Yugoslav Journal of Operations Research
34 (2024), Number 3, 423–438
DOI: https://doi.org/10.2298/YJOR220615041A

# SKEWED GENERAL VARIABLE NEIGHBORHOOD SEARCH TO SOLVE THE MULTI-COMPARTMENT VEHICLE ROUTING PROBLEM

Amina AROUSSE
*Modeling and Optimization for Decisional, Industrial and Logistic Systems Laboratory,*
*Faculty of Economics and Management, University of Sfax, Tunisia*
*Arousse.amina@gmail.com*

Ahmed BEN AOUICHA
*Modeling and Optimization for Decisional, Industrial and Logistic Systems Laboratory,*
*Faculty of Economics and Management, University of Sfax, Tunisia*
*ahmed.ben.aouicha@gmail.com*

Mohamed CHEIKH
*Modeling and Optimization for Decisional, Industrial and Logistic Systems Laboratory,*
*Faculty of Economics and Management, University of Sfax, Tunisia*
*mohamed.cheikh@fsegs.usf.tn*

Bassem JARBOUI
*Modeling and Optimization for Decisional, Industrial and Logistic Systems Laboratory,*
*Université Polytechnique Hauts-de-France, LAMIH, CNRS, UMR 8201, Valenciennes 59313, France*
*bassem_jarboui@yahoo.fr*

Received: June 2022 / Accepted: August 2023

**Abstract:** Skewed General Variable Neighborhood Search (SGVNS) is shown to be a powerful and robust methodology for solving vehicle routing problems. In this paper we suggest new SGVNS for solving the multi-compartment vehicle routing problem

(MCVRP). The problem of multi-compartment vehicle routing is of practical importance in the petrol and food delivery and waste collection industries. A comparison between our algorithm and the memetic algorithm and the tabu search is provided. It was clear that the proposed algorithm is capable of solving the available instances. Skewed General Variable Neighborhood Search was used because it makes it easy to explore the space of realizable solutions for MCVRP. As a result, the SGVNS is much faster and more effective. It is able to solve 50 to 484 customers from the literature.

**Keywords:** Variable neighbourhood search, vehicle routing problem, multi-compartment vehicle routing problem.

**MSC:** 90B85, 90C26.

## 1. INTRODUCTION

The vehicle routing problem with compartments is concerned with solving the generalized vehicle routing problem with a homogeneous fleet of vehicles having several compartments as well as additional constraints on the goods loaded in each compartment such as incompatibility between different products in a compartment and between products and compartments. The MCVRP is defined by an undirected graph with a set of nodes $N = \{0, 1, ..., n\}$ including a depot (node 0) and a set $N'$ of $n$ customers. Each edge $(i, j)$ has a cost $c_{ij} = c_{ji}$ . The shops in the depot have a set $P = \{1, 2, ..., M\}$ of m products, which must be delivered by a fleet $V = \{1, 2, ..., V\}$ of identical vehicles with m compartments. The compartment $p$ of each vehicle is dedicated to product $p$ and has a known capacity $Q_p$. Each customer $i$ has a known demand $q_{ip} \leq Q_p$ for each product $p$ , which may be zero for a product not ordered by the customer. In our version, there is also a maximum path length $L$. The formulation of the multi-compartment vehicle routing problem was proposed by El Fallahi et al. [1].

$$MinZ = \sum_{i,j \in N} \sum_{k \in V} c_{ij} x_{ijk} \tag{1}$$

S/C

$$\sum_{i \in N} x_{ijk} \leq 1 \qquad \forall j \in N', \forall k \in V \tag{2}$$

$$\sum_{i \in N} x_{ijk} = \sum_{i \in N} x_{jik} \qquad \forall j \in N', \forall k \in V \tag{3}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \qquad \forall S \subseteq N', |S| \geq 2, \forall k \in V \tag{4}$$

$$y_{jkp} \leq \sum_{i \in N} x_{ijk} \qquad \forall j \in N', \forall k \in V, \forall p \in P \tag{5}$$

$$\sum_{k\in V} y_{jkp} = 1 \qquad \forall j \in N', \forall p \in P \tag{6}$$

$$\sum_{j\in N'} y_{jkp} q_{jp} \leq Q_p \qquad \forall k \in V, \forall p \in P \tag{7}$$

$$\sum_{i,j\in N} c_{ij} x_{ijk} \leq L \qquad \forall k \in V \tag{8}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in N, \forall j \in N, i \neq j, \forall k \in V \tag{9}$$

$$y_{jkp} \in \{0,1\} \qquad \forall j \in N', \forall k \in V, \forall p \in P, q_{jp} \neq 0 \tag{10}$$

The binary variables $x_{ijk}$ (9) are equal to 1 if and only if the edge *(i, j)* ( is crossed by the vehicle *k*. The binary variables $y_{jkp}$ (10) take the value 1 if and only if customer j receives the product $p$ from vehicle *k*. The objective function (1) is to minimise the total cost of the tours formed. In equation (2), each customer can be visited at most once by each route. Equation (3) ensures the continuity of each route: if a vehicle enters node j, it must leave it. Equation (4) presents classical sub-route elimination constraints. The constraints presented in equation (5) are zeroing of the variable $y_{jkp}$ for each product $p$ if customer j is not visited by vehicle *k*. Each product ordered by a customer is delivered by only one vehicle, thanks to the constraints (6). Constraints (7) and (8) respectively ensure the non-violation of the compartment capacity and the route length. Like most of the routing problems considered so far, MCVRP is NP-hard. This class encompasses the set of problems whose solutions cannot be verified by a polynomial algorithm as a function of the instance size. The NP-hard [2] class which has numerical type answers the optimization problems associated with NP-complete decision problems. Despite several real-life applications, MCVRP has not been studied in depth. A well-known application, studied by Brown and Graves [3] in 1981 and Brown *e*t al. [4], is the nationwide shipment of different petroleum products simultaneously to gas stations using tankers with isolated compartments. Brown and Graves [3] introduced a heuristic method to solve this problem while retaining the need for human intervention during the process. In 1990, Apotheker [5] developed a comparison between separate waste collection and mixed collection, and in 1995, Jahre [6] studied vehicles with multi-compartments that are also deployed for the collection of source-separated waste streams where one compartment is dedicated to rubbish and another compartment to recyclables. Research work also includes Kaabachi *e*t al. [7] who worked on the MCVRP. Their main objective of the problem is to minimize the total distance traveled while using a minimum number of trucks. According to the computational results, the optimization approach can give the optimal solution only in small instances. For large problem instances, two algorithms for solving the MCVRP are proposed: a hybrid artificial bee colony algorithm and a hybrid self-adaptive general variable neighborhood algorithm. Henke *e*t al. [8] tried to minimize the total distance to be covered by the disposal vehicles. To solve this problem in an optimal way, a branch-and-cut algorithm developed him. Extensive numerical experiments were conducted to evaluate the algorithm and to better understand the structure of the problem. Eshtehadi *e*t al. [9] proposed an improved adaptive large neighborhood search algorithm for the studied routing problem. The computational results highlight the efficiency of the proposed algorithm in terms of quality and resolution time, and also provide useful information for urban logistics. Martins *e*t al. [10] work extends the research on multi-compartment vehicle routing problems (MCVRP) by tackling a multi-period framework with a product-oriented time window assignment. Silvestrin *e*t al. [11] Studied a variant of the vehicle routing problem that enables multi-compartment vehicles. They proposed a tabu search heuristic and integrate it with an iterated local search to solve the MCVRP. Ostermeiera's *e*t al. [12] paper addresses the problem of routing and selecting single and multi-compartment vehicles for food distribution. They solved the problem using a large neighborhood search. Lahiyani *e*t al. [13] proposed an exact branch-and-cut algorithm to solve MCVRP. They evaluated the performance of the algorithm on real data sets in different transportation scenarios. Reed *e*t al. [14] used the ACS algorithm that is extended to model the use of multi-compartment vehicles with curbside waste sorting into separate compartments for glass and paper. Abdulkader *e*t al. [15] proposed a hybrid algorithm that combines local search with an existing ant colony algorithm to solve MCVRP. In this paper, the main idea of the Variable Neighborhood Search is a systematic change of neighborhoods in a local search procedure. From an initial solution, a perturbation procedure is performed by randomly choosing a solution from the first neighborhood. It is followed by applying an iterative improvement algorithm to improve the initial solution while minimizing the transport cost. This procedure is repeated as long as a new solution is found. An experimental study will be presented in the next section. We'll also show the neighborhood structures and the method used to solve this problem.

## 2. APPLICATION OF THE SKEWED GENERAL VARIABLE NEIGHBORHOOD SEARCH (SGVNS) TO SOLVE THE MULTI-COMPARTMENT VEHICLE ROUTING PROBLEM MCVRP

### 2.1. SGVNS

For the application of the skewed general variable neighborhood search [16] for the multi-compartment vehicle routing problem, we defined an initial randomly created solution, four neighborhood structures, the VND algorithm and the local search associated with the neighborhood structures. We can now elaborate our skewed general VNS algrithm for the MCVRP.

*Algorithm 1: SGVNS Algorithm*

***Input:***

- The set of neighborhood $N_k(k = 1, 2, \dots, k_{max})$ $k_{max} = 4$
- Find an initial solution $s$

***Output:***

- $s^* \longleftarrow s$

***repeat***
  $k \leftarrow 1$;
  ***While*** $K \leq k_{max}$ ***do***,

  - a random neighbor $s' \in N_k(s)$; /*Shaking*/
  - $s'' \leftarrow local\_search(s')$

  ***if*** $f(s'') \leq f(s^*)$ ***then***

  - $s^* \leftarrow s''$

  ***end***
  ***if*** $f(s'') < (1 + \mu * d(s, s'')) * f(s^*)$ ***then***;

  - $s \leftarrow s''$;
  - $k \leftarrow 1$;

  ***else***

  - $k \leftarrow k + 1$;

  ***end***
  ***end***

***until*** a termination condition is met;

***return*** $s^*$

In this algorithm, we used a parameter $\mu$to weight the importance of the distance between solutions, the $\mu \in ]0, 1[$ parameter. The notation $d(s, s'')$ refers to a distance between two solutions. Consider two different solutions of size 6 of our problem: $S_1 = \{d, 1, 2, 3, d\} \cup \{d, 4, 5, 6, d\}$ and $S_2 = \{d, 2, 1, 3, d\} \cup \{d, 4, 5, 6, d\}$. We can represent them differently; each solution will be presented by a vector that contains the next customer of each visited customer, the solutions $S_1$ and $S_2$ become as follows :

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| $S_1$ | 2 | 3 | 1 | 5 | 6 | 4 |

$\rightarrow$

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| $S_2$ | 3 | 1 | 2 | 5 | 6 | 4 |

We note that there are 3 differences between the two solutions; the distance $d(S_1, S_2)$ will be calculated as follows: $d(S_1, S_2) = 3/6 = 1/2$. Generally speaking, the distance between two solutions is the ratio of the difference between the two solutions and the total number of customers.

*2.1.1. Solution representation and initialization*

For our problem, a solution consists in determining the set of tours that form it; this solution is represented by the set of vectors; each representing a tour. Each vector starts with the depot followed by the customers to visit in order and ends with the depot. As shown in the figure, that instance composed of 8 customers and one depot will be represented; we can then distinguish 4 distinct tours.

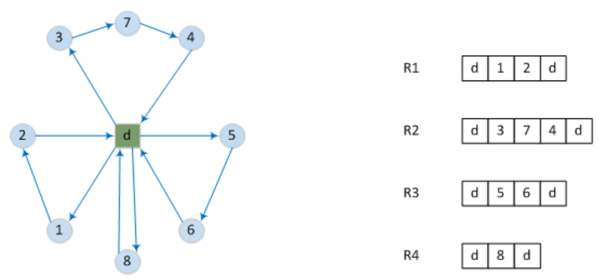


Figure 1: initial Solution

To solve our problem, we must first start with an initial solution. There are two ways, either random or with a heuristic. For our case we will generate the initial solution randomly. The creation of the initial solution is done by randomly assigning a vehicle for each customer.

*2.1.2. Perturbation*

In the perturbation phase, we modify the current local optimum to obtain a new intermediate solution in order to avoid the stopping of this local optimum. Each time a perturbation is made throughout the skewed general VNS procedure, a new solution is created. We perturb the solution in the following way ; we select a vehicle at random and remove a customer already assigned to that vehicle, then insert him into another vehicle. This random insertion is repeated at least twice. This kind of movement generates a new solution that gives the possibility to explore neighbors that have not been explored before.

*2.1.3. Evaluation function*

To enlarge the neighborhood space of a solution, the capacity and length constraints (see constraints (7) and (8) in the proposed formulation) are relaxed. The capacity and length violations are multiplied by two coefficients $\alpha$ and $\beta$ respectively and then added to the objective function. The purpose of this penalty is to expand the search space by visiting infeasible solutions. The penalty costs of a tour $R_k$, which correspond to the overflow of capacity and tour length, can be defined by the following equations:

$$penq(R_k) = Max\{0, \sum_{j \in N'} y_{jkp} q_{jp} - Q_p\} \forall k \in V, \forall p \in P \tag{11}$$

$$penl(R_k) = Max\{0, \sum_{i,j \in N} c_i j x_{ijk} - L\} \forall k \in V \tag{12}$$

The cost resulting from the new cost function defined below is inspired by the one proposed by Gendreau *e*t al. [17] for the vehicle routing problem:

$$F'(x) = F(x) + \alpha Q(x) + \beta L(x)$$

The function $F(x)$ is the cost of solution $x$, $Q(x)$ is the sum of capacity violations for all routings, and $L(x)$ denotes the sum of length violations for all routing. The penalty factors $\alpha$ and $\beta$ are self-adjusting parameters, initialized to given values and dynamically changed during the search. The penalty factors $\alpha$ and $\beta$ are reduced by a constant after each block of successive solutions meeting the capacity and routing length constraints, and multiplied by another constant after each block of successive solutions violating these constraints.

## 2.2. The Variable Neighborhood Descent method (VND)

the proposed solution of the multi-compartment vehicle routing problem, we will change the local search previously used in the skewed variable neighborhood search method algorithm to another method named variable neighborhood descent which will enable us to better explore the neighborhood of each solution.

### *2.2.1. VND Algorithm*

The variable neighborhood descent method was introduced by Hansen and Mladenovic [18] as a local search used in the variable neighborhood search method containing multiple neighborhood structures. The VND aims at iterating the different neighborhood structures used in order to explore the maximum possible number of neighbors. The different steps of VND will be presented in the following Algorithm [19] :

*Algorithm 2: VND Algorithm*

***Input:***

- The set of neighborhood $N_k(k = 1, 2, \ldots, k_{max})$ $k_{max} = 4$
- Find an initial solution $s$
- $k \leftarrow 1$

***Output:***

- $s$

***repeat***
 ***While*** $K \leq k_{max}$ ***do***,
 - $s' \leftarrow local\_search(N_k, s)$
 ***if*** $f(s') < f(s)$ ***then***;
 - $s \leftarrow s'$
 - $k \leftarrow 1$
 ***else***
 - $k \leftarrow k + 1$;
 ***end***
 ***end***

***until*** a termination condition is met;

***return*** $s$

During the implementation of our VND , we then chose to rank the neighborhood structures according to the intensity of their impact on the solution. A permutation of blocks from two clients is performed, followed by an insertion of blocks from two clients, ending with the permutation and insertion.

*2.2.2. Local Search*

A local search method is an iterative process based on two essential elements [20]: a neighborhood and a procedure exploiting the neighborhood. This method consists in :
- Starting with any configuration $s$ of $X$.
- Choosing a neighbor $s'$ of s such that $f(s') > f(s)$ and replacing $s$ by $s'$ and repeating (2) until (for any neighbor) $s'$ of $s$, $f(s') \geq f(s)$(maximization case). The main advantage of this method is that it is very simple and fast. But the solutions produced are often of poor quality and of a cost much higher than the optimal cost [18]. For our local search, we have chosen to stop the search as soon as the first improvement is obtained in order to reduce the search time and to enlarge the exploration space of the feasible solutions.

### 2.3. Neighborhood structures

The proposed variable neighborhood descent method algorithm is made up of four neighborhood structures $V_1, V_2, V_3$ *and* $V_4$ which are mainly insertions and permutations. The neighborhood structures $V_1$ *and* $V_2$ represent the classical insertion and permutation respectively. Neighborhood structures $V_3$ *and* $V_4$ are block insertion and permutation of two customers; they consist in changing the location of a block of two successive customers in the same route or in two different routes. All these structures will be represented more explicitly in the following:

#### *2.3.1. Insertion* $V_1$

There are two types of insertions, an intra-route insertion and an inter- route insertion.

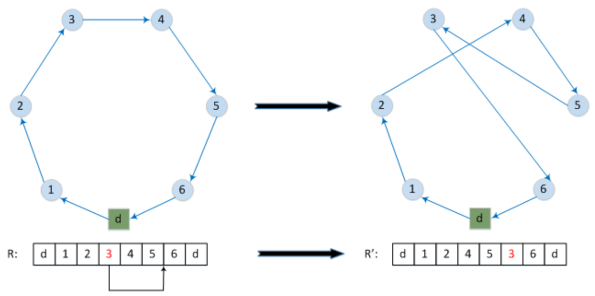


Figure 2: Intra-route insertion

This neighborhood structure consists in modifying the order in which the customers visit. This is a classic move in routing problems. For example, we can see in the figure that customer 3 existing in the 3rd position of the routing is inserted in the 5th position of the same routing.

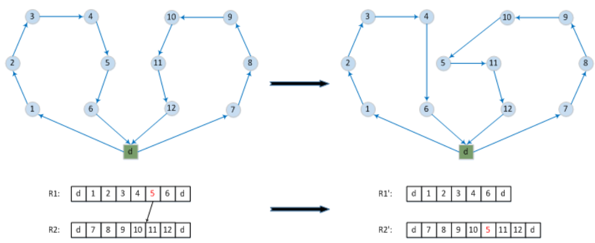


Figure 3: Inter-route insertion

The associated movement in this type of neighborhood is to remove a customer from a given routing and insert it into another routing. In the example shown in the figure, we have eliminated customer 5 existing in the 5th position of the 1st routing and inserted it in the 5th position of the 2nd routing.

### 2.3.2. *Permutaion $V_2$*

There are two types of permutations, an intra-route permutation and an inter-route permutation.

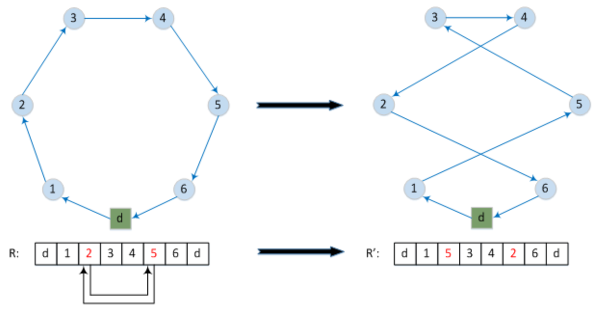


Figure 4: Intra-route Permutaion

This neighborhood structure is obtained by permuting the positions of two customers present in the same routing. In the example shown in the figure, customer 2 is permuted with customer 5.

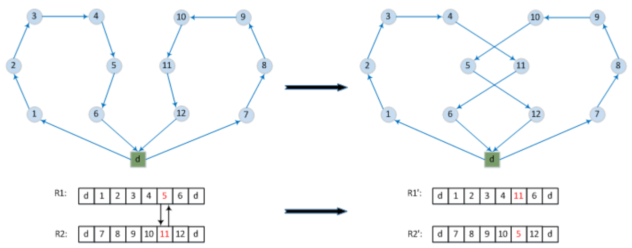


Figure 5: Inter-route Permutaion

This neighborhood structure consists in swapping a customer in a given routing with another customer belonging to another routing. The figure shows us how customer 5 existing in the 5th position of routing R1 is swapped with customer 11 of the 5th position in routing R2.

### 2.3.3. *Block insertion of two customers $V_3$*

There are two different types of block insertions, an intra-route block insertion and an inter-route block insertion.

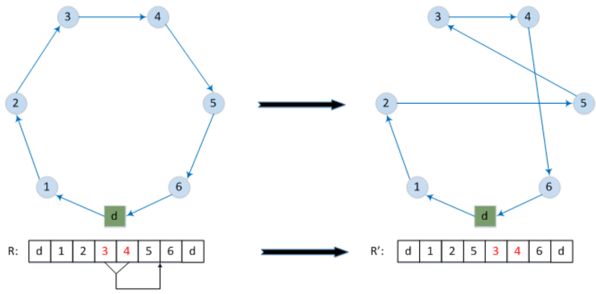


Figure 6: Intra-route block insertion

Block insertion is almost identical with conventional insertion except that moving one customer is replaced by moving a block of customers that includes two customers. In the example in the figure, we have eliminated the existing sequence of customers 3 and 4 between customer 2 and customer 5 and inserted it between customers 5 and 6 of the same routing while keeping the same customer order.

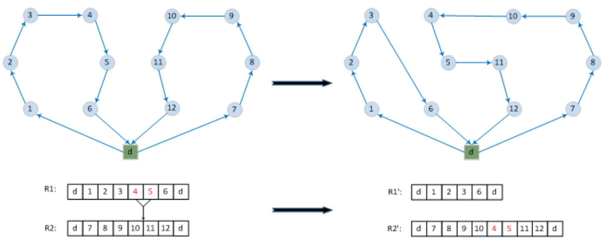


Figure 7: Inter-route block insertion

### *2.3.4. Block permutation of two customers* $V_4$

Like block insertion, block permutation involves permuting two blocks of customers in the same or different routings.

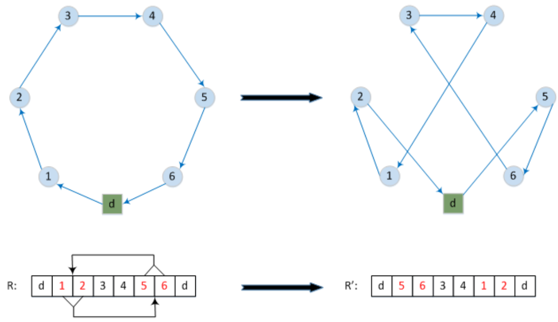


Figure 8: Intra-route block permutation

In this example of a neighborhood structure, we permute between two blocks of customers located in the same routing. In the illustration in the figure we have permuted the sequence of customers 1 and 2 of a given routing with that of customers 5 and 6 while keeping the order of the customers.

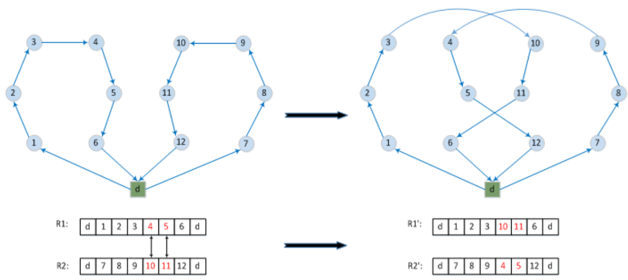


Figure 9: Inter-route block permutation

In this neighborhood structure, the solutions are obtained by permuting a sequence of customers from one of the routings of the current solution with another sequence located in another routing. In the Figure, the sequence of customers 4 and 5 of the 1st routing is permuted with the sequence of customers 10 and 11 of the other routing.

## 3. COMPUTATIONAL RESULTS

The mentioned algorithm was coded with the C language using a computer with an Intel Core i5 processor with 3.2 Ghz and 4 GB of RAM. The experimental study was performed with a collection of instances created by El Fallahi [1] and tested on a pentium 4 computer. Twenty well-known instances for VRP were transformed into two sets of MCVRP instances with m= 2, creating two compartments and dividing the demand of each customer into two quantities. The first 14 problems (vrpnc) are due to Christofides and contain 50-199 customers. Instances 6-10, 13 and 14 include a routing length restriction. The last six cases are from Eilon: their names start with "E", followed by the number of customers (76-484). All of these cases can be downloaded from or at the Beasley library. We tested the proposed skewed general VNS algorithm on two different scenarios. The first scenario requires the delivery of customer requests by a single vehicle; the second enables us to separate the requests provided that each request is fully delivered in full by a single vehicle. The table summarizes our results. The first and second columns contain the label of each instance used and its size. The next four columns contain the best results obtained by El Fallahi [1], either by the memetic algorithm or by the tabu search, regarding the two scenarios and the associated computation time. The next four columns respectively list the best results obtained by the proposed skewed general VNS algorithm and the associated computation time. The last column summarizes the percentage gain of our results in comparison with the El Fallahi [1] and Silvestrin [11] results.

We first compare the results of each scenario separately. Starting with the first scenario of delivering customers without demand segregation, we find that the skewed general VNS (SGVNS) successfully improved the results of 17 instances and failed to improve the results of 3 instances, these results are colored in green. The average improvement rate is 2%. The total execution time was improved with an improvement rate of 96.25%. We notice that the skewed general VNS solved the MCVRP well regarding the first scenario and the efficiency of our algorithm to provide better results in a considerably fast execution time. Regarding the second scenario, the customers' requests can be separated, i.e., each customer can receive its two product requests by two different vehicles. The proposed skewed general VNS algorithm successfully improved 18 instances with an improvement rate ranging from 0.04% to 4% and it managed to achieve the same result for the instance ($E076-08s$) with an execution time of 11.7 seconds in comparison with 52.41 seconds for the El Fallahi [1] result, all these results are colored in blue. The average improvement rate is 1.5%. The total execution time was improved with an improvement rate of 79.71%. Finally, since both scenarios are generations of the same MCVRP, we need to discuss all the results together by comparing our VNS results with the results El Fallahi [1]. Our skewed general VNS Algorithm successfully improved 19 instances. We also notice that the skewed general VNS algorithm for the first scenario is indeed the fastest, given that the search space is limited, but it only managed to improve 7 instances, all of which were moderately large. As for the second scenario, it managed to improve 14 instances with a considerably good execution time. The overall improvement rate is 1.55%. Comparing with the work of Silvestrin [11], and despite his use of a good processor, his obtained results show a minor compromise between the execution time and the quality of the solution. In other words, the deviation he found is not so important

| n_d_l | Size | Mem/Tabu | | | | ITS | | | | SGVNS | | | | Success rate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Without separation | | With separation | | Without separation | | With separation | | Without separation | | With separation | | |
| | | Best | t(s) | Best | t(s) | Best | t(s) | Best | t(s) | Best | t(s) | Best | t(s) | |
| *vrpnc*1 | 50 | 556.1 | 15.3 | 548.36 | 22.5 | 525,9156 | 5.0 | 525,9156 | 6.0 | 546.95 | 1.453 | 543.38 | 5.93 | 0.9164857 |
| *vrpnc*2 | 75 | 863.6 | 13.9 | 873.59 | 56 | 851,5096 | 6.8 | 847,1916 | 9.8 | 861.25 | 0.85 | 852.88 | 6.885 | 1.2569177 |
| *vrpnc*3 | 100 | 837.6 | 39.8 | 832.89 | 158.97 | 829,5684 | 11.2 | 829,5684 | 15.3 | 830.07 | 1.565 | 829.94 | 18.642 | 0.3554474 |
| *vrpnc*4 | 150 | 1070.7 | 109.7 | 1075.99 | 346.98 | 1049,286 | 16.0 | 1051,4274 | 23.3 | 1054.03 | 18.705 | 1054.57 | 103.584 | 1.5815489 |
| *vrpnc*5 | 199 | 1361.4 | 208.4 | 1362.75 | 390.78 | 1342,3404 | 22.8 | 1345,0632 | 32.9 | 1347.27 | 13.587 | 1346.91 | 78.64 | 1.0757957 |
| *vrpnc*6 | 50 | 563.4 | 10.2 | 558.6 | 29.89 | 555,2484 | 5.8 | 555,2484 | 5.0 | 557.49 | 0.39 | 556.07 | 0.24 | 0.4549787 |
| *vrpnc*7 | 75 | 949 | 22 | 952.59 | 33.56 | 913,887 | 7.0 | 914,836 | 10.5 | 922.43 | 0.795 | 914.45 | 27.5 | 3.7782273 |
| *vrpnc*8 | 100 | 916.2 | 18.3 | 890.08 | 101.3 | 866,0673 | 9.7 | 866,0673 | 10.3 | 866.86 | 0.889 | 866.86 | 14.82 | 2.6786332 |
| *vrpnc*9 | 150 | 1262.7 | 98.7 | 1186.15 | 310.56 | 1174,338 | 12.5 | 1173,1518 | 18.9 | 1166.36 | 3.445 | 1170.18 | 134.16 | 1.6967317 |
| *vrpnc*10 | 199 | 1490.2 | 190.3 | 1475.8 | 412.5 | 1427,0986 | 16.6 | 1425,6228 | 26.2 | 1432.45 | 10.96 | 1444.14 | 133.131 | 3.0262836 |
| *vrpnc*11 | 120 | 1122.9 | 47.8 | 1113.38 | 94.7 | 1053,2764 | 11.0 | 1048,8228 | 13.6 | 1141.78 | 11.005 | 1105.45 | 56.82 | 0.7173549 |
| *vrpnc*12 | 100 | 926.5 | 18.2 | 906.94 | 33.7 | 866,9964 | 12.1 | 866,9964 | 16.2 | 900.53 | 0.904 | 898.78 | 33.112 | 0.9078974 |
| *vrpnc*13 | 120 | 1542.4 | 76.4 | 1541.23 | 141.3 | 1542,7412 | 7.8 | 1542,7412 | 12.6 | 1546.71 | 0.379 | 1537.48 | 106.61 | 0.2439056 |
| *vrpnc*14 | 100 | 966.5 | 23.3 | 934.73 | 61.7 | 914,1366 | 13.0 | 915,0713 | 9.1 | 932.51 | 2.964 | 913.97 | 23.697 | 2.2714093 |
| $E072-04$f | 72 | 262.3 | 5.6 | 262.45 | 12.4 | 1162,5695 | 49.8 | 1173,149 | 77.9 | 262.34 | 0.203 | 262.83 | 2.699 | 0 |
| E076-08s | 76 | 772.2 | 13.9 | 748.36 | 52.41 | 743,4368 | 23.0 | 738,0384 | 33.2 | 748.36 | 1.9 | 748.36 | 11.7 | 0 |
| E076-07u | 76 | 697.8 | 16.5 | 699.2 | 21.36 | 248,9227 | 7.2 | 244,4636 | 7.3 | 693.33 | 0.49 | 693.33 | 3.275 | 0.6447146 |
| $E135-07f$ | 135 | 1233.2 | 47.3 | 1248.3 | 260.58 | 693,6132 | 7.3 | 692,9154 | 10.1 | 1214.59 | 2.918 | 1217.06 | 10.67 | 1.5322043 |
| $E241-22k$ | 241 | 787.8 | 202.9 | 771.21 | 919.26 | 747,6516 | 7.2 | 745,4064 | 9.4 | 766.27 | 11.396 | 755.5 | 54.025 | 2.0794176 |
| $E484-19k$ | 484 | 1177.3 | 2122.5 | 1175.47 | 3878.5 | 1206,0696 | 18.5 | 1201,1368 | 22.7 | 1169.697 | 38.797 | 1151.82 | 662.767 | 2.0532722 |
| *average* | | 967.99 | 165.05 | 957.9035 | 366.9475 | 935,733665 | 13.5 | 935,14169 | 18.5 | 948.06385 | 6.17975 | 943.198 | 74.44535 | 1.5591106 |

Table 1: Computational results

compared to our work.

## 4. CONCLUSION

In this paper, We developed the skewed general variable neighborhood search to solve the multi-compartment vehicle routing problem. This proposed method aims at minimizing the transportation cost. Subsequently, a comparison between our algorithm and the memetic algorithm and tabu search is developed. It was clear that our algorithm is capable of solving the available instances containing from 50 to 484 customers and it proved its efficiency regarding good quality results and faster execution time than the previous algorithms. Our study also showed that the separation of customer requests provided good quality results for most instances. The experimental results showed that the proposed skewed general VNS performs well.In general, it achieves a good solution in a small computation time which is better than the solutions in the literature.

**Funding.** This research received no external funding.

## REFERENCES

[1] A. El Fallahi, C. Prins, and R. Wolfler Calvo, "A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem," *Computers and Operations Research*, vol. 35, no.5, 2008, pp. 1725-1741.

[2] A. Mjirda, B. Jarboui, R. Macedo, S. Hanafi, and N. Mladenovic, "A two phase variable neighborhood search for the multi-product inventory routing problem," *Computers and Operations Research*, vol. 52, Part B, 2014, pp. 291-299.

[3] G.G. Brown and W.G. Graves, "Real-time dispatching of petroleum tank trucks," *Management Science*, vol. 27, no. 1, 1981, pp. 19-32.

[4] G.G. Brown, C.J. Ellis, W.G. Graves, and D. Ronen, "Real-time, wide area dispatch of Mobil tank trucks," *Interfaces*, vol. 17, no. 1, Franz Edelman Award Papers (Jan. - Feb., 1987), pp. 107-120.

[5] S. Apotheker, "Curbside collection:Complete separation versus commingled collection," *Resource Recycling*, 1990.

[6] M. Jahre, "Logistics systems for recycling-efficient collection of household waste," Ph.D. thesis, Chalmers University of Technology, 1995.

[7] I. Kaabachi, H. Yahyaouia, S. Krichena, and A. Dekdoukb, "Measuring and evaluating hybrid metaheuristics for solving the multi-compartment vehicle routing problem, Measurement," *Measurement*, vol. 141, July 2019, pp. 407-419.

[8] T. Henke, M. Grazia Speranza, and G. Wäscher, "A branch-and-cut algorithm for the multi-compartment vehicle routing problem with flexible compartment sizes," *Annals of Operations Research*, vol 275, 2019. pp. 321–338.

[9] R. Eshtehadi, E. Demirb, and Y. Huang, "Solving the vehicle routing problem with multi-compartment vehicles for city logistics," *Computers and Operations Research*, vol 115, March 2020, 104859.

[10] S. Martins, M. Ostermeier, P. Amorim, A. Hübner, and B. Almadalobo, "Product-oriented time window assignment for a multi-compartment vehicle routing problem," *European Journal of Operational Research*, vol 276, no 3, August 2019, pp. 893-909.

[11] P.V. Silvestrin, and M. Ritt, "An iterated tabu search for the multi-compartment vehicle routing problem," *Computers & Operations Research*, vol. 81, 2017, pp. 192-202.

[12] M. Ostermeiera, and A. Hübnerb, "Vehicle selection for a multi-compartment vehicle routing problem," *European Journal of Operational Research*, vol. 269, no. 2, September 2018, pp. 682-694.

[13] R. Lahyani, L. C. Coelho, M. Khemakhem, G. Laporte, and F. Semet, "A Multi-Compartment Vehicle Routing Problem Arising in the Collection of Olive Oil in Tunisia," CIRRELT, 2014.

[14] M. Reed, A. Yiannakou, and R. Evering, "An ant colony algorithm for the multi-compartment vehicle routing problem," *Applied Soft Computing*, vol 15, February 2014, pp. 169-176.

[15] M.S. Abdulkader, Y. GajpalbTarek, and Y. ElMekkawy, "Hybridized ant colony algorithm for the Multi Compartment Vehicle Routing Problem," *Applied Soft Computing*, vol. 37, December 2015, pp. 196-203.

[16] S. Toumi, M. Cheikh, and B. Jarboui, "0-1 Quadratic Knapsack Problem solved with VNS algorithm," *Electronic Notes in Discrete Mathematics*, vol. 47, February 2015, pp. 269-276.

[17] M. Gendreau, A. Hertz, and G. Laporte, "A tabu search heuristic for the vehicle routing problem," *Management Science*, vol. 40, no. 10, October 1994, pp. 1207-1393.

[18] P. Hansen, N. Mladenović, and J.A. Moreno-Pérez, *Variable neighbourhood search: methods and applications (invited survey)* . Annals of Operation Research, 367- 407.

[19] B. Jarboui, H. Derbel, S. Hanafi, and N. Mladenovic, *Variable neighborhood search for location routing.* Computers and Operations Research, 2013.

[20] P. C. Pop, and A. Horvat-Marc, *Local search heuristics for the generalized vehicle routing problem.* International Conference on System Modeling and Optimization, 2012.

[21] Henke Tino, M. Grazia Speranza and Gerhard Wäscher. *The multi-compartment vehicle routing problem with flexible compartment sizes.* European Journal of Operational Research, 2015.

[22] M.Ostermeier, T.Henke, A. Hübner, G. Wascher. *Multi-compartment vehicle routing problems: State-of-the-art, modeling framework and future directions.* European Journal of Operational Research, 2021.

[23] P.VitorSilvestrin, M.Ritt. *An iterated tabu search for the multi-compartment vehicle routing problem.* Computers and Operations Research, 2021.