A Tabu search for the multi-compartment vehicle routing problem

Paulo Vitor Silvestrin
Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
pvsilvestrin@inf.ufrgs.br

Marcus Ritt
Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
mrpritt@inf.ufrgs.br

ABSTRACT

The multi-compartment vehicle routing problem (MC-VRP) consists in finding routes with minimum total length that satisfy the demands of a set of customers for several products that must be loaded in specific vehicle compartments. Despite its wide practical applicability this problems has been rarely studied in the literature. In this paper we propose a constructive heuristic and a Tabu search algorithm for this problem. We evaluate the algorithm on twenty different VRP instances and four different sets of MC-VRP instances with two compartments and compare with previous results from the literature.

KEYWORDS. Vehicle routing problem. Multiple compartments. Tabu search.

MH – Metaheuristics

RESUMO

O problema de roteamento de veículos com multi-compartimentos (MC-VRP) consiste em encontrar rotas com distância total mínima que satisfazem as demandas por vários produtos de um conjunto de clientes onde produtos devem ser alocados em compartimentos distintos. Este problema, apesar de ter uma ampla aplicabilidade em casos reais, é pouco estudado na literatura. Este artigo apresenta uma heurística construtiva e uma busca tabu para este problema. Os algoritmos foram testados com vinte diferentes instâncias do VRP que geraram quatro conjuntos diferentes de instancias MC-VRP com dois compartimentos e comparados com os resultados da literatura.

PALAVRAS CHAVE. Problema de roteamento de veículos. Busca Tabu. Múltiplos compartimentos.

MH – Metaheurísticas

1. Introduction

Vehicle routing problems (VRPs) are concerned with finding shortest routes for a fleet of vehicles in order to attend the demands of a set of customers. Numerous variants of this problem have been studied in the literature. They include VRPs with single or multiple depots, with pickup and delivery or backhauls, with time windows or multiple visits of the customers, different types of capacities, e.g. with loading constraints, etc. For a good overview we refer the reader to the surveys of Laporte (2009) and Vidal et al. (2013).

In this paper we are concerned with a single depot vehicle routing problem for multiple product types given a homogeneous fleet. Each customer may have a different demand for each product type, and the vehicles have multiple compartments of different sizes for them. As in the standard vehicle routing problem, the aim is to satisfy the demand of each customer, visiting it only once, such that the total travel-time of all vehicles is minimized. This problem is called the *multi-compartment vehicle routing problem* (MC-VRP) in the literature. The MC-VRP generalizes the $\mathcal{NP}$-hard capacitated VRP, and thus is also $\mathcal{NP}$-hard. State-of-the-art exact approaches for basic VRP problems are limited to about 100 customers. Thus larger problems, or problems with additional constraints are usually solved by heuristic algorithms, which are able to find solutions of less than 1% above optimality for instances containing about 400 customers in an hour (Vidal et al. 2013).

The MC-VRP has several real-world applications where products must be transported in different compartments for some specific reason. Dairies often use vehicles with multiple compartments to collect milk of different types, e.g., from cows and goats, and different qualities, e.g., from different suckling dates (Mendoza et al. 2010). Petroleum companies deliver different types of fuel to outlet retailers using multi-compartment tankers and multi-compartment vehicles (Cornillier et al. 2012; Benantar and Ouafi 2012), public utilities use trucks with compartments to perform selective waste collection (Reed et al. 2014), and food companies distribute in compartmentalized vehicles groceries that require different levels of refrigeration.

Although there exist several real-world applications of the MC-VRP in industry, the problem is not widely studied. Avella et al. (2004) have presented a heuristic and an exact approach to solve a real world fuel delivery problem with a fleet of vehicles with several tanks of different capacity. Fallahi et al. (2008) studied an MC-VRP applied to the distribution of cattle food to farms, where the different feeds are kept separate to avoid contamination. They proposed a memetic algorithm and a Tabu search to find good solutions. Reed et al. (2014) present a basic CVRP applied to the collection of domestic waste, and also a MC-VRP in the case where the vehicle crew must perform kerbside sorting of the waste in customers' recycling boxes.

Variants of the MC-VRP have been also considered. Cornillier et al. (2012) studied a MC-VRP with multiple depots and time windows applied to petrol station replenishment, and Mendoza et al. (2010) have introduced stochastic demands, i.e. the exact value of demands is not known at the moment when the routes are planned, to obtain the MC-VRPSD.

The remainder of this paper is structured as follows. In next section we formally define the problem and present a mathematical formulation. Section 3 presents a savings method and Tabu search to solve the problem. Section 4 presents computational results and compares them to related works. Finally, we conclude and discuss future work in Section 5.

2. Problem Definition

The MC-VRP is a variation of the classical VRP where the fleet consists of identical vehicles with compartments and the customers have demands for multiple different products. We are given a set of locations $V = \{V_0, V_1, \ldots, V_n\}$, where V_0 is the *depot*, and $V_1, \ldots, V_n$ are *customers*. Each pair of locations $i, j \in V$ has a travel time d_{ij} and each customer may have additionally a drop time t_i, i.e. the time needed to deliver the product. Travel times are assumed to be symmetric ($d_{ij} = d_{ji}$). There are m different types of products, and we have a homogeneous fleet of vehicles, each with m compartments and a capacity $C \in \mathbb{R}^m$. Each client $i \in [n]$[1] has a demand $c_i \in \mathbb{R}^m$, and we assume $c_i \leq C$. A valid route of a vehicle starts and ends at the depot and visits a number of customers. There is no constraint on the number of visited customers per route. A route is represented by an ordered subset $R = \{r_1 = V_0, \ldots, r_{l(R)}\}$ of V of length $l(R)$. The total time of a route is $d(R) = d_{r_{l(R)},r_1} + \sum_{1 \leq i < l(R)} d_{r_i,r_{i+1}} + \sum_{i \in [l(R)]} t_i$, and its demand is $c(R) = \sum_{i \in [l(R)]} c_{r_i}$.

We want to find a set of valid routes $R = \{R_1, \ldots, R_r\}$ that partition the set of customers, $R_i \cap R_j = \{V_0\}$ for all $i, j \in V$, and $\cup_{i \in [k]} R_i = V$, satisfying the capacity constraints $c(R_i) \leq C$, and such that the total time $d(R) = \sum_{i \in [k]} d(R_i)$ is minimized. The total time travelled by each vehicle must not exceed a maximum time D. There is no limit on the number of routes. For $m = 1$ the problem reduces to the standard CVRP.

In our definition of MC-VRP a vehicle is not allowed to visit a customer twice, thus the demand of a customer must be attended in one visit. This definition of the problem is also used in Reed et al. (2014). In Fallahi et al. (2008) the authors have tested a similar scenario but the focus of their algorithm is a variant where the demand of each product type may be satisfied by different vehicles, i.e. a customer may be visited up to m times. Avella et al. (2004) has applied MC-VRP in a real case of fuel delivery where compartments must be completely unloaded when attending a client demand, i.e. compartments travel only full or totally empty.

The MC-VRP can be formulated as follows. Let x_{ijk} indicate that vehicle k travels from $i \in V$ to $j \in V$. Then we want to

$$\textbf{minimize} \quad \sum_{i,j \in V} \sum_{k \in [r]} (d_{ij} + t_j) x_{ijk}, \tag{1}$$

$$\textbf{subject to} \quad \sum_{i \in V} \sum_{k \in [r]} x_{ijk} = 1, \qquad \forall j \in V \setminus \{V_0\}, \tag{2}$$

$$\sum_{i \in V} x_{ijk} = \sum_{i \in V} x_{jik}, \qquad \forall j \in V, k \in [r], \tag{3}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1, \qquad \forall S \subseteq V \setminus \{V_0\}, |S| \geq 2, k \in [r], \tag{4}$$

$$\sum_{i,j \in V} c_j x_{ijk} \leq C, \qquad \forall k \in [p], \tag{5}$$

$$\sum_{i,j \in V} d_{ij} x_{ijk} \leq D, \qquad \forall k \in [p], \tag{6}$$

$$x_{ijk} \in \{0, 1\}, \qquad \forall i, j \in V, k \in [r]. \tag{7}$$

In this formulation we minimize the total travel time (1). By constraint (2) every customer has to be attended exactly once in some route. Constraint (3) establishes flow conservation, and constraint (4) eliminates subroutes that do not include the depot. The capacity and total length constraints are guaranteed by (5) and (6). Note that constraint (6) is vector-valued and will be expanded into m separate constraints, one for each product type. Solving this model directly is unpractical due to the exponential number of contraints (4).

3. A Tabu search for the MC-VRP

We propose a Tabu search to solve the MC-VRP. We construct an initial solution by a modified version of the savings method of Clarke and Wright (1964). The following subsection presents our constructive heuristic and in Section 3.2 we present the Tabu search in detail.

[1] We use the notation $[n] = \{1, 2, \ldots, n\}$.

3.1. A savings method for the MC-VRP

To generate an initial solution we modified the savings heuristic proposed by Clarke and Wright (1964). This heuristic is widely used in VRP problems because of simplicity, speed and it often obtains good results (see Cordeau et al. (2002)). Consider a route which visits customer V_i last, and another route which visits customer V_j first. We can join these routes by going directly from V_i to V_j. This results in *savings* of $s_{ij} = d_{V_i,V_0} + d_{V_0,V_j} - d_{V_i,V_j}$. By symmetry of the travel times, we can also join routes by other endpoints. The heuristic of Clarke and Wright determines the savings s_{ij} for each pair of customer V_i and V_j, and sorts them in non-increasing order. Then, the algorithm creates one route for each customer, starting at the depot, visiting only this customer, and then returning to the depot. Finally, it visits the savings list in the sorted order cyclically, and repeatedly applies feasible joins, until no such join is possible. A join is feasible for a saving s_{ij} if two routes with endpoints V_i and V_j exist.

The generalization to the multi-compartment and time-restricted case is straightforward. We consider a join only feasible if the combined route still satisfies the time and capacity constraints. A initial solution of good quality has shown experimentally to be important to get better final results for the problem.

3.2. Tabu search

The Tabu search meta-heuristic has been proposed by Glover and is a heuristic based on modification of a solution (see Glover and Laguna (1997)). For a search space S and a neighborhood function $N : S \rightarrow 2^S$ it starts from some initial solution, repeatedly passes from the current solution $s \in S$ to a neighboring solution $s' \in N(s)$ until some stopping criterion is satisfied. Similar to local search, Tabu search chooses a neighbor of better objective function value, until no such neighbor exists. In standard Tabu search, one of the best such neighbors is chosen. Otherwise, the best neighbor which has not been declared tabu is chosen. The tabu mechanism is a short-term memory designed to avoid cycling in local minima and to diversify the search. Commonly, some attributes of recently visited solutions are declared tabu for a number of steps, called the *tabu tenure*, and a solution is considered tabu if it has some tabu attribute. Attributes may be elements of solutions, e.g. an arc visited by some vehicle in a solution of a VRP, or complete solutions. Tabu search also frequently includes so-called *aspiration criteria*, i.e. rules that allow neighboring solutions to be chosen even if they are tabu. After stopping, Tabu search returns the best found solution during the search.

3.2.1. Neighborhoods and tabu mechanism

We use four different neighborhoods in our Tabu search. They are defined in terms of *moves types*, i.e. modifications of the current solution to obtain some neighboring solution. A *shift move* removes some customer from his current route, and inserts him into an arbitrary position in some other route; a *swap move* selects two customers in different routes, and exchanges their positions, i.e. the first customer is inserted into the second route in place of the second customer and *vice versa*. A *crossover move* selects two customers in two different routes, and combines the initial and final parts of the routes to obtain two new routes. For routes $R = \{r_1, r_2, \ldots, r_{l(R)}\}$ and $S = \{s_1, s_2, \ldots, s_{l(S)}\}$ selecting customers r_i and s_j produces new routes $R' = \{r_1, \ldots, r_{i-1}, s_j, \ldots, s_{l(S)}\}$ and $S' = \{s_1, \ldots, s_{j-1}, r_i, \ldots, r_{l(R)}\}$. Finally a *route swap move* selects two customers in a route and swaps their positions.

The Tabu search examines all moves in the presented order (shift,swap,crossover,and route swap). Within a move category, routes are always visited in a random order, and customers always in order of the route. We consider only feasible solutions that respect the capacity and length constraints. The number of examined route swap moves has been limited to $\min\{n^2/4, 250\}$. The search adopts a first improvement strategy, accepting the first non-tabu neighbor which is better than the current solution, or the best non-tabu neighbor, if no better one exists. Ties among several best neighbors are broken in favour of the first best neighbor. The only aspiration criterion is to accept tabu solutions that improve the incumbent.

To define the tabu rules, we consider a given customer being part of some route as a solution attribute. For any of the move types, a client that has been moved from some source route is prohibited to return to that route during the tabu tenure. In some preliminary experiments we have fixed the tabu tenure at 15 steps.

Figure 1 shows a pseudo code of the proposed Tabu search.

Algorithm 1: Tabu search Pseudocode

```
Data: current solution $s$
/* initialize the best solution with current                    */
$s^* = s$;
while timeout do
    /* initialize best neighbor as worst possible
       solution                                                  */
    $N' = worstpossiblesolution$;
    /* flag to stop neighbor search when a neighbor
       better then $C$ is found                                  */
    $improved = false$;
    if $!improved$ then
        $improved =$ try All $ShiftMoves(S', C, N')$;
    end
    if $!improved$ then
        $improved =$ try All $SwapMoves(S', C, N')$;
    end
    if $!improved$ then
        $improved =$ try All $CrossOverMoves(S', C, N')$;
    end
    if $!improved$ then
        $maxMoves = min(n^2/4, 250)$;
        while $!maxMoves$ & $!improved$ do
            $improved = RouteSwapMove(S', C, N')$;
        end
    end
    update $TabuList$ with most recent move;
    if $N' < S'$ then
        $S' = N'$;
    end
    $C = N'$;
end
return $S'$
```

4. Computational Results

The Tabu search has been implemented in C++ and tested on a PC with an AMD FX-8150 Eight-Core processor running at 3.4 GHz, and with 32 GB of main memory. For the tests only one core has been used. The algorithms were tested with classical VRP instances and multiple compartments generated from existing VRP instances since we were not able to find publicly available MC-VRP instances. This section describes how the four sets of instances was generated then it shows the obtained results and compare against Fallahi et al. (2008) and Reed et al. (2014), which are, to our knowledge, the only publications which address the same problem.

Table 1: Characteristics of the VRP instances used in the computational experiments.

Name	n	D	drop time	C
vrpnc1	50	∞	0	160
vrpnc2	75	∞	0	140
vrpnc3	100	∞	0	200
vrpnc4	150	∞	0	200
vrpnc5	199	∞	0	200
vrpnc6	50	200	10	160
vrpnc7	75	160	10	140
vrpnc8	100	230	10	200
vrpnc9	150	200	10	200
vrpnc10	199	200	10	200
vrpnc11	120	∞	0	200
vrpnc12	100	∞	0	200
vrpnc13	120	720	50	200
vrpnc14	100	1040	90	200
E072-04f	71	-	-	30000
E076-07u	75	-	-	220
E076-08s	75	-	-	180
E135-07f	134	-	-	2210
E241-22k	240	-	-	200
E484-19k	483	-	-	1000

4.1. Sets of instances

We have used a set of 20 well-known VRP instances in our test. The first 14 (vrpnc1–vrpnc14) come from Christofides and the last six from Eilon. Details about the instances can be found in Table 1. The table shows the number of customers (column "n"), the time limit for a route ("column D"), the drop time (column "drop time") and the capacity of the vehicles (column "C"). From these instances we generated four test sets for the MC-VRP (S1, S2, S3, and S4) as follows.

The set of instances S1 are the original instances from Christofides and Eilon with only one compartment. They are used to validate the performance of our algorithm with the best known solutions of a standard scenario. The set of instances S2 and S3 are generated as described in Reed et al. (2014). The set S2 is obtained by splitting the vehicle capacity into two compartments using a 3:1 ratio. The customer demands are obtained by a similar splitting: all demands are split using a 3:1 ratio, except the demands on a limited subregion ($0 < x, y < 35$), which are split using a 2:1 ratio. The set S3 is obtained in the same way, but splitting vehicle compartments and customer demands using a 4:1 ratio, except for region mentioned above, which maintains a 2:1 ratio.

The set of instances S4 was generated to be able to compare our approach to that of Fallahi et al. (2008), which use two different sets of instances. The first is obtained by dividing the capacity of each vehicle and the demands into two equal parts and the second by dividing randomly each customer demand in two parts. The first does not apply to our case because the compartments will be occupied in the same proportion and we will fall back to the single-compartment case.

Thus we focus on the second set of instances proposed by Fallahi et al. (2008). They are generated as follows. For each customer $i \in V \setminus \{V_0\}$, the demand for the first product is $c_{i1} = c_i/k$, where k is a random integer in $\{3, 4, 5\}$ and c_i is the demand of the corresponding VRP. The demand for the second product is $c_{i2} = c_i - c_{i1}$. To define the capacity of the compartments of the vehicles, let $\overline{C}_1$ be the average demand for the first product, and $\overline{C}_2$ the average demand for the second product. Then the capacity of the compartments is set to

$$C_1 = \frac{C\overline{C}_1}{\overline{C}_1 + \overline{C}_2}; \qquad C_2 = \frac{C\overline{C}_2}{\overline{C}_1 + \overline{C}_2}.$$

4.2. Analysis of the results

The results obtained in our tests are reported in Tables 2 and 3. Table 2 shows the results obtained on instances sets S1, S2, and S3. Each instance of the set was executed ten times with the same parameters and a different random seed. We present for each instance the best known value of the VRP case (column "BKV") and the solution obtained by the constructive method of Clarke and Wright (column "C/W"). For the Tabu search we report the average relative deviation from the best known value (column "TS"), the average time in seconds to find the best solution (column "T (s)") and the relative deviation of the best solution in all ten replications from the best known value (column "Best"). The results have been obtained with a time limit of $n^2/100$ seconds, where n is the number of customers of the instance.

Table 2: Results of the constructive heuristic and the Tabu search on instance sets S1, S2, and S3 compared to best known values of the VRP.

		S1				S2				S3			
Name	BKV	CW	TS	T (s)	Best	C/W	TS	T (s)	Best	C/W	TS	T (s)	Best
vrpnc1	524.6	11.4	2.1	10.0	0.6	18.8	5.6	9.7	5.0	17.8	6.2	13.3	4.8
vrpnc2	835.3	8.6	6.8	24.4	5.4	10.2	7.6	27.7	5.7	13.2	8.8	25.8	8.0
vrpnc3	826.1	7.6	4.0	68.6	2.0	10.8	8.8	58.5	6.9	13.0	7.1	71.4	6.6
vrpnc4	1028.4	10.9	6.1	144.9	4.8	17.4	12.9	169.5	10.9	18.6	13.5	182.9	11.6
vrpnc5	1291.4	8.1	6.8	265.8	6.2	13.2	12.4	300.1	12.1	16.2	14.0	325.9	13.6
vrpnc6	555.4	11.3	1.3	14.6	0.6	10.9	5.5	14.6	4.0	10.9	4.1	10.5	1.2
vrpnc7	909.7	7.2	4.4	35.1	3.4	7.0	4.8	33.2	3.0	7.4	5.7	21.8	4.5
vrpnc8	865.9	12.5	5.2	75.8	3.6	15.0	8.8	70.1	6.0	15.0	8.0	70.6	5.2
vrpnc9	1162.5	10.8	7.0	191.0	5.6	14.0	10.9	121.7	9.3	12.7	8.9	167.1	6.6
vrpnc10	1395.8	10.2	7.8	266.4	6.7	13.9	10.6	280.4	9.0	13.9	11.3	318.2	10.5
vrpnc11	1042.1	2.5	2.5	0.0	2.5	7.0	6.4	72.8	6.0	23.0	20.3	108.3	16.0
vrpnc12	819.6	1.7	0.9	2.2	0.9	12.2	8.4	70.2	6.7	19.7	17.2	62.7	16.6
vrpnc13	1541.1	3.3	1.5	75.7	1.0	3.3	1.4	84.6	1.1	3.3	1.5	68.6	1.4
vrpnc14	866.4	1.1	0.9	10.7	0.8	6.4	4.7	30.7	4.5	16.6	12.8	52.2	12.4
E072-04f	241.9	5.9	4.2	15.6	2.2	11.5	9.4	40.4	8.5	9.6	9.6	2.6	9.2
E076-07u	690.8	6.9	2.9	20.4	2.2	6.3	4.0	34.3	2.9	11.0	5.7	36.5	4.6
E076-08s	742.6	7.0	2.9	28.6	1.8	9.6	7.1	20.9	5.5	12.3	8.4	36.6	5.6
E135-07f	1162.9	4.8	2.7	101.6	2.5	6.0	5.0	82.9	5.0	14.7	13.8	100.3	13.8
E241-22k	666.8	14.8	13.4	421.0	12.9	23.1	22.0	449.7	21.5	26.7	24.4	485.8	23.7
E484-19k	1137.2	11.8	8.9	2056.6	8.6	17.6	16.1	2117.0	15.9	17.4	13.8	2023.4	13.3
Average	915.3	7.9	4.6	191.5	3.7	11.7	8.6	204.5	7.5	14.7	10.8	209.2	9.5

In the results of set S1 we can see that our algorithm is not far from the classical VRP solutions with results 4.6% worse in average, although it has not been designed for this problem. The results obtained for set S2 show that splitting the vehicle capacity and the customers demands in different ratios makes different routes necessary to attend all customers. This happens since one of the compartments can get full and forces the vehicle go back to depot even when other compartment still has a residual capacity. The solution of set S2 are in average 8.6% above the best-known values for the VRP. (Note that the optimal values in this case are probably higher than the best known values for the VRP.)

Looking at the results of instance set S3 we can see that splitting the compartments in a more unbalanced way cause the total time of the routes tends to increase, which results in solutions with more routes. In this instance set the solutions are on average 10.7% worse than the best-known values for the VRP. We can also notice a slight increase in the average time to find the best value from 204.45 to 209.23 seconds.

Reed et al. (2014) present results only for the instance *vrpnc1* with vehicle capacity and customers demands split in the same way as instance sets S2 and S3. They have obtained a total route length of 560.74 and 564.04 for splitting methods S2 and S3, respectively. For this instance were able to improve their results. We obtain a total length of 553.76 in average for splitting method S2, and a total length of 556.91 in average for splitting method S3. The best found solutions were with total length of 550.62 for splitting method S2 and 549.51 for splitting method S3.

In Table 3 we report the results for instance set S4 and compare them with the results of Fallahi et al. (2008). For each instance the table reports the best known value obtained by Fallahi et al. (2008), and the relative deviations from this best known values in percent (columns “Cost”) and the time to find them (columns “Time”) for their Memetic Algorithm (MA) as well as their Tabu search Algorithm (TS). These are the only known results for this set of MC-VRP instances. The last two columns give the same results obtained by running our Tabu search algorithm ten times for each VRP instance with demands and capacities randomly generated as described above for instance set S4. The times reported are total execution times. In our case the execution time has been limited to $n^2/300$ seconds, for an instance with n customers. This time has been chosen to provide a fair comparison, considering that the results of Fallahi et al. (2008) have been obtained on a Pentium 4 processor running at 2.4GHz. This processor is about a factor two slower than the processor of our machine. The comparison is further complicated by the fact that Fallahi et al. (2008) report the result of only a single random instance. In our experiments we have found a considerable variation of the results for different demand splittings of the same instance.

Table 3: Results of Tabu search on instances S4 compared to Fallahi et al. (2008).

		Fallahi et al. (2008)				This paper	
		MA		TS			
Name	BKV	Cost	Time(s)	Cost	Time(s)	Cost	Time(s)
vrpnc1	556.1	0.5	17.4	0.0	15.3	-1.8	8.3
vrpnc2	863.6	2.9	25.5	0.0	13.9	0.9	18.5
vrpnc3	837.6	4.9	21.8	0.0	39.8	1.4	33.0
vrpnc4	1070.7	1.7	93.9	0.0	109.7	2.0	74.3
vrpnc5	1361.4	3.5	115.9	0.0	208.4	2.3	130.7
vrpnc6	563.4	1.1	16.5	0.0	10.2	-0.8	8.3
vrpnc7	949.0	0.6	39.2	0.0	22.0	-0.5	18.5
vrpnc8	916.2	4.7	18.7	0.0	18.3	-1.6	33.0
vrpnc9	1262.7	0.0	98.7	2.2	8.6	-4.2	74.3
vrpnc10	1490.2	1.3	140.2	0.0	190.3	0.2	130.8
vrpnc11	1122.9	0.0	47.8	7.0	27.9	1.8	47.5
vrpnc12	926.5	0.0	18.2	0.8	15.8	-4.4	33.0
vrpnc13	1542.4	0.0	76.4	2.6	21.9	1.1	47.5
vrpnc14	966.5	0.0	23.3	18.1	35.7	-3.5	33.0
E072-04f	262.3	0.5	11.7	0.0	5.6	-1.2	17.0
E076-07u	697.8	0.6	15.1	0.0	16.5	0.0	19.1
E076-08s	772.2	2.8	15.4	0.0	13.9	1.6	19.1
E135-07f	1233.2	0.0	47.3	0.2	51.9	1.6	59.1
E241-22k	787.8	1.1	504.5	0.0	202.9	-1.5	190.2
E484-19k	1177.3	5.4	1643.6	0.0	2122.5	5.6	770.6
Average	968.0	1.6	149.6	1.5	157.6	-0.05	88.3

On average, our tabu search is able to find results that are about 1.5% better than those of Fallahi et al. (2008) in a comparable time. The actual differences in solution quality may vary for the instances used in the experiments of Fallahi et al. (2008), but we observe that in 10 of the 20 instances our method consistently obtains equal or better solution values, so we expect this result to be robust. Our results show that a much simpler Tabu search can obtain comparable results, but also show that there is still a potential for an improvement. Another interesting observation is that the overall gain of about 1.5% is of the same order of the improvement that Fallahi et al. (2008) obtain by allowing the splitting of routes, i.e. the demand of a customer for different product types can be satisfied by multiple vehicles.

5. Conclusions

The MC-VRP has important real-world applications but is rarely studied in the literature. We have proposed a constructive heuristic based on the savings method of Clarke and Wright (1964) and a Tabu search to solve this problem. We have presented results for twenty different VRP instances on four different sets of MC-VRP with instances of two compartments . Our algorithm has generated good results compared to existing algorithms, but still has potential for improvement in performance and neighborhood exploration. It would be interesting, in particular, to find a heuristic which combines the advantages of our approach and that of Fallahi et al. (2008) and to study the potential gain of our method by allowing the satisfaction of customer demand for different product types in separate routes.

References

Avella, P., M. Boccia, and A. Sforza (Jan. 2004). “Solving a fuel delivery problem by heuristic and exact approaches”. In: *European Journal of Operational Research* 152.1, pp. 170–179. ISSN: 03772217. DOI: 10.1016/S0377-2217(02)00676-8.

Benantar, A. and R. Ouafi (2012). “Optimization of vehicle routes: an application to logistic and transport of the fuel distribution”. In: *Proc. 9th Int. Conf. Modeling, Optim., and Simulation*. Bordeaux, France.

Clarke, G. u. and J. Wright (1964). “Scheduling of vehicles from a central depot to a number of delivery points”. In: *Operations research* 12.4, pp. 568–581.

Cordeau, J.-F. et al. (May 2002). “A guide to vehicle routing heuristics”. In: *Journal of the Operational Research Society* 53.5, pp. 512–522. ISSN: 0160-5682. DOI: 10.1057/palgrave/jors/2601319.

Cornillier, F., F. Boctor, and J. Renaud (July 2012). “Heuristics for the multi-depot petrol station replenishment problem with time windows”. In: *European Journal of Operational Research* 220.2, pp. 361–369. ISSN: 03772217. DOI: 10.1016/j.ejor.2012.02.007.

Fallahi, A. E., C. Prins, and R. Wolfler Calvo (May 2008). “A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem”. In: *Computers & Operations Research* 35.5, pp. 1725–1741. ISSN: 03050548. DOI: 10.1016/j.cor.2006.10.006.

Glover, F. and M. Laguna (1997). *Tabu Search*. Kluwer.

Laporte, G. (2009). “Fifty years of vehicle routing”. In: *Trans. Sci.* 43.4, pp. 408–416.

Mendoza, J. E. et al. (Nov. 2010). “A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands”. In: *Computers & Operations Research* 37.11, pp. 1886–1898. ISSN: 03050548. DOI: 10.1016/j.cor.2009.06.015.

Reed, M., A. Yiannakou, and R. Evering (Feb. 2014). “An ant colony algorithm for the multi-compartment vehicle routing problem”. In: *Applied Soft Computing* 15, pp. 169–176. ISSN: 15684946. DOI: 10.1016/j.asoc.2013.10.017.

Vidal, T. et al. (2013). “Heuristics for multi-attribute vehicle routing problmes: A survey and synthesis”. In: *European Journal of Operational Research* 231, pp. 1–21.