment of product types to compartments: fixed (FixA) or flexible (FlexA). c) Sharing of compartments: shared (SC) and not shared (UC). Regarding the way to attend consumer demand, two types of attributes were identified: a) number of visits per consumer: visit by a single vehicle (SV) or multiple vehicles (MV). b) Demand fulfillment: a demand of a consumer for a single product can be fulfilled by more than one vehicle (SD) or the demand must be fulfilled by a single vehicle (UD). Regarding the main applications, the authors identified

the following: fuel distribution, collection of recyclable materials, food distribution and maritime transport.

Chajakis e Guignard [2003] studied the problem applied to distribution in convenience stores with vehicles with variable size compartments and developed a solution using Lagrangian relaxation. Avella et al. [2004] describes an application of fuel distribution with a heterogeneous fleet. The authors developed a simple heuristic to solve the problem and a branch and price algorithm. Cornillier et al. [2008] extends the problem to multiple delivery periods and a heterogeneous fleet.

Fallahi et al. [2008] describes the problem on a context of distribution of feed for farm animals. The compartment sizes are fixed and each product can only be allocated in one compartment type. They can not be mixed with any other similar product. A solution for the problem was proposed using memetic algorithms and tabu search. Tasar et al. [2019] extends the problem considering loading and unloading times and developed a variable neighborhood heuristic (VNS).

Muyldermans e Pang [2010] apply MCVRP to waste collection and developed a guided local search (GLS) to solve the problem. Reed et al. (2014) study a similar problem and developed an ant colony algorithm to solve it. Lahyani et al. [2015] present a case study of olive oil distribution, which are categorized into different compartments according to their quality. A variable neighborhood search (VNS) algorithm was developed to solve the problem. Mendoza et al. [2010] solved the problem using a memetic algorithm. The work by Mendoza et al. [2011] solves the MCVRP with stochastic demand, using constructive heuristics. Derigs et al. [2011] developed an integer programming model to solve the problem. In the same article, the authors implemented several heuristic algorithms and compared the results.

Henke et al. [2015] present a version of the problem adapted to the collection of recycled material where the sizes of the compartments can assume pre-defined discrete values. The problem was solved using a variable neighborhood search (VNS) algorithm. The works of Ostermeier e Hübner [2018]; Hübner e Ostermeier [2019] present an extension of the problem, where the capacity of the compartments is flexible and can be adjusted. The authors developed an integer programming model and a large neighborhood search (LNS) heuristic. Chen et al. [2019] applied the MCVRP in a frozen products distribution company, considering the restriction of time windows and fuel consumption. The authors developed a integer programming model and an adaptive large neighborhood search (ALNS) heuristic to solve the problem.

Several heuristics were implemented to solve the MCVRP studied in this work. We can highlight the work of Fallahi et al. [2008] where the first MCVRP instances were proposed. Silvestrin e Ritt [2017] developed an Iterative Tabu Search (ITS). Mendoza et al. [2010] implemented a memetic algorithm and improved some results found by Fallahi et al. [2008]. Abdulkader et al. [2015] proposed a hybridized algorithm which combines local search with ant colony algorithm to solve the problem and a new set of MCVRP instances was generated. Recently, Póvoa et al. [2021] implemented a algorithm that combines GRASP with Tabu Search. In comparison with the works of Fallahi et al. [2008] and Mendoza et al. [2010], the authors improved the results of 10 instances.

3. The problem

According to the typology developed by [Ostermeier et al., 2020], the MCVRP focused on this work is classified as follows: fixed sizes of the compartments (FixS), i.e., we know previously the size of each compartment. The types of products associated with the compartments are fixed (FixA). In other words, the products can only be assigned to specific compartments. The compartments are shared (SC) and can contain the demand of several customers. For the variant focused in this paper, the fleet is homogeneous and each vehicle has a compartment reserved for each type of product. A customer's demand for multiple products in different compartments must be supplied

by a single vehicle (SV) and a consumer's demand for a single product cannot be made by more than one vehicle (UD). In other words, the customer must to be served by a single vehicle during a single route.

The problem is formally defined as follows. Given a complete undirected graph $G = (V, E)$. The vertices set $V = \{0, 1, \dots, n\}$ consists of a depot (node 0) and set of customers $V^+ = \{1, \dots, n\}$. Each edge $e \in E$ has a traverse cost denoted by c_e . The problem also considers a set of product types P . The demand of customer $j \in V^+$ for product type $p \in P$ is given and is denoted by d_{jp} . To serve the customers, there is a set of K homogeneous vehicles where each one of them has $|P|$ compartments, each one reserved for a different product $p \in P$ with a given capacity denoted by Q_p . The MCVRP consists of designing at most K routes with minimal total cost that satisfy the following two constraints. The first one is that each customer is visited exactly once by a single vehicle where all positive demands of this customer are served during that visit. The second one is that the capacity of each vehicle compartment is respected.

In order to illustrate the several concepts described in this paper, we propose a toy MCVRP instance with $n = 3$, $K = 2$, $|P| = 2$ and $Q_1 = Q_2 = 2$. Besides all demands and costs are unitary. The Figure 1 provides a graph representation of the proposed instance. We marked the node 0 in yellow in order to indicate that each route must start and end at this vertex. We put the values 1, 1 close to each vertex associated with a customer in order to indicate the demands for each customer. We put the number 1 on each edge to represent its traverse cost.

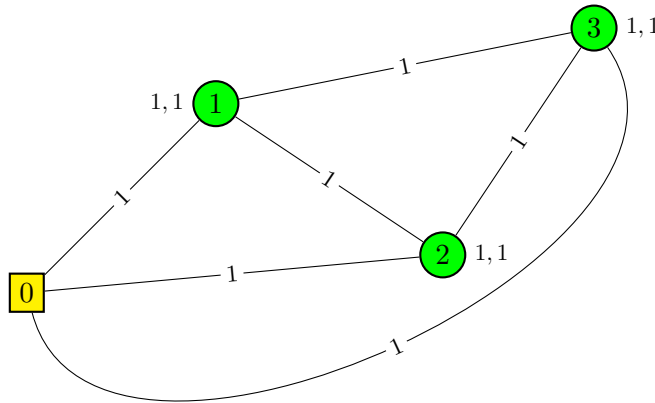


Figure 1: Graph illustration for the proposed MCVRP instance.

Observing the proposed instance, we can note that the following routes are feasible: $r_1 = (0, 1, 0)$, $r_2 = (0, 2, 0)$, $r_3 = (0, 3, 0)$, $r_4 = (0, 1, 2, 0)$, $r_5 = (0, 2, 1, 0)$, $r_6 = (0, 1, 3, 0)$, $r_7 = (0, 3, 1, 0)$, $r_8 = (0, 2, 3, 0)$, $r_9 = (0, 3, 2, 0)$. One optimal solution is to design the routes r_3 and r_4 with total cost $2 + 3 = 5$.

4. A VRPSolver model for the MCVRP

The VRPSolver is a framework proposed by Pessoa et al. [2020] that facilitates the design of BCP algorithms for VRPs and related problems. The framework solves a Mixed Integer Linear Programming (MILP) formulation for the problem through a generic BCP algorithm that generates path variables dynamically by solving the pricing subproblems which are resource constrained shortest path problems (RCSP) over one or more directed graphs.

Now we present the proposed VRPSolver model for the MCVRP. In each step, we indicate which is provided by the user and which is not. First, we present the resource constrained

path generator graph. Second, we present the MILP model and the BCP algorithm used by the framework.

4.1. Graph for RCSP subproblems

Let $G' = (V', A')$ be a directed graph such that $V' = \{v_0, v_1, \dots, v_n\}$ and $A' = \{(v_i, v_j), (v_j, v_i) : e = (i, j) \in E\}$. Let $R = \{1, \dots, |P|\}$ be a set of resources. Let p' be a path starting and ending at the node v_0 . When p' starts, the accumulated consumption value of a given resource $r \in R$ is zero. Every time a node v_i ($0 < i \leq n$) is visited, we increase this value by d_{ir} units. We say that p' is feasible if the accumulated consumption value of a given resource r never exceeds Q_r . All the previous information are provided by us for the VRPSolver framework through specialized functions.

The path generator graph $G' = (V', A')$ for the toy instance proposed in the Section 3 is illustrated by the Figure 2. We marked the node v_0 in yellow in order to indicate that each path must start and end at this vertex. We put two intervals $[0, 2]$ close to each vertex in order to indicate that, for visiting any vertex, the accumulated resource consumption of each one of the two resources must be within that interval. For each arc, we show the consumption of the resources 1 and 2, denoted by respectively q_1 and q_2 .

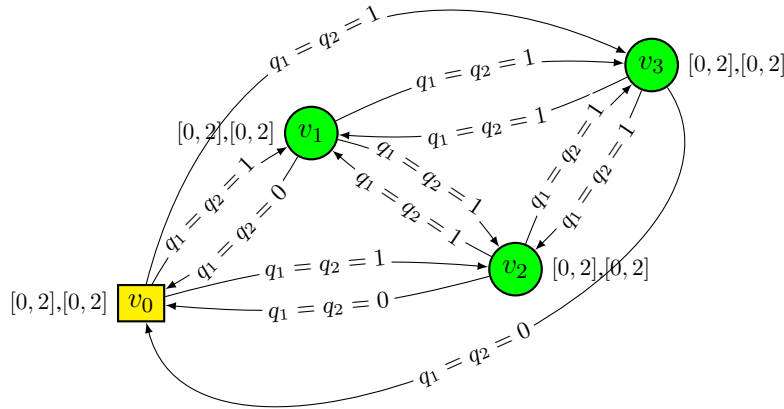


Figure 2: Path generator graph G' for the illustrative MCVRP instance.

Observing the graph in the Figure 2, we can note that the following paths are feasible: $p'_1 = (v_0, v_1, v_0)$, $p'_2 = (v_0, v_2, v_0)$, $p'_3 = (v_0, v_3, v_0)$, $p'_4 = (v_0, v_1, v_2, v_0)$, $p'_5 = (v_0, v_2, v_1, v_0)$, $p'_6 = (v_0, v_1, v_3, v_0)$, $p'_7 = (v_0, v_3, v_1, v_0)$, $p'_8 = (v_0, v_2, v_3, v_0)$, $p'_9 = (v_0, v_3, v_2, v_0)$. Another interesting observation is that if we associate each vertex v_i to the vertex i in the original graph, then each feasible path represents a feasible route of the proposed MCVRP instance and vice-versa.

4.2. MILP formulation

Now we describe the MILP model solved by the VRPSolver framework that uses an integer variable x_e for each edge $e \in E$ that indicates the number of times that the edge e is traversed. These variables are provided by us for the framework. The formulation uses yet a non-negative integer variable λ'_p for each $p' \in P'$ created by the framework, where P' represents the set of feasible paths on G' . Each variable λ'_p indicates how many times the path p' is used at the final solution. The model follows.

$$\text{Min } \sum_{e \in E} c_e x_e \quad (1a)$$

$$\text{S.t. } \sum_{e \in \delta(j)} x_e = 2, \quad j \in V^+; \quad (1b)$$

$$x_e = \sum_{p' \in P'} \left(\sum_{a \in M(x_e)} h_a^{p'} \right) \lambda_{p'}, \quad e \in E; \quad (1c)$$

$$0 \leq \sum_{p' \in P'} \lambda_{p'} \leq K, \quad (1d)$$

$$x_e \in \mathbb{Z}, \quad e \in E \quad (1e)$$

$$\lambda_{p'} \in \mathbb{Z}_+, \quad p' \in P' \quad (1f)$$

The objective function (1a) aims to minimize total cost. The constraints (1b) use the constants $\delta(j)$ that represents the set of edges adjacent to the node j . Those constraints ensure that each customer is visited exactly once. The variables x and the equations (1a) and (1b) are all provided by us for the framework. The constraints (1c) ensure the relation between variables x and λ . Each one of those constraints uses two types of constants: the constants $h_a^{p'}$ that indicates the number of times that an arc a appears in a path p' and the constants $M(x_e)$ that is given by $M(x_e) = \{(v_i, v_j), (v_j, v_i)\}$, where $e = (i, j)$. For the constraints (1c), we only provide for the VRPSolver each mapping M . Basically, the mapping M is provided in a way that the value of a variable x_e is equal to the sum of times that e is traversed considering all the paths given by the variables λ . The constraint (1d) indicates a minimum and a maximum number of feasible paths at the final solution. Both limits are provided.

The formulation (1) applied to the proposed instance is the following:

$$\text{Min } x_{(0,1)} + x_{(0,2)} + x_{(0,3)} + x_{(1,2)} + x_{(1,3)} + x_{(2,3)} \quad (2a)$$

$$\text{S.t. } x_{(0,1)} + x_{(1,2)} + x_{(1,3)} = 2, \quad (2b)$$

$$x_{(0,2)} + x_{(1,2)} + x_{(2,3)} = 2, \quad (2c)$$

$$x_{(0,3)} + x_{(1,3)} + x_{(2,3)} = 2, \quad (2d)$$

$$x_{(0,1)} = 2\lambda_{p'_1} + \lambda_{p'_4} + \lambda_{p'_5} + \lambda_{p'_6} + \lambda_{p'_7} \quad (2e)$$

$$x_{(0,2)} = 2\lambda_{p'_2} + \lambda_{p'_4} + \lambda_{p'_5} + \lambda_{p'_8} + \lambda_{p'_9} \quad (2f)$$

$$x_{(0,3)} = 2\lambda_{p'_3} + \lambda_{p'_6} + \lambda_{p'_7} + \lambda_{p'_8} + \lambda_{p'_9} \quad (2g)$$

$$x_{(1,2)} = \lambda_{p'_4} + \lambda_{p'_5} \quad (2h)$$

$$x_{(1,3)} = \lambda_{p'_6} + \lambda_{p'_7} \quad (2i)$$

$$x_{(2,3)} = \lambda_{p'_8} + \lambda_{p'_9} \quad (2j)$$

$$0 \leq \sum_{k=1}^9 \lambda_{p'_k} \leq K, \quad (2k)$$

$$x_e \in \mathbb{Z}, \quad e \in E \quad (2l)$$

$$\lambda_{p'} \in \mathbb{Z}_+, \quad p' \in P' \quad (2m)$$

The VRPSolver framework uses a generic BCP algorithm to solve the formulation (1). For that, the framework considers the following master formulation which is the linear programming (LP) relaxation (1) without the variables x that are eliminated through the constraints (1c). The LP relaxation follows.

$$\text{Min } \sum_{e \in E} c_e \sum_{p' \in P'} \left(\sum_{a \in M(x_e)} h_a^{p'} \right) \lambda_{p'} \quad (3a)$$

$$\text{S.t. } \sum_{e \in \delta(j)} \sum_{p' \in P'} \left(\sum_{a \in M(x_e)} h_a^{p'} \right) \lambda_{p'} = 2, \quad j \in V^+; \quad (3b)$$

$$0 \leq \sum_{p' \in P'} \lambda_{p'} \leq K, \quad (3c)$$

$$\lambda_{p'} \geq 0, \quad p' \in P' \quad (3d)$$

The master formulation (3) applied to the proposed instance is the following:

$$\text{Min } 2\lambda_{p'_1} + 2\lambda_{p'_2} + 2\lambda_{p'_3} + 3\lambda_{p'_4} + 3\lambda_{p'_5} + 3\lambda_{p'_6} + 3\lambda_{p'_7} + 3\lambda_{p'_8} + 3\lambda_{p'_9} \quad (4a)$$

$$\text{S.t. } 2\lambda_{p'_1} + 2\lambda_{p'_4} + 2\lambda_{p'_5} + 2\lambda_{p'_6} + 2\lambda_{p'_7} = 2, \quad (4b)$$

$$2\lambda_{p'_2} + 2\lambda_{p'_4} + 2\lambda_{p'_5} + 2\lambda_{p'_8} + 2\lambda_{p'_9} = 2, \quad (4c)$$

$$2\lambda_{p'_3} + 2\lambda_{p'_6} + 2\lambda_{p'_7} + 2\lambda_{p'_8} + 2\lambda_{p'_9} = 2, \quad (4d)$$

$$0 \leq \sum_{k=1}^9 \lambda_{p'_k} \leq K, \quad (4e)$$

$$\lambda_{p'} \geq 0, \quad p' \in P' \quad (4f)$$

The master formulation (3) is solved through a iterative procedure based on column generation algorithm. In each iteration, a restrict version of the formulation (3) with a subset of variables λ is solved. Variables λ that are not present in the previous resolution and with negative reduced cost are then added to the restrict master formulation. The framework finds such variables by solving RCSPP over the graph described in the Section 4.1. If there is no variable with negative reduced cost then the current solution is optimal for the master formulation (3).

In every node of the branch-and-bound search tree, the master formulation (3) is solved by the BCP algorithm. Branching on variables x is then used to solve MILP formulation (1) to optimality.

4.3. Using state-of-art BCP elements

The VRPSolver framework allows that the user incorporates state-of-art BCP elements such as *ng*-path relaxation, rounded capacity cuts, limited-memory rank-1 cutting planes, and elementary route enumeration to the generic BCP algorithm. For that, the user must define the so-called *packing sets*, that can be defined on vertices or arcs. Here we present a brief explanation about packing sets on vertices. Let $\mathcal{P}^V \subset 2^{V' \setminus \{v_0\}}$ be a collection of mutually disjoint subsets of $V' \setminus \{v_0\}$. We say that the elements of $\mathcal{P}^V$ are packing sets if there is at least one optimal solution the of formulation (3) satisfying the following constraints:

$$\sum_{p' \in P'} \left(\sum_{v \in S} h_v^{p'} \right) \lambda_{p'} \leq 1, \quad \forall S \in \mathcal{P}^V, \quad (5)$$

where the constants $h_v^{p'}$ indicates the number of times that the vertex v appears in a path p . In other words, for a given packing set $S \in \mathcal{P}^V$, the vertices in S appear in all paths of some optimal solution at most once. For the MCVRP, we set a different packing set for each customer, i.e., $\mathcal{P}^V = \bigcup_{j \in V_+} \{\{v_j\}\}$. For more details about how the framework incorporates the state-of-art elements to the model based on the packing sets provided by the user, we refer to Pessoa et al. [2020].

5. Computational Results

In this section, we present several computational results in order to prove the robustness of the proposed VRPSolver model for the MCVRP. All experiments were performed on a computer with an Intel Core i7-10700 processor with 2.90 GHz and 16 GB of RAM on Ubuntu 18.04.2 LTS operating system. We used the VRPSolver v0.4.1 that can be downloaded at <https://vrpsolver.math.u-bordeaux.fr/>. Each instance is solved using a single thread. CPLEX v12.10 is used as the LP and MILP solver.

5.1. Proposed instances

To the best of our knowledge, there are just heuristic approaches for the MCVRP in the literature and naturally they are tested on large instances (most of them with at least 100 customers). In this context, we decided to generated new benchmark MCVRP instances with different sizes. For this end, we generate MCVRP instances with two compartments from CVRP instances through the two steps proposed by Fallahi et al. [2008]. First, for each customer i , we calculate d_{i1} and d_{i2} through respectively equations $d_{i1} = d_i/k$ and $d_{i2} = d_i - d_{i1}$, where k is a random integer in $[3, 5]$ and d_i is the the customer i demand in the CVRP instance. All resulting demands are real numbers, i.e., they are not rounded to the closest integer. Second, we calculate Q_1 and Q_2 through respectively equations $Q_1 = (Q * a_1)/(a_1 + a_2)$ and $Q_2 = (Q * a_2)/(a_1 + a_2)$, where Q is the vehicle capacity in the CVRP instance while a_1 and a_2 are the average of the demands of respectively products 1 and 2 considering all customers.

We applied the two steps described before to the CVRP instances with more than 35 customers belonging to classes A, B, E and F. These classes are widely used for testing CVRP approaches and for more details about them, we refer to Uchoa et al. [2017]. In addition, we ignore the number of available vehicles present in the third term of each CVRP instance name. In other words, the solutions generated by our algorithm can use as many vehicles as needed. In order to guarantee the previous characteristic, we set $K = |V^+|$ for all instances. The resulting 52 MCVRP instances can be downloaded at <http://www.loggeo.org/mcvrp.zip>.

Each one of the resulting 52 proposed instances contain the x and y coordinates of each node. For each edge $e = (i, j) \in E$, the cost c_e is given by the euclidean distance between the nodes i and j . Each one of those distances is rounded to the closest integer.

5.2. Results for the proposed instances

Table 1 presents the computational results of the proposed BCP algorithm over the 52 generated MCVRP instances. The following information are presented in that table for each instance. The first information is the instance name (column Instance). We include the term "MC-" at the beginning of each original CVRP instance name in order to indicate that they were adapted to the MCVRP through the steps described in the Section 5.1. The second information is a valid upper bound for each instance that is given by the heuristic proposed by Póvoa et al. [2021] (column ub_0).

Each bound was provided by the authors of the previous paper and it enhances the computational performance of the proposed BCP algorithm. The third information is the best bound found by the proposed BCP algorithm (column ub^*). The fourth information is the relative percentage difference between the bound at the BCP tree root node and the best solution found for each instance (column Root gap(%)). Finally, the fifth information is the total CPU time in seconds consumed by the proposed BCP algorithm for each instance (column Time(s)). We stopped any experiment if the total consumed time reaches 18000 seconds.

Instance	Root				Instance	Root			
	ub_0	ub^*	gap(%)	Time(s)		ub_0	ub^*	gap(%)	Time(s)
MC-A-n37-k5	669	669	0.00	1.1	MC-B-n44-k7	970	956	0.00	11.84
MC-A-n37-k6	977	977	0.00	3.0	MC-B-n45-k5	753	753	0.00	5.15
MC-A-n38-k5	749	749	0.00	1.6	MC-B-n45-k6	715	715	2.52	18000.00
MC-A-n39-k5	832	827	0.00	7.6	MC-B-n50-k7	749	749	0.00	8.19
MC-A-n39-k6	843	843	0.00	1.5	MC-B-n50-k8	1332	1328	0.38	161.9
MC-A-n44-k6	949	949	0.00	1.2	MC-B-n51-k7	1059	1059	0.00	13.45
MC-A-n45-k6	971	971	0.00	1.2	MC-B-n52-k7	762	754	0.00	7.66
MC-A-n45-k7	1158	1158	0.00	2.8	MC-B-n56-k7	722	722	0.00	8.98
MC-A-n46-k7	958	950	0.00	5.4	MC-B-n57-k7	1149	1148	0.00	49.97
MC-A-n48-k7	1109	1101	0.00	11.7	MC-B-n57-k9	1646	1620	0.00	20.12
MC-A-n53-k7	1039	1035	0.00	7.6	MC-B-n63-k10	1578	1554	0.58	1027.28
MC-A-n54-k7	1187	1187	0.00	5.6	MC-B-n64-k9	910	908	0.88	1168.65
MC-A-n55-k9	1129	1126	0.00	3.5	MC-B-n66-k9	1359	1331	0.23	187.89
MC-A-n60-k9	1416	1397	0.00	66.3	MC-B-n67-k10	1081	1078	1.86	18000.00
MC-A-n61-k9	1079	1076	0.00	14.6	MC-B-n68-k9	1311	1294	0.00	87.01
MC-A-n62-k8	1347	1327	0.30	540.2	MC-B-n78-k10	1299	1257	0.16	1788.4
MC-A-n63-k9	1691	1664	0.24	252.4	MC-E-n51-k5	522	522	0.00	1.78
MC-A-n63-k10	1348	1333	0.00	3.2	MC-E-n76-k7	688	687	0.00	37.29
MC-A-n64-k9	1477	1445	0.35	368.7	MC-E-n76-k8	762	746	0.13	724.92
MC-A-n65-k9	1242	1226	0.57	197.3	MC-E-n76-k10	858	845	0.47	466.96
MC-A-n69-k9	1204	1192	0.08	30.8	MC-E-n76-k14	1060	1044	0.00	25.35
MC-A-n80-k10	1823	1779	0.00	200.5	MC-E-n101-k8	827	820	0.00	184.95
MC-B-n38-k6	817	817	0.00	6.9	MC-E-n101-k14	1135	1103	0.36	1005.96
MC-B-n39-k5	568	568	0.00	4.6	MC-F-n45-k4	740	740	0.00	16.53
MC-B-n41-k6	850	850	0.00	5.2	MC-F-n72-k4	250	249	2.01	8079.25
MC-B-n43-k6	746	746	0.00	8.6	MC-F-n135-k7	1242	1242	7.00	18000.00
Global Average						1031.87	1022.81	0.35	1362.36

Table 1: Computational results for the generated 52 MCVRP instances

Observing the Table 1, we can note that the root gaps generated by the proposed BCP algorithm are equal to zero for 35 of 52 instances. In these cases, there was no need to do branch over any variable. Besides the VRPSolver could not optimally solve just three instances within the time limit. The good performance of the proposed exact method is also due to the goods upper bounds provided by the heuristic proposed by Póvoa et al. [2021] (column ub_0 of the Table 1). The closer the value of ub_0 is to the optimal solution, the shorter the computational time required by the proposed BCP algorithm to find the optimal solution.

6. Conclusions

In this paper, we propose for the first time an exact approach for the MCVRP. That approach is a BCP algorithm that was coded within the VRPSolver framework [Pessoa et al., 2020]. Thus, the proposed algorithm contains several state-of-art BCP components such as limited-memory rank-1 cuts, reduced cost-based arc elimination, enumeration of elementary routes, hierarchical strong branching, etc. In addition the previous algorithm, we proposed 52 instances generated from classical CVRP instances. The number of customers in the proposed instances varies from 36 to 134.

The computational results showed that our method was able to solve 49 of 52 instances within 18000 seconds of execution. Besides the proposed BCP algorithm obtained a root gap equal to zero for 35 instances. The heuristic (GRASP/TS) proposed by Póvoa et al. [2021] found good upper bounds, increasing the computational performance of the proposed exact algorithm and reached 20 optimal results.

For the future, we intend to apply our method to the instances used by the papers that propose heuristic approaches for the MCVRP. Another natural research direction is to adapt our model to encompass other characteristics such as flexible compartment sizes and multiple visits to each customer.

References

- Abdulkader, M. M., Gajpal, Y., e Elmekawy, T. Y. (2015). Hybridized ant colony algorithm for the Multi Compartment Vehicle Routing Problem. *Applied Soft Computing Journal*, 37:196–203. ISSN 15684946.
- Avella, P., Boccia, M., e Sforza, A. (2004). Solving a fuel delivery problem by heuristic and exact approaches. *European Journal of Operational Research*, 152(1):170 – 179. ISSN 0377-2217.
- Brown, G. e Graves, G. W. (1981). Real-time dispatch of petroleum tank trucks. *Management Science*, 27:19 – 32.
- Brown, G. G., Ellisa, C. J., Graves, G. W., e Ronen, D. (1987). Real-time, wide area dispatch of mobil tank trucks. *INFORMS Journal on Applied Analytics*, 17(1):107 – 120.
- Chajakis, E. e Guignard, M. (2003). Deliveries in vehicles with multiple compartments. *Journal of Global Optimization*, 26:43 – 78.
- Chen, L., Liu, Y., e Langevin, A. (2019). A multi-compartment vehicle routing problem in cold-chain distribution. *Computers & Operations Research*, 111:58–66. ISSN 0305-0548.
- Cornillier, F., Bector, F. F., Laporte, G., e Renaud, J. (2008). A heuristic for the multi-period petrol station replenishment problem. *European Journal of Operational Research*, 191(2):295 – 305. ISSN 0377-2217.
- Derigs, U., Gottlieb, J., Kalkoff, J., Piesche, M., Rothlauf, F., e Vogel, U. (2011). Vehicle routing with compartments: Applications, modelling and heuristics. *OR Spectrum*, 33(4):885–914.
- Fallahi, A. E., Prins, C., e Wolfler Calvo, R. (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers and Operations Research*, 35(5): 1725–1741. ISSN 03050548.

- Henke, T., Speranza, M. G., e Wäscher, G. (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research*, 246(3):730–743. ISSN 03772217.
- Hübner, A. e Ostermeier, M. (2019). A multi-compartment vehicle routing problem with loading and unloading costs. *Transportation Science*, 53(1):282–300. ISSN 15265447.
- Lahyani, R., Laporte, G., Coelho, L. C., e Khemakhem, M. (2015). A Multi-Compartment Vehicle Routing Problem Arising in the Collection of Olive Oil in Tunisia. *Omega*, 51(March):1–10.
- Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., e Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers and Operations Research*, 37(11):1886–1898. ISSN 03050548.
- Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., e Velasco, N. (2011). Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363. ISSN 15265447.
- Muyldermans, L. e Pang, G. (2010). On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm. *European Journal of Operational Research*, 206(1): 93–103. ISSN 03772217.
- Ostermeier, M., Henke, T., Hübner, A., e Wäscher, G. (2020). Multi-compartment vehicle routing problems: State-of-the-art, modeling framework and future directions. *European Journal of Operational Research*. ISSN 0377-2217.
- Ostermeier, M. e Hübner, A. (2018). Vehicle selection for a multi-compartment vehicle routing problem. *European Journal of Operational Research*, 269(2):682–694. ISSN 03772217.
- Pessoa, A., Sadykov, R., Uchoa, E., e Vanderbeck, F. (2020). A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183(1):483–523.
- Póvoa, C. L. R., Velasco, A. S., Paes, F. G., e Silva, P. d. C. d. P. (2021). Problema de roteirização de ordens de serviço com frota limitada e heterogenea. In *Anais do LIII Simpósio Brasileiro de Pesquisa Operacional*.
- Silvestrin, P. V. e Ritt, M. (2017). An Iterated Tabu Search for the Multi-compartment Vehicle Routing Problem. *Computers & Operations Research*, 81:192–202. ISSN 03050548.
- Tasar, B., Eliyi, D. T., e Kandiller, L. (2019). Vehicle routing with compartments under product incompatibility constraints. *Promet - Traffic&Transportation*, 31(1):25 – 36.
- Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., e Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858.