RAIRO-Oper. Res. 55 (2021) 2265–2283
https://doi.org/10.1051/ro/2021100

RAIRO Operations Research
www.rairo-ro.org

A CONSTRUCTION HEURISTIC FOR FINDING AN INITIAL SOLUTION TO A VERY LARGE-SCALE CAPACITATED VEHICLE ROUTING PROBLEM

Bapi Raju Vangipurapu[1,*] and Rambabu Govada[2]

Abstract. In this paper, a deterministic heuristic method is developed for obtaining an initial solution to an extremely large-scale capacitated vehicle routing problem (CVRP) having thousands of customers. The heuristic has three main objectives. First, it should be able to withstand the computational and memory problems normally associated with extremely large-scale CVRP. Secondly, the outputs should be reasonably accurate and should have a minimum number of vehicles. Finally, it should be able to produce the results within a short duration of time. The new method, based on the sweep algorithm, minimizes the number of vehicles by loading the vehicles nearly to their full capacity by skipping some of the customers as and when necessary. To minimize the total traveled distance, before the sweeping starts the customers are ordered based on both the polar angle and the distance of the customer from the depot. This method is tested on 10 sets of standard benchmark instances found in the literature. The results are compared with the results of the CW^{100} method by Arnold *et al.* [*Comput. Oper. Res.* **107** (2019) 32–42]. The results indicate that the new modified sweep algorithm produces an initial solution with a minimum number of vehicles and with reasonable accuracy. The deviation of the output from the best-known solution (BKS) is reasonable for all the test instances. When compared with the CW^{100} the modified sweep provides a better initial solution than CW^{100} whenever the capacity of the vehicle is more and the depot is located eccentrically. The heuristic does not face any memory problems normally associated with the solving of an extremely large-scale CVRP.

Mathematics Subject Classification. 90C27, 90C59, 90-08, 90B06, 68T20.

Received December 5, 2020. Accepted July 6, 2021.

1. Introduction

Vehicle routing problems form one of the most studied areas in combinatorial optimization. Dantzig and Ramser first formulated it in 1959 [7]. The vehicle routing problem can be formulated as follows. Starting from a source (also called depot) several destinations (customers) should be visited in such a way that the length of the resulting tour(s) is minimized. In the canonical vehicle routing problem (also called the capacitated VRP or CVRP) the capacity of the used vehicles is limited and thus more than one tour is usually necessary to visit all customers.

Keywords. Capacitated Vehicle Routing Problem, construction heuristic, modified sweep algorithm, extremely large-scale problems.

[1] Department of Mechanical Engineering, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada 520007, India.

[2] Department of Mechanical Engineering, Andhra University, Visakhapatnam 530003, India.

*Corresponding author: rajuvpublications@gmail.com

A very large-scale CVRP consists of thousands of customers. Kytöjoki *et al.* [12] presented an example of this type of CVRP where the author discusses the waste collection system in which trucks have to collect waste from tens of thousands of customers. Another example is the distribution of parcels given by Arnold *et al.* [2] and Arnold and Sörensen [1]. Here, the authors report that in Belgium the daily quantity of delivered parcels corresponds to about 1% of the population so that in cities like Brussels 20 000 and more deliveries need to be carried out every day.

CVRP is an NP-hard problem [17]. Since the first VRP was presented, many algorithms have been proposed for solving either the classical VRP or its variants. Those algorithms can be divided into three main groups: exact algorithms, heuristics, and metaheuristics [13]. Exact methods can only solve problems with a limited number of customers as their complexity grows rapidly with problem size. On the other hand, although constructive heuristics find a solution quickly they give only an approximate solution that needs to be improved by metaheuristics to get an optimum solution. Metaheuristics find high-quality solutions in a reasonable amount of computing time. Therefore, a major research stream on the design of metaheuristics for the CVRP has evolved over the past two decades [9].

Metaheuristics take reasonable processing time as long as the problem has less number of customers (say less than 1000). However, as the size of the problem grows and the number of customers runs into thousands the time taken by the metaheuristics is considerable. For example, consider the state-of-the-art KGLS$^{\mathrm{XXL}}$ (longer run time) by Arnold *et al.* [3]. To get a near-optimum solution the authors allow the metaheuristics to run for 20 min for every 1000 customers (longer run time) so that a test instance with 30 000 would take approximately 10 h to run. It has to be noted that some test instances (especially ones with shorter routes) give near-optimum solutions earlier but some test instances (especially ones with longer routes) take more time. Therefore, whenever a quick approximate solution is needed constructive heuristics are useful. One such example is the combinatorial auction problem where a quick approximate solution is needed [11] for several combinations. Another example would be the case of an online grocery store like the big basket where thousands of deliveries would be sent from one single store. Because of the stringent delivery times (*e.g.*, [21] `https://www.bigbasket.com/delivery-information/`) sometimes there may not be enough time to run the metaheuristics to get optimum solutions. In those cases, a quick approximate solution is provided by constructive heuristics. Besides this, obtaining a good initial solution plays an important role in obtaining a near-optimal solution at least in some of the cases as pointed out by some of the researchers (though it has not been verified in the present study). For example, Van Breedam [18] states that the performance of the tabu search heuristic is highly dependent on the quality of the initial solutions. Brandão [4] also shows in his research that the initial solution can give an important contribution to enhance the final solution.

Most of the existing constructive heuristics fail to provide good approximate solutions to the large-scale vehicle routing problem because of the time complexity and memory problems associated with solving such a large-scale VRP [3]. For instance, the savings heuristic stores the complete distance matrix, *i.e.*, the distance between each pair of nodes, to compute the length of edges and routes during the execution of the algorithm. This results in n^2 entries (where n is the number of nodes). Processing such huge data slows down the process and sometimes may exceed the available computer memory [3]. Then there are other heuristics like sweep algorithms which do not face time complexity and memory problems but produce inferior solutions in terms of distance traveled and the number of vehicles [3]. In order to mitigate the problems of space and time complexities arising out of the large-scale problems, Arnold *et al.* [3] developed CW^{100} based on the savings algorithm where savings are calculated for each node and its 100 nearest nodes. The authors then compared the sweep algorithm [10] the savings algorithm [5], and the CW^{100} algorithm for their capability to generate good initial solutions for large-scale problems. After preliminary evaluations, the authors found that the modified savings algorithm, CW^{100}, is best suited to get the initial solution for large-scale problems. As per the knowledge of the authors, CW^{100} seems to be the best constructive heuristic as far as large-scale CVRP is concerned. However, the performance of the CW^{100} is not very good for the test instances with large capacity vehicles and eccentrically located depot (depot located away from customers). This is evident from the large variation between the output of CW^{100} and the BKS for these test instances compared to the other test instances [3]. To overcome this limitation, a heuristic is proposed

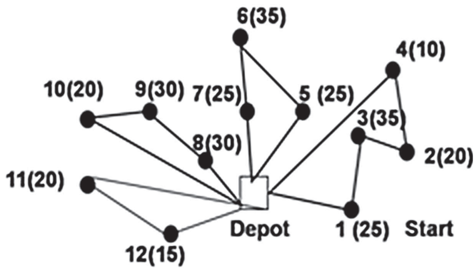

FIGURE 1. Solution using a normal sweep.

in this paper which is based upon the modified sweep heuristic of Vangipurapu *et al.* [19, 20]. This heuristic can find a reasonably good initial solution within a few seconds and it does not face any memory problems because there is no storage of huge amounts of data. It gives a solution with less number of vehicles when compared with the output of CW^{100} and at least in some cases (with large capacity vehicles and eccentrically located depot), the traveled distance of the solution is less when compared with the solution from CW^{100}.

The rest of the document is organized as follows. The problem is defined in Section 2. In Section 3 the methodology that is used to obtain an initial solution for a large-scale CVRP is explained. Section 4 describes the experimental tests that were conducted. In Section 5 a discussion is carried out on the results that were obtained. Finally, in Section 6 the conclusion of this work is presented.

2. Problem statement

The problem studied in this paper concerns obtaining an initial solution to an extremely large-scale vehicle routing problem. The aim is to develop a heuristic that is capable of

(a) Withstanding the computational and memory problems normally associated with extremely large-scale CVRP.
(b) Producing an initial solution that is reasonably accurate (from the traveled distance point of view) and has the number of vehicles equal to (or close to) the BKS.
(c) Producing the results within a short duration of time.

3. Methodology

The methodology involves taking the existing modified sweep algorithm by Vangipurapu *et al.* [19, 20] and modifying it suitably to meet the objectives of the study. First, the existing algorithm is described later on the changes that are done the algorithm to meet the objectives are discussed.

3.1. Modified sweep algorithm

The Modified sweep algorithm by Vangipurapu *et al.* [19, 20] is a heuristic to produce an initial solution with a minimum (or less) number of vehicles. The objective of minimizing the number of vehicles is achieved by loading the vehicles nearly to their full capacity by skipping some of the customers during sweeping as and when necessary. This concept is explained using Figures 1 and 2. In this example, the capacity of each vehicle is assumed to be 100. In Figure 1, normal sweeping is done from customer 1 in the anticlockwise direction and a new vehicle is formed whenever the total demand exceeds the capacity of the vehicle. This results in 4 vehicles. Figure 2 corresponds to the modified sweep to reduce the number of vehicles. Here, after adding customers 5–7 to vehicle 2 customers 8–11 are skipped (since they result in violating capacity constraint) and customer 12 is added. This would result in only three vehicles.

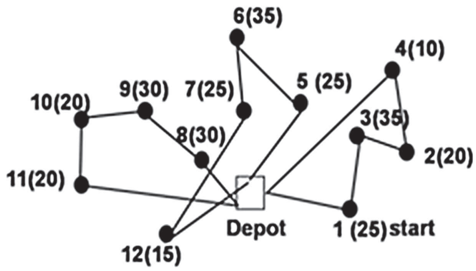

Figure 2. Solution using a modified sweep.

Though their algorithm can minimize the number of vehicles the quality of the solutions in terms of traveled distance is rather poor. Therefore, this algorithm is modified based on the following two perspectives.

(1) The process of skipping involves grouping far away customers together to load the vehicles to their full capacity. This means that whenever the next candidate customer (in the sweeping order) violates the capacity constraint, the algorithm tries to find some far away (ahead in the sweeping order) customer who does not violate the capacity constraint. This however is likely to increase the traveled distance. Thus in a quest to minimize the number of vehicles the objective of accuracy (in terms of traveled distance) is sacrificed. However, when the vehicle is sufficiently full it does not make sense to do a skipping, as this process will only contribute to the increase in the traveled distance with bleak chances of reduction in the number of vehicles. Therefore, there should be a check to prevent skipping whenever the vehicle is sufficiently full.
(2) The modified sweep arranges the customers based on the polar angle. This should work fine for the cases whenever the number of customers is low and all the customers are located near the depot. However, whenever the number of customers is large and the customers are spread over a wide range of distance from the depot this would result in the inferior solution from the traveled distance point of view. This is because if customers are very far away but the angle between them is very less they are likely to be grouped together resulting in a large traveled distance. For example, consider the arrangement of customers and depot as shown in Figure 3. The demand of all customers is assumed to be "1" unit and the capacity of each vehicle is assumed to be "5" units. If sweeping were to be done in clockwise direction customers 1–5 would be grouped in one vehicle and 6–10 would be grouped in another. It is evident from the figure that because of the distantly located customer 4 the traveled distance of vehicle 1 increases enormously. Therefore, it makes sense to replace it with rather a closely located customer like customer 7. This would reduce the traveled distance of vehicle 1 without adversely affecting the traveled distance of vehicle 2. Hence, the sweeping order should be based not only on the polar angle but also on the distance of the customers from the depot. Therefore, there should be a mechanism to arrange the customers in the proper order before sweeping sorts.

3.2. Modifications to reduce the distance during skipping

(1) As explained above skipping a customer to load the vehicles fully, achieves the objective of reducing the number of vehicles at the expense of an increase in traveled distance. Therefore skipping should be minimized as far as possible if the total traveled distance needs to be minimized. This is achieved by the following steps. First, a minimum loading factor (Mf) of a vehicle is defined as follows

$$\text{Mf} = \text{Capacity of each vehicle} - \frac{\text{Total capacity} - \text{Total demand}}{\text{Total number of vehicles}}.$$

This parameter indicates to what extent the vehicles must be loaded to have a solution with a minimum number of vehicles. Skipping a customer to maximize the loading of the vehicle is done only when the current vehicle is loaded below the Mf. This reduces the amount of skipping and hence the traveled distance.

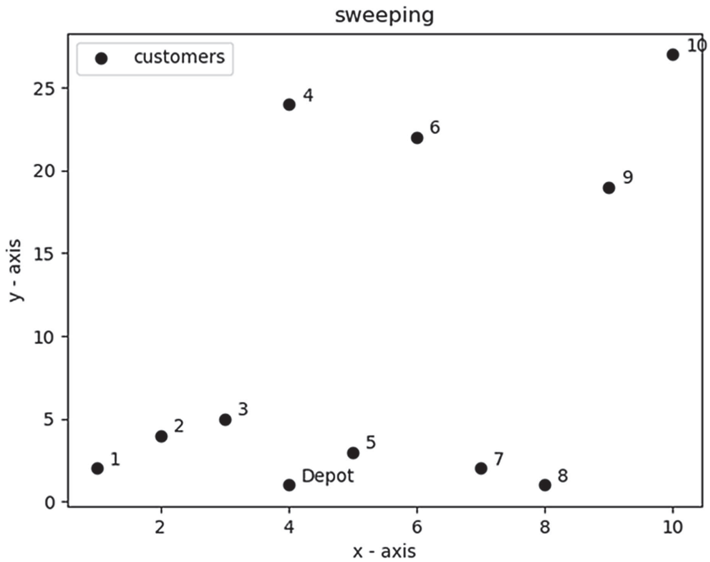

FIGURE 3. Huge variation in distance of customers from depot.

Example: Let the total demand be 950. The capacity of each vehicle is 100. So the minimum vehicle required $= (950/100) = 10$ (rounded up).
Therefore Mf $= 100 - \frac{1000-950}{10} = 95$ which means that skipping should be attempted only if the capacity of the vehicle is less than 95%.

(2) If the quality of the solution in terms of the traveled distance needs to be good, the routes should be as compact as possible [1]. This implies that the angular distance between the current cluster and the next potential customer to be added to the vehicle should be as low as possible. Considering this point, the following changes are done to the modified sweep algorithm. While skipping the customers, apart from finding the customer with the right demand, the angular deviation from the previous customer is also considered. If the angular deviation from the previous customer is more than $\frac{\pi}{6}$, the customer is skipped even if it has the right demand to make the vehicle loaded above the Mf level and meets the capacity constraints. This step prevents the customers who are very far away from the current group of customers from being clubbed together and consequently making the solution poor. The threshold limit of $\frac{\pi}{6}$ is based on a thumb rule that was arrived at after many trials

3.3. Modifications done to the sweeping order to reduce the traveled distance during sweeping

As explained earlier in order to minimize the traveled distance, the sweeping order of the customers should be based both on the traveled distance and on the polar angle. This is achieved by the following process. Based on their distance from the depot customers are divided into "N" groups (proposed heuristic will determine the best value of N) each of which is separated from its neighboring groups by concentric circles. The grouping is done in such a way that a more or less equal number of customers is present in each group. All customers within each group are arranged as per the polar angle. Only after sweeping is done in one particular group does the sweeping start in the next adjacent group. To maintain the continuity of the sweep across the groups, sweeping is done in clockwise and in anti-clockwise directions in alternate groups respectively. This arrangement ensures that far away customers that have a very small polar angle between them do not get into the same vehicle. A sample arrangement of customers is Figure 4. This example contains 138 customers which are divided into 4 groups based on their distance from the depot. Each group will have 34 (138/4 rounded down to the nearest

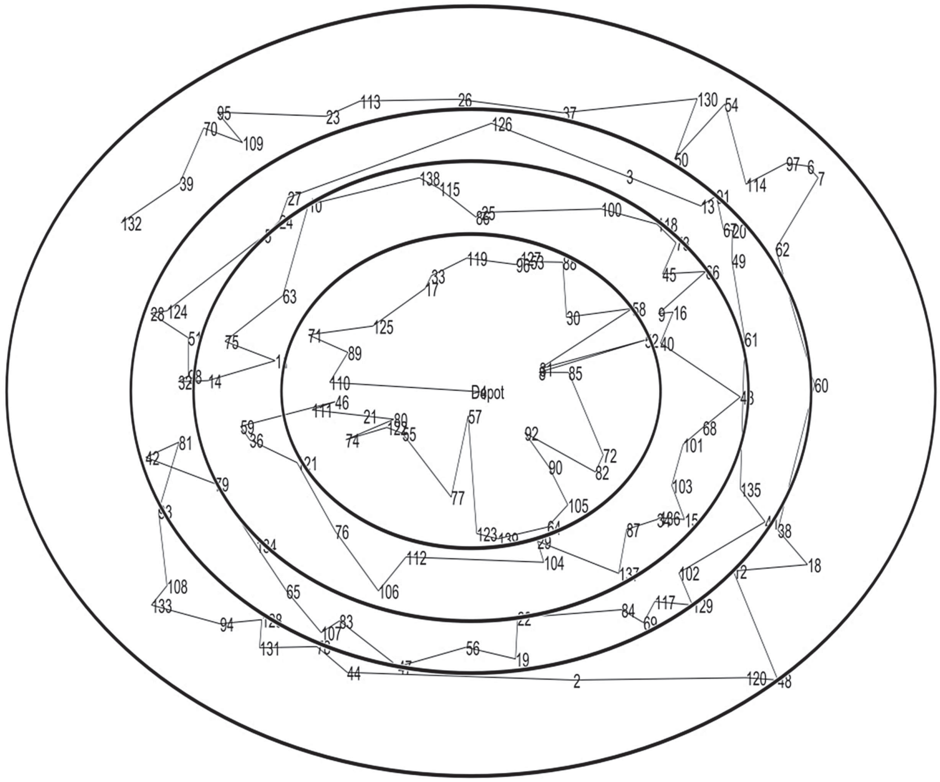

Figure 4. Sample arrangement of customers.

integer). This arrangement ensures that the far away customers are not grouped together and hence the traveled distance of the vehicle is automatically reduced.

3.4. Algorithm to generate the initial solution to large-scale vehicle routing problem using modified sweep

(1) Locate the depot as the center. Compute the polar coordinates of all customers.
(2) Do until no more improvement is done consecutively for 2 iterations starting with $N = 1$.
   - Sort all customers as per the distance from the depot.
   - Divide customers into N divisions based upon their distance from the depot. The number of customers in each division would be equal to $\frac{\text{Total number of customers}}{N}$ (rounded down to the nearest integer). (Thus, each group will have an approximately equal number of customers.)
   - Sort customers in the first division as per the polar coordinates in a clockwise direction. Sort customers in the next division as per the polar coordinates in an anti-clockwise direction. This process of sorting in a clockwise and in an anti-clockwise direction is continued until all the divisions are sorted. Now all the divisions are joined one by one in the same order so that we get a total list of customers in one particular order.
   - Starting from the first customer modified sweep algorithm is applied to get the first solution in the iteration. Nearest neighbor algorithm with 2-opt improvement is used for solving traveling salesman problem of individual routes. Starting from the last customer modified sweep algorithm is applied in the

reverse direction to get the second solution of the iteration similarly to the first. The better of the two solutions is taken as the solution from the iteration.

- The solution from the current iteration is compared with the solution from the previous two iterations.
- N in increased to $N + 1$.

End do.

4. Experimental tests

For testing the modified sweep algorithm, test instances from Arnold *et al.* [3, 16] benchmark problems are chosen. The new algorithms are implemented on Matlab [14]. The experiments have been done on a PC (Intel Core i7-8550U CPU @ 1.80 GHz, 8 GB RAM) with Windows 10 OS. For comparison, the normal sweep is also run on the test instances.

5. Results & discussion

The results from the algorithm are tabulated in Table 1. As can be seen from Table 1 modified sweep algorithm produces reasonably accurate initial solutions. None of the solutions deviate more than 8% from their respective BKS. The number of vehicles taken is always equal to the corresponding solution of BKS. The CPU time is reasonable (even for instances with 30 000 customers the CPU time taken is slightly more than 1 min). It can be seen that the normal sweep produces rather poor solutions and hence removed from further contention. Hence, a further comparison is made only between CW^{100} and the proposed modified sweep algorithm.

When compared with the initial solutions produced by the CW^{100} the new algorithm appears to provide better initial solutions whenever the depot is located at a corner and the vehicles have high capacity (corresponding to the instances L2, A2, G2, B2). This is due to two reasons. The first reason is that the CW^{100} (just like the original savings algorithm) is greedy in nature [8]. Right from the beginning, it groups customers with shorter distances between them. This makes sense for shorter routes (relatively small capacity vehicles with very few customers in each vehicle) as close customers are grouped together. However, as the routes get longer (relatively high capacity vehicles with many customers in each vehicle) the algorithm has to group more customers together into a single-vehicle. Therefore, towards the end of the construction process, CW^{100} is left with no option but to add relatively distant customers to the current group of customers resulting in unprofitable routes. On the other hand, the proposed algorithm is not greedy in nature it divides all customers into different groups based on both the polar angle and the distance from the depot (The algorithm determines the best way to divide). Therefore, the proposed algorithm results in a better solution for larger-capacity vehicles. The relatively good performance of the proposed algorithm for large-capacity vehicles becomes even better as the location of the depot is moved from the center to the corner position. This is because, as the depot moves farther away from the customers, the average distance of the customers in a vehicle from the depot plays a prominent role (than the distance between the customers) in calculating the traveled distance of the vehicle. The proposed algorithm groups customers based on the distance from the depot. This ensures that at least some of the vehicles have relatively short routes contributing to the lower total traveled distance even when the number of customers per vehicle is more (large-capacity vehicles). The CW^{100}, on the other hand, makes no such attempt and hence the average distance of the customers in any vehicle is relatively large. This effect is more pronounced especially for large-capacity vehicles (for reasons mentioned earlier) contributing to a higher total traveled distance. The second reason for the proposed algorithm to show better performance for longer routes is its ability to minimize the number of vehicles. While the CW^{100} checks for capacity constraints, it makes no explicit effort to minimize the number of vehicles. On the other hand, the proposed algorithm can minimize the number of vehicles by ensuring that vehicles are loaded nearly to their full capacity. Even in this case, the performance of the proposed algorithm becomes even better as the location of the depot is moved from the center to the corner position. The reason for this is the angular threshold limit of $\frac{\pi}{6}$ that is used in the skipping process to find the right customer to make vehicles sufficiently full. When the depot is at the center the customers are distributed around the depot and, hence the number of customers falling within the threshold limit of the current vehicles will be

TABLE 1. Comparison of results from the proposed algorithm with the results of CW^{100} by Arnold [3].

Problem instance				CW^{100} [3] (#)				Normal sweep ($)				Proposed modified sweep ($)				BKS [3] (#)		
Instance	No of customers	No of vehicles	Capacity	Traveled distance	% Deviation from BKS	*No of vehicles	Time (s)	Traveled distance	% Deviation from BKS	#No of vehicles	Time (s)	Traveled distance	% Deviation from BKS	#No of vehicles	Time (s)	KGLS XXL [3]	No of vehicles	Time (s)
L1	3000	203	25	**200 957**	3.34		6	289 379.5	48.81	208	6	206 145.3	6.01	203	1.95	194 456	203	3600
L2	4000	46	150	126 122	9.85		6	132 745.7	15.62	46	6	**123 522.4**	7.58	46	2.12	114 817	46	4800
A1	6000	343	30	**497 441**	3.29		12	689 615.4	43.2	349	12	505 156.7	4.9	343	3.48	481 583	343	7200
A2	7000	120	100	322 073	8.79		12	392 251.8	32.49	121	12	**314 415.4**	6.2	120	3.46	296 055	120	8400
G1	10 000	485	35	**489 556**	3.38		24	747 088.1	57.76	493	24	501 647.9	5.93	485	12.28	473 568	485	12 000
G2	11 000	110	170	288 942	9.24		24	315 341.3	19.22	111	24	**279 518.1**	5.67	110	8.56	264 512	110	13 200
B1	15 000	512	50	**531 980**	4.91		54	867 123.9	71	518	54	545 226	7.52	512	18.24	507 103	512	18 000
B2	16 000	182	150	384 437	8.06		66	475 913.1	33.77	183	66	**376 074.7**	5.7	182	16.3	355 779	182	19 200
F1	20 000	684	50	**7 518 845**	3.06		102	10 803 076	48.08	691	102	7 760 184	6.37	684	52.97	7 295 447	684	24 000
F2	30 000	256	200	**4 803 502**	6.64		228	6 317 914	40.26	257	228	4 864 143	7.99	256	66.33	4 504 416	256	36 000
Avg					6.06		53.4		41		53.4		6.39		18.57			

Notes. (#)AMD Ryzen 3 1300X CPU @ 3.5 GHz on Windows 10 Average CPU mark (6902) Single Thread Rating: 2094 Overall Rank: 697 (https://www.cpubenchmark.net/). ($)Intel Core i7-8550U @ 1.80 GHz on Windows 10 Average CPU mark (5965) Single Thread Rating: 2086 Overall Rank: 817 (https://www.cpubenchmark.net/). (*)The number of vehicles is not mentioned in the output. However, they are more than the number of vehicles present in the BKS at least for longer routes [3]. The bold value indicates the best solution among all methods

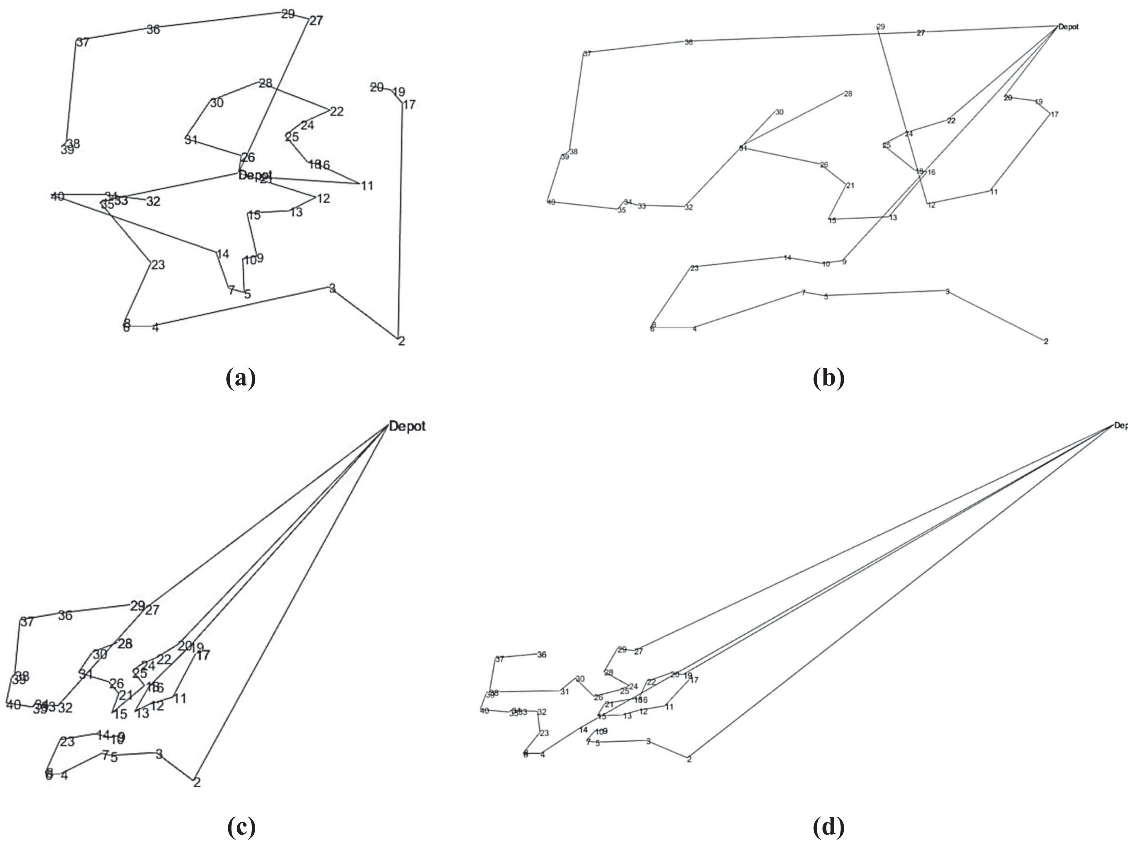

FIGURE 5. Depot Located at various distances from customers. (a) Short distance (center). (b) Medium distance. (c) Long distance. (d) Very long distance.

much less (the customer's polar angles calculated w.r.t depot is spread over large values). Therefore, chances of finding the right customer to make the vehicles sufficiently full will be less. As the depot starts to move away from the center towards one corner number of customers falling within the threshold limit of the current vehicles will be much more (the customer's polar angles calculated w.r.t depot is spread over small values). Therefore, chances of finding the right customer to make the vehicles sufficiently full will be more. Further, if the depot is located very far the distance saved would be substantial (as the average distance of routes increases) thus increasing the gap between the solutions of CW^{100} and the proposed algorithm. To provide support for the two reasons further testing is done as follows. Two sets of smaller datasets having 39 customers are created, one for validating the effect of the capacity of the vehicles on the solution quality and the other for validating the effect of minimization of number vehicles by the proposed algorithm. The details of the datasets are presented in Appendix A. First, the effect of the capacity of the vehicle and the location of the depot on the performance of the proposed algorithm is discussed using the first dataset. To eliminate the effect of reduction of the number of vehicles on the traveled distance all the customers are assumed to have unit demand. This would ensure that the outputs from both the CW^{100} and the proposed algorithm would have an equal number of vehicles. 36 test instances are derived from this data set by varying the capacity of the vehicle from 7 units to 15 units and by varying the location of the depot from the center (a short distance from customers) to an extreme corner (very large distance from customers). The various locations of the depot are shown in Figure 5. CW^{100} of Arnold was run using this tool [22] and modified sweep was run by using Authors tool on his own machine.

TABLE 2. Comparison of outputs from the proposed algorithm and from the CW^{100} algorithm for smaller test instances with unit demand.

Demand 1 for all customers No of customers 39			CW^{100} [3]	Proposed modified sweep
Problem instance (Distance of depot from the customers)	No of vehicles	Vehicle capacity	Traveled distance	Traveled distance
S.1 (Short)	3	15	**1568**	1693
S.2 (Short)	3	14	**1591**	1656
S.3 (Short)	3	13	**1585**	1672
S.4 (Short)	4	12	**1585**	1851
S.5 (Short)	4	11	**1594**	1718
S.6 (Short)	4	10	**1685**	1695
S.7 (Short)	5	9	**1793**	1902
S.8 (Short)	5	8	**1917**	1949
S.9 (Short)	6	7	**1987**	2017
M.1 (Medium)	3	15	2212	**2096**
M.2 (Medium)	3	14	2187	**2071**
M.3 (Medium)	3	13	2183	**2061**
M.4 (Medium)	4	12	**2171**	2245
M.5 (Medium)	4	11	**2366**	2393
M.6 (Medium)	4	10	**2403**	2460
M.7 (Medium)	5	9	**2607**	2741
M.8 (Medium)	5	8	**2829**	2906
M.9 (Medium)	6	7	**3134**	3162
L.1 (Long)	3	15	4227	**4063**
L.2 (Long)	3	14	4230	**3984**
L.3 (Long)	3	13	4225	**4047**
L.4 (Long)	4	12	4994	**4971**
L.5 (Long)	4	11	5130	**5049**
L.6 (Long)	4	10	**5066**	5105
L.7 (Long)	5	9	**6081**	6104
L.8 (Long)	5	8	**6172**	6246
L.9 (Long)	6	7	**7211**	7284
L.1 (Very Long)	3	15	6325	**6165**
L.2 (Very Long)	3	14	6328	**6104**
L.3 (Very Long)	3	13	6322	**6197**
L.4 (Very Long)	4	12	7936	**7846**
L.5 (Very Long)	4	11	8064	**7867**
L.6 (Very Long)	4	10	8054	**7914**
L.7 (Very Long)	5	9	**9590**	9613
L.8 (Very Long)	5	8	9804	**9760**
L.9 (Very Long)	6	7	**11 423**	11 459

Notes. The bold value indicates the better solution between the two methods

The two algorithms are run on these test instances. The results are presented in Table 2 and Figure 6. As expected, the performance of the proposed algorithm gets better as the capacity of the vehicles increases in each case. However, this performance is insufficient to beat the performance of CW^{100} in the case of a centrally located depot (depot located at a short distance). Nevertheless, as the depot starts moving towards a corner the performance of the proposed algorithm increases rapidly (for the reasons explained above) to such an extent that it starts to outperform CW^{100} at least for large-capacity vehicles. Thus, the proposed algorithm gives superior

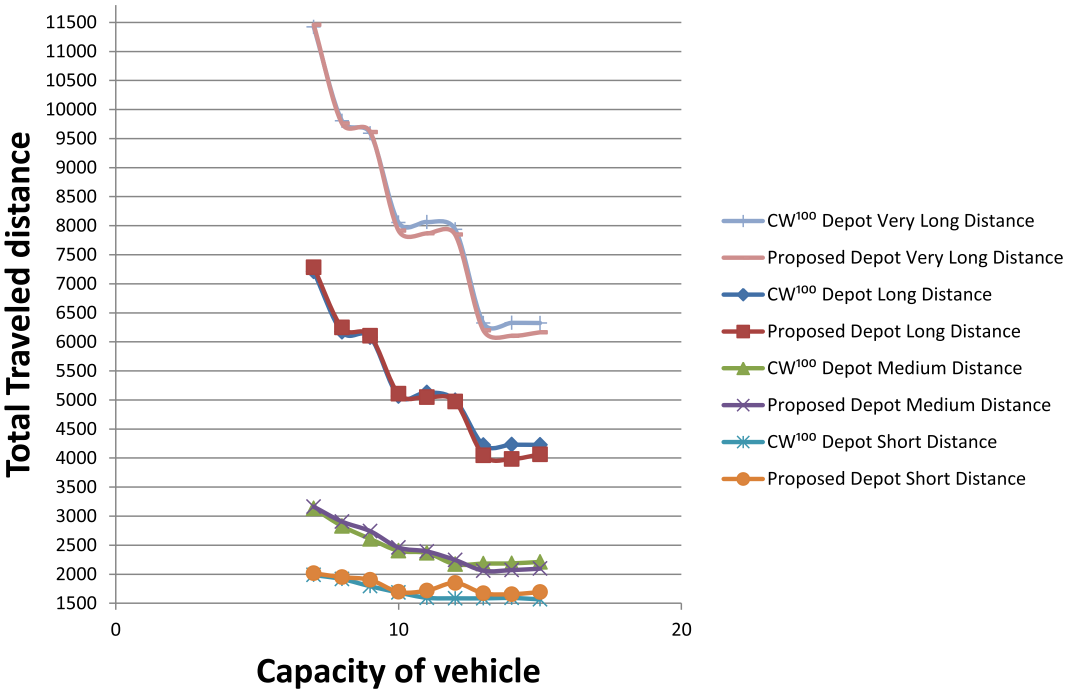

FIGURE 6. Comparison of output from the CW^{100} and the proposed algorithm for various depot locations and vehicle capacities (demand of each customer is equal to 1).

results as both the distance of the depot and the capacity of the vehicles increase. It has to be noted that the effect of minimizing the number of vehicles has been nullified by taking unit demand for all the customers. Next, to assess the effect of minimization of the number of vehicles by the proposed algorithm further testing is carried out using the second dataset, which has varying demands (data is given in Appendix A). 24 test instances are derived from this data set by varying the capacity of the vehicle from 10 units to 60 units and by varying the location of the depot from the center (a short distance from customers) to an extreme corner (very large distance from customers) [22]. Again, the two algorithms are run on these test instances and the results are provided in Table 3 and Figure 7. From the results, it can be observed that the ability to minimize the number of vehicles increases, as the depot, starts moving away from the center towards a corner (for the reasons explained above). Further, the ability to minimize the traveled distance increases as the capacity of the vehicle increases. Therefore, the minimization of the number of vehicles and consequently the minimization of the total traveled distance is at its best for the cases where the depot is located very far and the capacity of the vehicles is very large. Overall, the performance of the proposed algorithm is at its highest when the capacity of the vehicles is very large and the depot is located at a faraway distance from the customers. This also explains why the proposed algorithm is not able to give better results for the test instances L1, A1, G1, B1, F1 of Arnold *et al.* [3] . F2 test instance has high capacity vehicles but the depot is not sufficiently far away from customers. This is evident from the figures of the solutions by CW^{100} [3] . To corroborate the findings, further testing is carried out this time on the large-scale test instances of Arnold *et al.* [3] by modifying some of the data. The test instances L1 and A1 both of which have centrally located depot with a large number of vehicles of small capacity are selected for this. The capacity of each vehicle in the L1 instance is 25 and that of the A1 instance is 30. For these test instances, the locations of the depot and the capacity of the vehicles are modified in different ways so that we get various test instances. The location of the customers and their demand is unchanged. CW^{100} of Arnold was run using this tool [22] and modified sweep was run by using

Table 3. Comparison of outputs from the proposed algorithm and from the CW^{100} algorithm for smaller test instances with varying demand.

No of customers 39 Total demand 300			CW^{100} [3]			Proposed modified sweep		
Problem instance (Distance of depot from the customers)	Theoretical minimum no of vehicles	Vehicle capacity	Total traveled distance	No of vehicles*	Average route length	Total traveled distance	No of vehicles	Average route length
S.1 (Short)	5	60	**1948**	6	498	2347	6	610
S.2 (Short)	10	30	**2755**	11	437	3040	11	549
S.3 (Short)	15	20	**3608**	17	408	3884	17	466
S.4 (Short)	20	15	**4822**	22	422	5146	22	432
S.5 (Short)	25	12	**6097**	**31**	407	6269	33	395
S.6 (Short)	30	10	**6498**	**36**	380	6523	37	384
M.1 (Medium)	5	60	**2988**	6	782	3050	**5**	856
M.2 (Medium)	10	30	**4805**	11	772	5491	11	820
M.3 (Medium)	15	20	**6943**	16	782	7461	16	761
M.4 (Medium)	20	15	**9286**	23	760	9509	**22**	752
M.5 (Medium)	25	12	12 624	31	762	**12 259**	31	724
M.6 (Medium)	30	10	**13 686**	36	715	13 807	36	715
L.1 (Long)	5	60	7035	6	1201	**6428**	**5**	1225
L.2 (Long)	10	30	12 632	11	1194	**12 290**	**10**	1214
L.3 (Long)	15	20	18 770	16	1261	**18 255**	16	1194
L.4 (Long)	20	15	26 060	22	1290	**24 817**	22	1192
L.5 (Long)	25	12	35 423	31	1256	**33 658**	31	1183
L.6 (Long)	30	10	38 584	36	1180	38 584	36	1180
L.1 (Very Long)	5	60	11 249	6	1875	**9928**	**5**	1986
L.2 (Very Long)	10	30	20 143	11	1831	**19 315**	**10**	1932
L.3 (Very Long)	15	20	30 719	16	1920	**29 492**	16	1843
L.4 (Very Long)	20	15	42 952	23	1867	**40 298**	**22**	1832
L.5 (Very Long)	25	12	56 926	31	1836	**55 477**	31	1790
L.6 (Very Long)	30	10	**63 923**	36	1776	63 929	36	1776

Notes. (*)The number of vehicles from CW^{100} may be slightly more for some test cases as the tool provides the number of vehicles only after some improvement steps. The bold value indicates the better solution between the two methods

Authors tool on his own machine. Now the CW^{100} algorithm and the proposed algorithm are run on the new test instances. The results are presented in Table 4, Figures 8 and 9 (to prevent cluttering the diagrams, the edges connecting the depot to the last customers are not displayed in Figs. 8 and 9). The proposed algorithm is a clear winner in the cases of 5 test instances where the capacity of the vehicles is large and the depot is located at the corner (L1.e, L1.f, A1.e, A1.f). In these cases, the modified sweep produces a solution with less traveled distance and less (or equal number) number of vehicles when compared with the solution produced by the CW^{100}. Out of the two instances where the capacity is high but the depot is located at a short distance (located at the center), the proposed algorithm gives a solution a better solution for one test instance (A1.b). For the other case, L1.b, though the CW^{100} algorithm produces slightly better solutions in terms of traveled distance its solution has one extra vehicle than the solution produced by the modified sweep algorithm. Hence, even in this case, the modified sweep algorithm produces a better solution (assuming that a solution with a less number of vehicles is considered superior when the traveled distances are comparable). The remaining six test instances either have shorter routes (centrally located depot) or have low-capacity vehicles. As expected, the CW^{100} gives better results in these cases. Therefore, it can be concluded that the modified sweep algorithm produces a better solution for instances with large-capacity vehicles and remotely located depot.

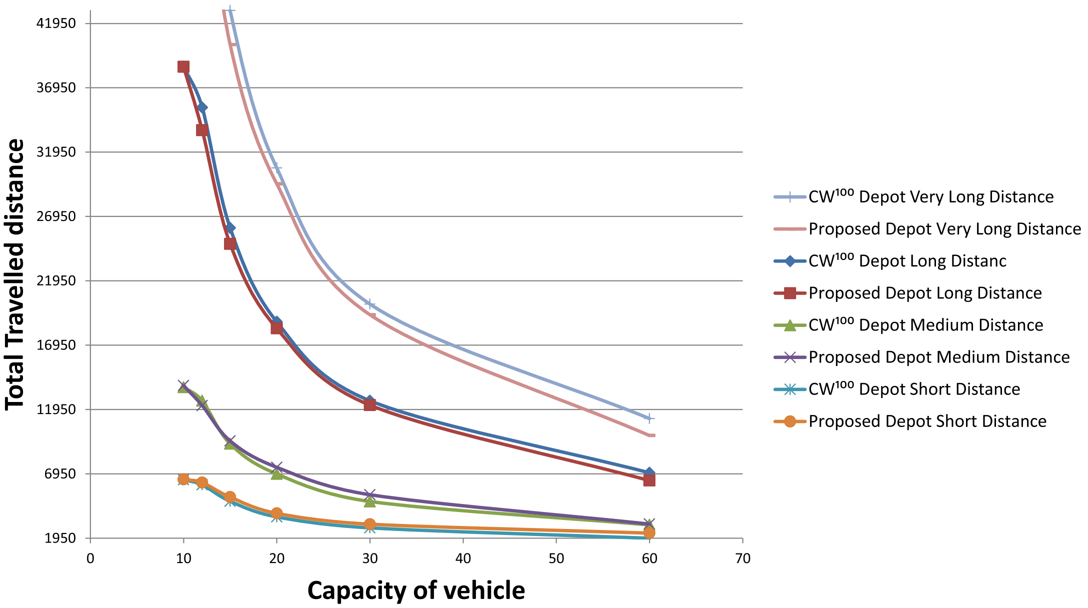

FIGURE 7. Comparison of output from the CW^{100} and the proposed algorithm for various depot locations and vehicle capacities (demand of customers is varying).

TABLE 4. Comparison of outputs from the proposed algorithm and from the CW^{100} algorithm for larger instances of Arnold *et al.* [3].

Problem description					CW^{100} [3]		Proposed algorithm		% Difference between two methods
Instance	Number of customers	Number of vehicles	Capacity of vehicle	Depot location	Traveled distance	Number of vehicles*	Traveled distance	Number of vehicles	
L1.a	3001	203	25 (Low)	Center	**199 496.7**	203	206 082.9	203	3.3
L1.b	3001	35	145 (High)	Center	**65 266.11**	36	66 080.93	**35**	1.2
L1.c	3001	203	25 (Low)	Corner 1	**435 648.9**	204	442 335.2	203	1.5
L1.d	3001	203	25 (Low)	Corner 2	**512 533.8**	203	519 043.9	203	1.3
L1.e	3001	35	145 (High)	Corner 1	108 243.9	37	**106 692.8**	**35**	1.5
L1.f	3001	35	145 (High)	Corner 2	119 122.8	36	**118 887.1**	**35**	0.2
A1.a	6001	343	30 (Low)	Center	**493 675.2**	343	505 014.1	343	2.3
A1.b	6001	42	245 (High)	Center	139 257.7	44	**132 580.4**	**42**	5
A1.c	6001	343	30 (Low)	Corner 1	**841 127.3**	343	850 376.6	343	1.1
A1.d	6001	343	30 (Low)	Corner 2	**954 325.4**	343	979 699.8	343	2.7
A1.e	6001	42	245 (High)	Corner 1	178 323.9	43	**175 233.5**	**42**	1.8
A1.f	6001	42	245 (High)	Corner 2	192 253.4	42	**188 228**	42	2.1
		Average			353 272.9	156.4	357521.3	155.8	

Notes. (*)The number of vehicles from CW^{100} may be slightly more for some test cases as the number of vehicles is provided by the tool only after some improvement steps. The bold value indicates the better solution between the two methods

The second comparison is made on the number of vehicles. In all the cases, the modified sweep algorithm resulted in the same number of vehicles as the BKS. In general, the modified sweep algorithm results in the same number of vehicles as the BKS for the reasons explained above but on some occasions, it might result

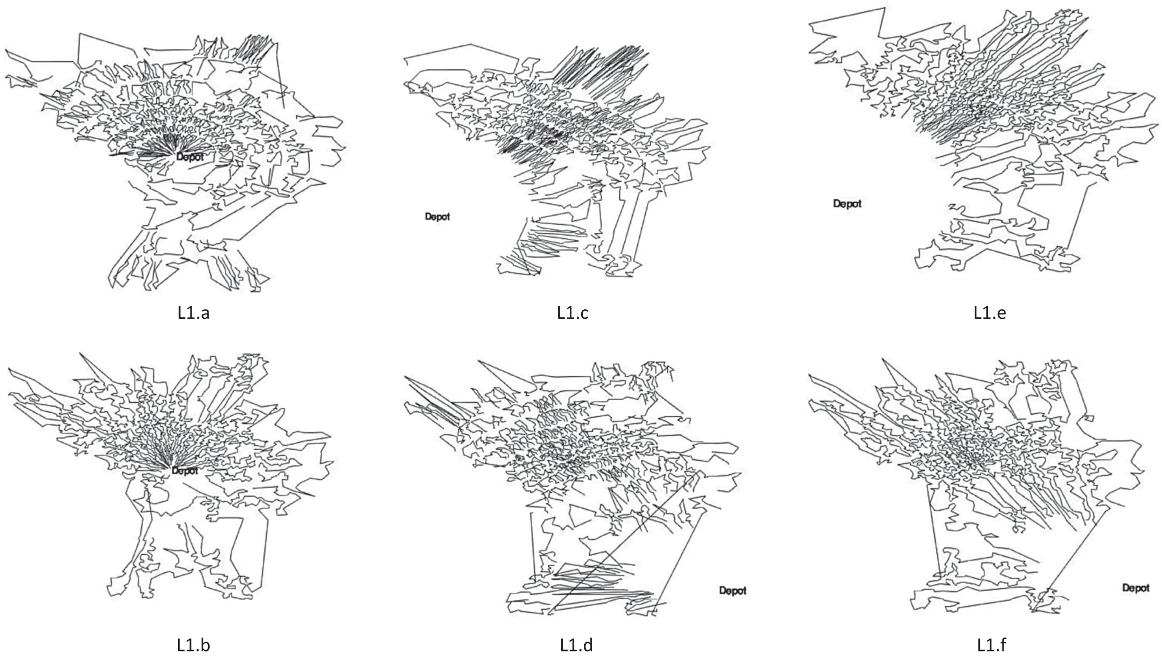

Figure 8. Initial Solutions by the proposed algorithm for L1 instances with various combinations of depot locations and vehicle capacity.

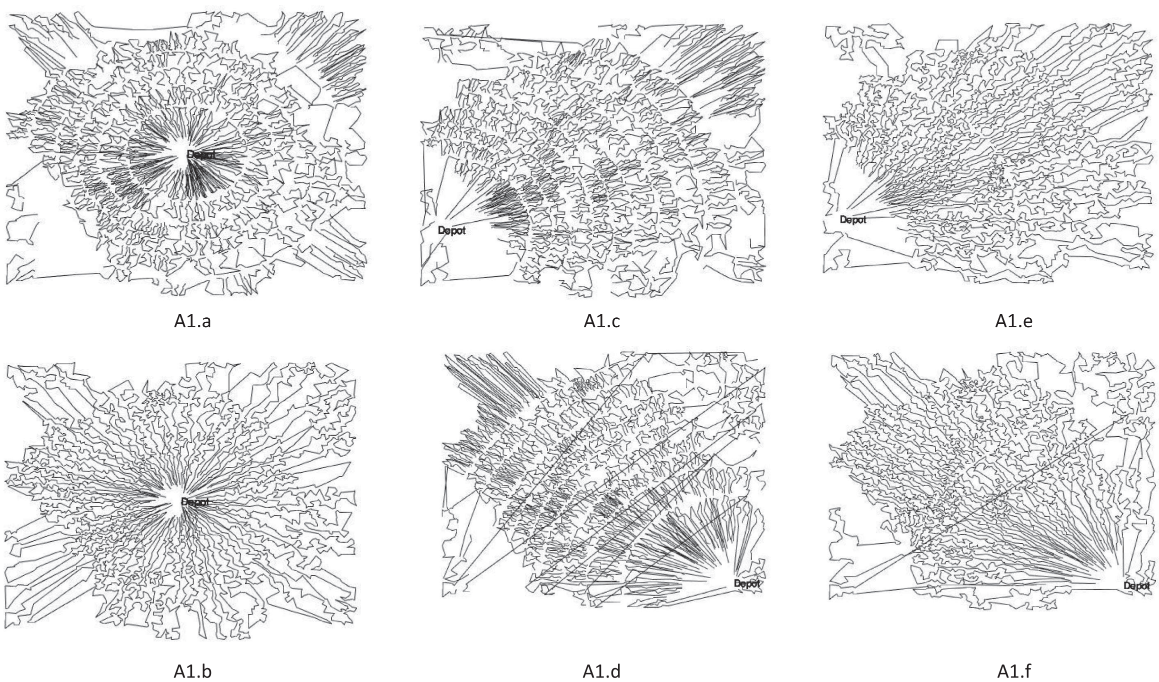

Figure 9. Initial Solutions by the proposed algorithm for A1 instances with various combinations of depot locations and vehicle capacity.

TABLE 5. Comparison of performance of two methods on large-scale test instances of Arnold *et al.* [3].

CW^{100} by Arnold *et al.* [3]	Proposed modified sweep
The average deviation from the BKS is 6.06%	The average deviation from the BKS is 6.39%
Performs better for test instances with low capacity vehicles with centrally located depot (instances L1, A1, G1, B1)	Performs better for test instances with high capacity vehicles with eccentrically located depot (instances L2, A2, G2, B2)
Generally results in more number of vehicles than BKS (especially for longer routes) and hence further local search is required to reduce the number of vehicles	Generally, results in fewer vehicles, and hence further local search is generally not required to reduce the number of vehicles. The ability to reduce the number of vehicles increases as the depot moves away from the customers
Auxiliary storage requirement is relatively more	Auxiliary storage requirement is relatively less
Average CPU time is relatively more (Average 53.4 s) (Note: run on a slightly faster computer)	Average CPU time is relatively less (Average 18.57 s)
Maximum Deviation is 9.85%	Maximum Deviation is 7.99%

in a solution having slightly more vehicles than the minimum as indicated by BKS. This would be the case especially if the depot is located centrally. On the other hand, CW^{100} produces a solution with more vehicles especially for instances with large capacity vehicles and eccentrically located depot [3]. Though the authors do not mention the number of vehicles generated in the initial solution, they do mention that for instances with longer routes and high capacity vehicles initial solution from CW^{100} would have more vehicles than BKS.

The third comparison is made from the memory point of view. The modified sweep algorithm does not face memory problems of any sort, as it does not store any matrices of a higher order than n (n is the number of customers). Though the space complexity of both the proposed algorithm and CW^{100} algorithms is $O(n)$, the auxiliary space required by the proposed modified sweep algorithm is comparatively less when compared to CW^{100}. The auxiliary space used by the quick sort used in the implementation proposed algorithm is at most n units [6]. This along with the other auxiliary data required by the proposed algorithm will be around a few times n. On the other hand, the CW^{100} stores the distance data of 100 nearest neighbors for all the customers. Hence, the distance data itself will have an auxiliary space of $100 \times n$ units. This along with the other auxiliary data required by the CW^{100} will be relatively much more than the auxiliary space required by the proposed algorithm.

The final comparison between the two algorithms is made on the processing times. The modified sweep algorithm comparatively takes less CPU time for processing. For example, for getting an initial solution for problem F2 with 30 000 customers the modified sweep algorithm takes 160 s as against 228 s taken by CW^{100}. It has to be noted down that according to PassMark Software [15] the setup used to run the modified sweep algorithm (Intel Core i7-8550U CPU @ 1.80 GHz, 8GB RAM on Windows 10 OS) is slightly slower than the setup used to run CW^{100} (AMD Ryzen 3 1300X CPU working at 3.5 GHz on Windows 10). Hence the difference between the processing times should be seen slightly on a higher side and modified sweep performs better in this aspect. The relevant details are provided at bottom of Table 1.

The proposed algorithm however has one limitation. Sometimes while dividing the customers into different groups based on the distance from the depot, there is a possibility that customers might be widely separated into a single group. As the algorithm tries to group these widely separate customers into a single vehicle the total traveled distance increases. The problem gets even more aggravated if these customers happen to be falling into the outer groups. An example of this is present in Figure 8 (A1.d) where very long clear edges cross across the whole map.

Overall, it can be observed that the modified sweep algorithm gives better results in at least some of the cases where the capacity of the vehicles is large and the depot is eccentrically located. Hence, it can be used along with CW^{100} to get good initial solutions. The comparison between these two methods is summarized in Table 5.

Initial Solutions produced by the modified sweep algorithm for L1 and A2 test instances of Arnold *et al.* [3] are shown in the Appendix B.

6. Conclusion

In this work, a heuristic that can find good initial solutions to extremely large-scale CVRP instances having tens of thousands of customers was developed. This heuristic minimizes the number of vehicles by loading the vehicles nearly to their full capacity. The minimization of the number of vehicles is achieved by the skipping method. Further, it minimizes the traveled distance by grouping the customers based on both the polar angle and the distance from the depot. The performance of the algorithm is at its best when the depot is located at a large distance and the capacity of the vehicles is large. An example of the real- life application of the proposed algorithm would be the case where the depot is situated outside the city, customers are located inside the city, and huge containers are used for transportation. When compared to CW^{100} (which is the best construction heuristic for large-scale problems as of now as per the best knowledge of the authors) the proposed heuristic outperforms whenever the problems have large-capacity vehicles and eccentrically located depot.

The proposed algorithm has following limitation:

(1) The solution quality can deteriorate if the customers who are approximately at equal distance from the depot are separated by a wide distance (as explained in the Sect. 6).
(2) The quality of the solution will not be up to the mark for centrally located depot.

Future works may explore following possibilities:

(1) Overcoming limitations discussed above without excessively complicating the heuristic.
(2) Study the effect of varying the parameter minimum loading factor (Mf) on the solution quality.

Appendix A.

Following is the data related to the test instances developed by the author for demonstration. Depot is represented by customer no 1 (Tabs. A.1 and A.2).

Table A.1. Location of depot for various test instances.

Depot location	X co-ordinate	Y co-ordinate
Short distance from customers (center)	380	380
Medium distance from customers	500	500
Long distance from customers	750	750
Very Long distance from the customers	1000	1000

TABLE A.2. Data of customers for various test instances.

Test instances with Unit demand				Test instances with varying demand			
Customer No	X co-ordinate	Y co-ordinate	Demand	Customer No	X co-ordinate	Y co-ordinate	Demand
1	*	*	0	1	*	*	0
2	493	256	1	2	493	256	7
3	444	295	1	3	444	295	8
4	319	266	1	4	319	266	10
5	384	291	1	5	384	291	9
6	298	266	1	6	298	266	5
7	373	294	1	7	373	294	9
8	299	269	1	8	299	269	5
9	393	318	1	9	393	318	9
10	383	316	1	10	383	316	8
11	466	372	1	11	466	372	10
12	435	362	1	12	435	362	9
13	416	352	1	13	416	352	10
14	364	321	1	14	364	321	7
15	386	350	1	15	386	350	5
16	435	387	1	16	435	387	6
17	496	432	1	17	496	432	5
18	429	388	1	18	429	388	10
19	488	442	1	19	488	442	8
20	473	445	1	20	473	445	7
21	395	377	1	21	395	377	8
22	445	427	1	22	445	427	5
23	318	313	1	23	318	313	6
24	424	417	1	24	424	417	7
25	413	408	1	25	413	408	10
26	382	393	1	26	382	393	6
27	430	495	1	27	430	495	9
28	394	448	1	28	394	448	7
29	410	500	1	29	410	500	8
30	360	434	1	30	360	434	10
31	342	406	1	31	342	406	7
32	315	360	1	32	315	360	7
33	292	361	1	33	292	361	8
34	285	364	1	34	285	364	6
35	282	358	1	35	282	358	9
36	315	488	1	36	315	488	9
37	265	479	1	37	265	479	8
38	258	403	1	38	258	403	5
39	254	399	1	39	254	399	9
40	247	364	1	40	247	364	9

Notes. (*)Location of the depot is to be taken from Table A.1. The bold value indicates the better solution between the two methods

Appendix B.

Initial solutions provided by the modified sweep algorithm to Arnold *et al.* [3] test instances L1 and A2 are shown below. To prevent cluttering of picture the edges connecting the depot to the last customers are not displayed. Figure B.1 has centrally located depot whereas Figure B.2 has eccentrically located depot.

Figure B.1. The Initial solution for L1 (3000 customers).

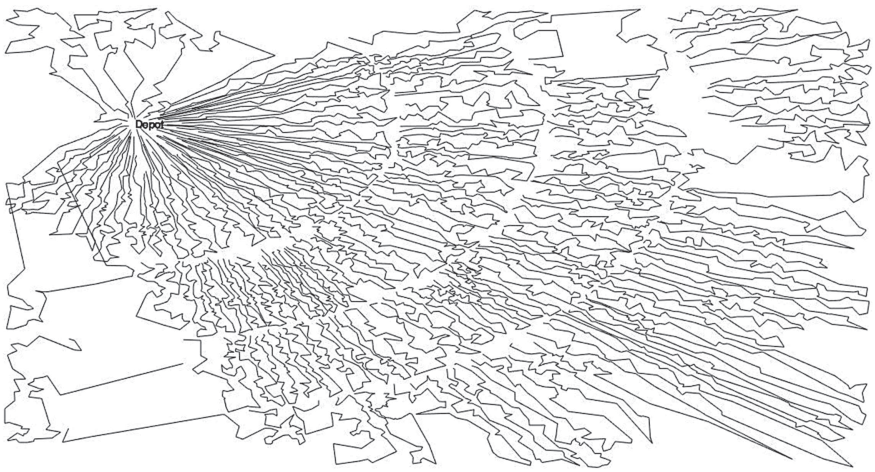

Figure B.2. The Initial solution for A2 (7000 customers).

Acknowledgements. We extend our sincere thanks to the anonymous reviewers for providing valuable remarks and helping us to improve our work.

References

[1] F. Arnold and K. Sörensen, What makes a VRP solution good? The generation of problem-specific knowledge for heuristics. *Comput. Oper. Res.* **106** (2019) 280–288.

[2] F. Arnold, I. Cardenas, K. Sörensen and W. Dewulf, Simulation of B2C e-commerce distribution in Antwerp using cargo bikes and delivery points. *Eur. Trans. Res. Rev.* **10** (2018) 1–13.

[3] F. Arnold, M. Gendreau and K. Sörensen, Efficiently solving very large-scale routing problems. *Comput. Oper. Res.* **107** (2019) 32–42.

[4] J. Brandão, A tabu search algorithm for the open vehicle routing problem. *Eur. J. Oper. Res.* **157** (2004) 552–564.

[5] G. Clarke and J.W. Wright, Scheduling of vehicles from a central depot to number of delivery points. *Oper. Res.* **12** (1964) 568–581.

[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, Introduction to Algorithms. MIT Press (2009).

[7] G.B. Dantzig and J.H. Ramser, The truck dispatching problem. *Manage. Sci.* **6** (1959) 80–91.

[8] T. Doyuran and B. Çatay, A robust enhancement to the Clarke–Wright savings algorithm. *J. Oper. Res. Soc.* **62** (2011) 223–231.
[9] R. Elshaer and H. Awad, A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Comput. Ind. Eng.* **140** (2020) 106242.
[10] B.E. Gillett and L.R. Miller, A heuristic algorithm for the vehicle dispatch problem. *Oper. Res.* **22** (1974) 340–349.
[11] V.C. Karels, L.P. Veelenturf and T. Van Woensel, An auction for collaborative vehicle routing: models and algorithms. *EURO J. Transp. Logistics* **9** (2020) 100009.
[12] J. Kytöjoki, T. Nuortio, O. Bräysy and M. Gendreau, An efficient variable neighborhood search heuristic for very large-scale vehicle routing problems. *Comput. Oper. Res.* **34** (2007) 2743–2757.
[13] G. Laporte, The Vehicle Routing Problem: an overview of exact and approximate algorithms. *Eur. J. Oper. Res.* **59** (1992) 345–358.
[14] MATLAB, version 9.0.0.341360 (R2016a). The MathWorks Inc., Natick, MA (2016).
[15] PassMark Software, Cpu benchmarks. Accessed: 2021-04-08 (2018). https://www.cpubenchmark.net/.
[16] Test data source. http://neo.lcc.uma.es/vrp/vrp-instances/.
[17] P. Toth and D. Vigo, editors, The Vehicle Routing Problem. Siam Monographs on Discrete Mathematics and Applications, Philadelphia, Pennsylvania (2001).
[18] A. Van Breedam, Comparing descent heuristics and metaheuristics for the vehicle routing problem. *Comput. Oper. Res.* **28** (2001) 289–315.
[19] B.R. Vangipurapu, R. Govada and N.R. Kandukuri, A new heuristic for solving vehicle routing problem with capacity constraints. *J. Adv. Eng. Res.* **6** (2019) 64–70.
[20] B.R. Vangipurapu, R. Govada and N.R. Kandukuri, A new heuristic for solving open vehicle routing problem with capacity constraints. In: Innovative Product Design and Intelligent Manufacturing Systems. Springer, Singapore (2020) 897–906.
[21] https://www.bigbasket.com/
[22] https://www.routing-solver.com/