Applied Soft Computing 110 (2021) 107655

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Hybrid particle swarm optimization algorithm for solving the clustered vehicle routing problem

Md. Anisul Islam [a,*], Yuvraj Gajpal [b], Tarek Y. ElMekkawy [c]

[a] Department of Mechanical Engineering, Room E2-327, EITC Building, University of Manitoba, 75A Chancellor Drive, Winnipeg, Manitoba, R3T 5V6, Canada
[b] Department of Supply Chain Management, 631-181 Freedman Crescent, Asper School of Business, University of Manitoba, Winnipeg, Manitoba, R3T 5V4, Canada
[c] Department of Mechanical and Industrial Engineering, Qatar University, P.O. Box 2713, Doha, Qatar

ARTICLE INFO

Article history:
Received 18 September 2019
Received in revised form 21 June 2021
Accepted 23 June 2021
Available online 30 June 2021

Keywords:
Clustered vehicle routing problem (CluVRP)
Particle swarm optimization (PSO)
VNS
Hybrid metaheuristic

ABSTRACT

This paper considers a variant of the classical capacitated vehicle routing problem called clustered vehicle routing problem (CluVRP). In CluVRP, customers are grouped into different clusters. A vehicle visiting a cluster cannot leave the cluster until all customers in the same cluster have been served. Each cluster and customer have to be served only once. A new hybrid metaheuristic, combining the particle swarm optimization (PSO) and variable neighborhood search (VNS) for the specific problem, is proposed to solve the CluVRP. In the hybrid PSO, the basic PSO principle ensures the solution diversity and VNS ensures solution intensity to bring the solution to the local optima. Extensive computational experiments have been performed on numerous benchmark instances with various sizes obtained from the CluVRP literature to evaluate the performance of the proposed hybrid PSO. The obtained results of the proposed algorithm are compared with the results found in the literature to validate the effectiveness of the proposed hybrid PSO. The proposed algorithm is proven to be superior to the state-of-the-art algorithms on the CluVRP. The proposed algorithm obtains 138 new best-known solutions among the 293 benchmark instances.

1. Introduction

The typical vehicle routing problem (VRP) is a logistic distribution problem. The VRP aims to obtain a list of least-cost vehicle routes serving many geographically scattered customers under various supply and demand constraints. It is a combinatorial optimization problem that requires exponential computational time to be optimized. This study presents a variant of the capacitated vehicle routing problem (CVRP) called the Clustered VRP (CluVRP). In CluVRP, customers are partitioned into predefined groups called clusters. The customers corresponding to a single cluster must all be visited by the same vehicle before it leaves the cluster. The notion of clustering in VRP has been well known due to its economic implications and its reduced complexity in modeling and solving a great range of real-world applications [1]. The CluVRP is a generalized form of the CVRP. As the CVRP is proven to be an NP-hard problem, the CluVRP is also NP-hard [2].

There are two variants of CluVRP such as CluVRP with strong cluster constraints (CluVRP) and CluVRP with weak cluster constraints (SoftCluVRP). In the CluVRP, all customers belong to the same cluster must be visited uninterruptedly by the same vehicle. Vehicles are not permitted to enter and leave clusters several times while serving the customers. In the SoftCluVRP, though customers belong to a specific cluster are visited by the same vehicle, but vehicles are allowed to leave and enter clusters many times during their trip in the route. This paper studies a CluVRP with strong cluster constraints referred as CluVRP. The CluVRPs are explored in many studies such as [1,3–10] and SoftCluVRPs are studied in the works of [8,10,11]. Most of the studies in the literature proposed metaheuristics based solution approaches.

The comprehensive CluVRP introduced by Sevaux and Sörensen [12] focused on a real-world parcel delivery problem in courier companies. The consignment parcels were arranged into bins corresponding to the specific delivery zones. The consignees belonged to the same zone designated as a cluster. The CluVRP can also arise in many scenarios such as transporting elderly people when the customers prefer to move with friends or neighbors, providing service to gated communities, collecting urban solid waste, providing the services of common repairmen, delivering

* Corresponding author.
E-mail addresses: islamma@myumanitoba.ca (M.A. Islam), Yuvraj.Gajpal@umanitoba.ca (Y. Gajpal), tmekkawy@qu.edu.qa (T.Y. ElMekkawy).

https://doi.org/10.1016/j.asoc.2021.107655
1568-4946/

healthcare providing service in both precedence ordered multitude of emergency environments and in logistics operations in an order-picking [1,13].

The key contribution of this paper is to design a hybrid metaheuristic for solving a CluVRP. The proposed metaheuristic algorithm is based on the combination of particle swarm optimization (PSO) and the CluVRP specific variable neighborhood search (VNS). The VNS helps to discover the local optimal solution of the search region. In the literature, VNS has been implemented mostly to improve particle solutions. But, this paper uses VNS to improve personal best solutions along with the global best solution by using improvement scheme. The contribution also includes the use of new features in the PSO algorithm such as the use of two types of particles. This hybrid PSO is targeted to achieve a better quality solution for the CluVRP problem.

The rest of the paper is structured as herein described. The literature of CluVRP is reviewed in Section 2. In Section 3, the CluVRP is defined, and its mathematical formulation is presented. The proposed hybrid PSO is discussed in detail in Section 4. The computational results are reported in Section 5. Finally, the conclusion is stated in Section 6.

2. Literature review

Sevaux and Sörensen [12] proposed a mixed integer linear programming formulation of a CluVRP for a distribution operation in a famous courier services company. Barthélemy et al. [3] designed a heuristic for a CluVRP, where a big value was added to all inter-cluster edges to convert the CluVRP into a CVRP and solve it by simulated annealing method. Pop et al. [4] presented two integer programming based exact solution approaches for a CluVRP. In another study, based on the integer programming formulation, two exact solution approaches such as branch-and-cut and branch-and-cut-and-price were presented by Battarra et al. [5]. A new hybrid algorithm based on the genetic algorithm combined with simulated annealing was developed to solve a CluVRP by Marc et al. [6]. Vidal et al. [7] proposed two hybrid metaheuristics for solving a CluVRP. The first one was based on the iterated local search (ILS) algorithm designed by Subramanian [14]. The second one was based on the unified hybrid genetic search (UHGS). An approximate two-level optimization technique was suggested to solve a CluVRP in Expósito-Izquierdo et al. [1]. Defryn and Sörensen [8] developed an efficient two-level variable neighborhood search (VNS) heuristic to solve a CluVRP. A study by Pop et al. [9] addressed a unique two-level optimization approach to solve a CluVRP. The problem was divided into two sub-problems: the upper-level (cluster) sub-problem and the lower-level (customer) sub-problem. In the approach, the route visiting the clusters was obtained by a genetic algorithm, then, the customers' visiting order within the clusters was determined by the Concorde TSP solver. The recent trend of metaheuristics shows its hybridization for performance improvement. Recently, Hintsch and Irnich [10] presented a large multiple neighborhood search (LMNS) based metaheuristic algorithm for the CluVRP. The problem was broken down into three sub-problems: assigning clusters to the routes, intra-cluster routing, and routing the clusters. In the LMNS approach, multiple destroy and repair moves for clusters were used first, then a VND-based local search improvement scheme was employed for further optimization. Most of the hybridization is done through the use of local search schemes. This observation motivated us to hybridize the PSO to improve its performance in this study.

Our current paper proposes a solution approach based on classical particle swarm optimization (PSO) combined with a variable neighborhood search (VNS) for solving a CluVRP. The PSO is a population-based combinatorial optimization technique originally familiarized in Eberhart and Kennedy [15]. The technique has been inspired by social collective behaviors seen in many natural swarms such as bird flocking, fish schooling, and human beings. The hybridized PSO approaches were used in many variants of VRPs such as hybridized with local searches in Ai and Kachitvichyanukul [16]; with local searches and path relinking strategy in Marinakis et al. [17] and with modified local search in Norouzi et al. [18]. Additionally, an adaptive PSO algorithm was built to solve an integrated quay crane and yard truck scheduling problem successfully [19]. Dridi et al. [20] developed a new PSO based solution approach for an optimization problem of multi-depots pick-up and delivery problems with time windows and multi-vehicles. It is clear from the literature that the efficiency of PSO can be improved by its hybridizing.

PSO algorithm has many advantages such as few parameters to tune, easy to implement, and requires less server memory compared to other metaheuristics. PSO algorithm is successfully utilized and found as a validated solution method for many combinatorial optimization problems in the areas of transport, manufacturing, and scheduling problems [21–23]. The VNS uses multiple local search methods to obtain the local optimum. The PSO has the ability to diversify the solution while VNS has the ability to intensify the solutions. These strengths are combined in our proposed metaheuristic algorithm.

The variable neighborhood search (VNS) was first introduced by Mladenovic and Hansen [24] to solve a traveling salesman problem in 1997. Usually, a VNS is used as a local search algorithm to obtain the local best solution [25]. The VNS is also a widely used heuristic search method in VRPs [26]. Many studies found using the VNS with the PSO for solving several optimization problems, where PSO solution used as a global search algorithm. Marinakis et al. [27] generated a hybrid PSO metaheuristic to solve a CVRP, by producing an initial solution from a greedy randomized adaptive search procedure and by improving the solution further by a VNS algorithm. Goksal et al. [28] introduced a hybrid metaheuristic based on PSO and variable neighborhood descent (VND), a lower-level VNS, to solve a vehicle routing problem with simultaneous pickup and delivery. Besides, Marinakis et al. [29] proposed a multi-adaptive PSO solution approach for a vehicle routing problem with time windows, where the PSO solutions were improved by applying VNS for each particle in the swarm. Zou et al. [30] presented a novel PSO algorithm hybridized with VNS to solve a multi-objective VRP with pickup and delivery problems with time windows. Zhang et al. [31] designed a hybrid solution based on VNS integrated with binary PSO to solve a location-routing problem (LRP). Marinakis [32] hybridized a PSO combined with a VNS for solving a capacitated LRP. In another study, Moghaddam et al. [33] used VNS in an advanced PSO based solution approach to solve a vehicle routing problem with uncertain demands. A novel decoding algorithm was used to increase the efficiency of the solution approach. The decoding was designed for generating vehicle routes and updating particle values. Moreover, due to the dominant behavior of PSO in producing a strong global solution and VNS having the advantages of generating the best local solution, PSO and VNS have also been used widely in job scheduling problems [34]. Liu et al. [35] used a hybrid metaheuristic based on PSO combined with VNS to solve a multi-objective flexible job-shop scheduling problem. In additional work [36], it was shown that a simpler VNS algorithm without hybridization with PSO produces a better quality solution with shorter CPU time than a hybrid PSO with a VNS algorithm for the job-shop scheduling problems. Furthermore, a hybrid metaheuristic combining a PSO and VNS algorithm was proposed for solving an unconstrained global optimization problem in Ali et al. [37]. In the study, the PSO was used to perform a wider diversification and deep intensification in the solution space, and VNS was used as a local search algorithm. Furthermore, a PSO-based hybrid metaheuristic was designed for permutation flow shop scheduling problems [38]. In the work, a PSO algorithm was incorporated with a stochastic VNS, a variant of VNS proposed in [32], hybridized with simulation annealing to enhance the exploration ability of PSO in the solution approach. Gumaida and Luo [39] developed a new hybrid optimization technique based on PSO combined with a VNS to enhance the localization process in wireless sensor networks. Marinakis et al. [40] designed a hybrid PSO incorporated with VNS to solve a constrained shortest path problem. Cai et al. [41] proposed a hybrid PSO based solution approach where the PSO was hybridized by VNS to solve a VRP with speed variables through reduced carbon emissions in the routes. A railway cargo transportation problem was studied by proposing a solution method based on PSO with VNS in Nie et al. [21]. Ranjbar and Saber [42] designed a VNS and modified PSO based solution approaches for a transshipment scheduling problem of multi-products at a single station. Islam et al. [43] presented a PSO and VNS based solution approach for solving a mixed fleet green logistics problem under carbon emission cap. Motivated by this observation, this paper embeds the VNS with the PSO to obtain a good quality solution of the CluVRP.

3. Problem definition of CluVRP

The CluVRP can be defined on an undirected graph $G = (V, E)$, where $V = \{0, 1, 2, \ldots, n\}$, a set of nodes (vertices) including the customers $\{1, 2, \ldots, n\}$, E is the set of arcs linking each pair of nodes (i, j) in V, and a depot 0. A homogeneous fleet of vehicles is situated at the depot, where the vehicles start and end their trip while serving the customers.

Parameters

n	Total number of customers
c	Total number of clusters
0	The depot
n_l	The number of customers for the lth cluster
m	Individual vehicle
M	Total number of vehicles available in the network
r	Individual cluster (mutually exclusive non-empty disjoint), $r \in R$
R	Group of the clusters
d_r	Demand of cluster, r (aggregated over all customers in the cluster), $d_r > 0$
tc_{ij}	The nonnegative travel cost for the edges from i to j, $(i, j) \in E$
Q	Maximum loading capacity of each vehicle, $Q > 0$
C_r	The group of customers within a cluster, $C_r = \{i \in n : r_i = r\}, \forall r \in R$
V	Set of vertices
S	Any subset of customer nodes, $\{1, 2, \ldots, n\}$
$\delta^+(S)$	Set of edges (i, j) where $i \in S$ and $j \in V \backslash S$
$\delta^-(S)$	Set of edges (i, j) where $i \in V \backslash S$ and $j \in S$

The binary decision variables are:

$$x_{ijm} = \begin{cases} 1 & \text{vehicle } m \text{ travels from customer } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

$$y_{im} = \begin{cases} 1 & \text{customer } i \text{ is served by vehicle } m \\ 0 & \text{otherwise} \end{cases}$$

The CluVRP can be formulated as follows:

$$\text{Minimize} \quad \sum_{(i,j)\in E} \sum_{m=1}^{M} tc_{ij} x_{ijm} \tag{1}$$

s.t.,

$$\sum_{m=1}^{M} y_{im} = 1 \qquad \forall i \in \{1, 2, \ldots\ldots n\} \tag{2}$$

$$\sum_{m=1}^{M} y_{0m} \leq M \tag{3}$$

$$y_{0m} \geq y_{im} \qquad \forall m \in \{1, 2, \ldots, M\}, \forall i \in \{1, 2, \ldots\ldots n\} \tag{4}$$

$$\sum_{j=1}^{n} x_{ijm} = \sum_{j=1}^{n} x_{jim} = y_{im} \qquad \forall m \in \{1, 2, \ldots, M\}, \forall i \in \{0, 1, 2, \ldots, n\} \tag{5}$$

$$\sum_{i=0}^{n} d_i y_{im} \leq Q \qquad \forall m \in \{1, 2, \ldots, M\} \tag{6}$$

$$\sum_{i\in S} \sum_{j\in V\backslash S} x_{ijm} \geq y_{hm} \qquad \forall S \subseteq \{1, 2, \ldots, n\}, h \in S, m \in \{0, 1, 2, \ldots, M\} \tag{7}$$

$$\sum_{(i,j)\in\delta^+(C_r)} \sum_{m=1}^{M} x_{ijm} = \sum_{(i,j)\in\delta^-(C_r)} \sum_{m=1}^{M} x_{ijm} = 1 \qquad \forall r \in R \tag{8}$$

$$\sum_{i=1}^{n} d_i y_{im} \geq \sum_{i=1}^{n} d_i y_{im+1} \qquad \forall m \in \{1, 2, \ldots, M-1\} \tag{9}$$

$$x_{ijm} \in \{0, 1\} \qquad \forall (i, j) \in E, \forall m \in \{1, 2, \ldots, M\} \tag{10}$$

$$y_{im} \in \{0, 1\} \qquad \forall i \in \{0, 1, 2, \ldots, n\}, \forall m \in \{1, 2, \ldots, M\} \tag{11}$$

The objective of minimizing the total travel cost is determined by Eq. (1). Constraint (2) guarantees that each customer is visited exactly once. Constraint (3) assures that the number of vehicles used does not exceed the number of available vehicles. Constraint (4) enforces the rule that each vehicle in the route should visit the depot. If a vehicle m does not visit the depot then it should not visit any customer. Constraint (5) ensures that the arriving and the departing vehicle is the same for a given customer. Constraint (6) states the maximum loading capacity of the vehicles is satisfied. Constraint (7) represents the sub-tour elimination constraint. Constraint (8) ensures that each cluster can be visited exactly once by a unique vehicle. Constraint (9) is the inequality ensuring partial symmetry.

4. Proposed hybrid PSO for the CluVRP

The proposed approach is a hybrid PSO algorithm that combines the standard PSO and the VNS. The structure of VNS in the proposed approach is inspired by a study by Vidal et al. [7]. Generally, the performance of the PSO is largely affected by the accuracy of the problem mapping. Thus, the PSO is modified in accordance with problem specifications in this study. The main features of the proposed hybrid PSO are the use of two types of particles representing clusters and customers, and the use of an improvement scheme for the personal best solutions. The pseudo code of the proposed hybrid PSO is shown in Algorithm 1.

The proposed hybrid PSO uses the following definition:

α_{il}	Current cluster position value of ith particle in lth dimension
γ_{ij}	Current customer position value of ith particle in jth dimension
β_{il}	Current cluster velocity value of ith particle in lth dimension
δ_{ij}	Current customer velocity value of ith particle in jth dimension
f_i	Fitness function of particle, i

α_{il}^b	Personal best cluster position value found so far for the ith particle in the lth dimension
γ_{ij}^b	The personal best customer position value found so far for the ith particle in the jth dimension
f_i^b	Fitness function of best particle, i
α_l^*	Global best cluster position value found in the lth dimension
γ_j^*	Global best customer position value found in the jth dimension
f^g	Fitness function of global best particle
w	Inertia coefficient
c_1	Cognitive coefficient
c_2	Social coefficient
r_1, r_2	Independent random numbers
K	Total number of the particles
X	Position matrix for customer swarm
Y	Position matrix for cluster swarm
U	Velocity matrix for customer swarm
V	Velocity matrix for cluster swarm
X^b/X^G	Customer personal best/global best position value for swarm
Y^b/Y^G	Cluster personal best/global best position value for swarm
S^b	Personal best solution for swarm

Algorithm 1: Pseudo code of the proposed algorithm

1: *Initialization*
2: **Set parameters:** $w = 0.7$, $c_1 = c_2 = 2$, $r_1 = r_2 = 0.5$, $K = n/4$.
3: **Initialize** position matrix X, Y and velocity matrix U , V
4: **Initialize** the personal best fitness vector f^b
5: **Initialize** the global best fitness vector f^g
6: *Main phase*
7: Do while
8: $S \leftarrow$ GenerateCluVRPSolution (X, Y, U, V)
9: $S \leftarrow$ VNS (S)
10: **Update** personal best matrix X^b, Y^b, fitness vector f^b, and personal best solution matrix S^b
11: **Improve** personal best matrix using improvement scheme $S^b \leftarrow$ Improvement scheme (S^b)
12: **Update** the best particle X^G, Y^G and fitness vector f^g
13: **Update** (X, Y, U, V)
14: *End Do*

4.1. Initialization phase

The position and velocity vectors are initialized as follows:

$$\alpha_{il} = \alpha_{min} + (\alpha_{max} - \alpha_{min}) * U(0, 1) \quad \forall i \in \{1, 2, \ldots K\}, \forall l \in \{1, 2, \ldots c\} \tag{12}$$

$$\gamma_{il} = \gamma_{min} + (\gamma_{max} - \gamma_{min}) * U(0, 1) \quad \forall i \in \{1, 2, \ldots K\}, \forall j \in \{1, 2, \ldots n\} \tag{13}$$

$$\delta_{il} = \delta_{min} + (\delta_{max} - \delta_{min}) * U(0, 1) \quad \forall i \in \{1, 2, \ldots K\}, \forall l \in \{1, 2, \ldots c\} \tag{14}$$

$$\beta_{il} = \beta_{min} + (\beta_{max} - \beta_{min}) * U(0, 1) \quad \forall i \in \{1, 2, \ldots K\}, \forall j \in \{1, 2, \ldots n\} \tag{15}$$

Where $\alpha_{max} = \gamma_{max} = \delta_{max} = \beta_{max} = 4$; $\alpha_{min} = \gamma_{min} = \delta_{min} = \beta_{min} = -4$.

Here, $U(0, 1)$ represents a uniform random number generated between 0 and 1. The personal best fitness vector for the particle, i and fitness vector of a global particle are initialized as infinity.

$$f_i^b = \infty \quad \forall i \in \{1, 2, \ldots, K\}$$
$$f^g = \infty$$

Table 1
An instance with 6 clusters with their position values and demands in any iteration, t.

Clusters	1	2	3	4	5	6
Position values, α_{il}	1.99	3.67	−2.25	2.50	−0.09	1.08
Cluster demand, d_r	45	10	25	15	25	30

4.2. Mapping position vectors to generate CluVRP solution

The PSO usually maps the position values of the particles to generate the solution for a given problem. The position values are used to generate the CluVRP solution ($S \leftarrow$ GenerateCluVRP-Solution (X, Y, U, V)) as stated in line 8 in algorithm 1. The two-phase approach is used in many studies to generate CluVRP solutions [8,9]. In the proposed PSO, the solution is generated in two phases. In the first phase, the cluster route for the vehicles is generated from the position values of clusters α_{il}, while the customer route for each cluster is generated in the second phase from the position values of customers γ_{ij}.

4.2.1. Generating cluster route

The generation of the cluster route starts with the empty trip for each vehicle, where the vehicles start and finish their trip at the depot. The clusters are iteratively added to the vehicle routes to find the complete solution. Firstly, the clusters with the highest position values are chosen for inclusion in the vehicle route, then the chosen cluster is inserted into the vehicle routes by using the cheapest insertion method. However, cluster insertion might face a situation where no vehicle has enough capacity for inserting a chosen cluster. In this situation, a tabu search based searching method is used to insert the chosen cluster. This method tries to maximize the available vehicle capacity using swap (1,1) and shift (1,0) neighborhood move. The selected swap move between clusters i and j is forbidden for next $U\left(\frac{c^2}{8}, \frac{c^2}{4}\right)$ iterations. Similarly in shift (1,0) move, insertion of cluster i is forbidden in cluster j for next $U\left(\frac{c*v}{8}, \frac{c*v}{4}\right)$ iterations.

To understand the mapping procedure for cluster routes, consider an instance with 6 clusters and 2 vehicles with a vehicle capacity of 80. In any iteration t, consider the following cluster position values for ith particle in lth dimension as shown in Table 1. In this example, 6 different dimensions represent 6 different clusters. Since different dimensions are associated with different clusters, we refer the cluster position value of lth dimension as a position value of lth cluster.

In the mapping, clusters are arranged in non-increasing order of their position values. The resultant order is π = 2-4-1-6-5-3. The two vehicles routes initially start with the first two clusters from π. The initial route is {0-2-0; 0-4-0} and the remaining vehicle capacity for each vehicle is updated accordingly. Then, cluster 1 is chosen for insertion on vehicle routes. The insertion cost (i.e., increase in total route length) of cluster 1 is evaluated on every position of two routes {0-2-0; 0-4-0}. Suppose the cheapest insertion of cluster 1 is obtained by inserting at position 3 of vehicle 2. Then the new route is {0-2-0; 0-4-1-0}. In the next iteration, cluster 6 is chosen for insertion. Suppose the cheapest insertion of cluster 6 is obtained by inserting at position 3 of vehicle 1. Then the new route is {0-2-6-0; 0-4-1-0}. In the next iteration, cluster 5 is chosen for insertion. Suppose the cheapest insertion of cluster 5 is obtained by inserting at position 2 of vehicle 1. Then the new route is {0-5-2-6-0; 0-4-1-0}. At this point, the remaining capacities for the two vehicles are 15 and 20. But the demand for unassigned cluster 3 is 25 and no vehicle has the required capacity to accommodate cluster 3. In this situation, we use the tabu search with swap (1, 1) and shift (1, 0) with the objective function of maximizing the remaining vehicle

Table 2
A vehicle route of 2 clusters with their customers and position values in any iteration, t.

Cluster 1	Customers	10	4	7	
	Position values, γ_{ij}	2.74	3.44	−1.81	
Cluster 3	Customers	2	17	9	5
	Position values, γ_{ij}	2.03	−0.96	1.60	1.87

capacity. The tabu search is stopped when the objective function (i.e., remaining vehicle capacity) becomes at least 25. Let assume the tabu search finds the new routes as {0-4-5-2-6-0; 0-1-0}. The remaining capacities are 0 and 35 for vehicle 1 and vehicle 2 respectively. Finally, cluster 3 is chosen for insertion. Suppose the cheapest insertion of cluster 3 is obtained by inserting at position 3 on vehicle 2. Consequently, the final routes is {0-4-5-2-6-0; 0-1-3-0}.

4.2.2. Generating customer route

Once the cluster routes are constructed, a sequence of the customers for each cluster is generated to find the complete solution of the CluVRP. The sequence of the customers is generated by selecting customers similar to the clusters routes generation method described in Section 4.2.1.

To understand the generation of customer routes, consider a cluster route in a vehicle is {0-1-3-0}. Suppose there are 3 customers and 4 customers in cluster 1 and cluster 3 respectively as shown in Table 2. In any iteration t, consider the following customer position values for ith particle in jth dimension as stated in Table 2. Since different dimensions are associated with different customers, we refer the position value of jth dimension as a position value of jth customer.

In the customer routes generation, customers are arranged in non-increasing order of their position values. The resultant customer order for cluster 1 is $\tau = 4 - 10 - 7$ and cluster 3 is $\tau = 2 - 5 - 9 - 17$. The complete customer route of the vehicle is {0-4-10-7-2-5-9-17-0}. The travel cost (i.e., objective function value) of the route is the sum of the travel costs of all customers in the route.

4.3. Variable neighborhood search (VNS) for CluVRP

The proposed PSO considers the position vector as a region instead of a particular point. The solution generated in the mapping phase represents one solution in the region, which might not be the best solution of the region. Therefore, the VNS is employed to achieve the local optima. The VNS procedure consists of three local search moves, which are inter-route search, intra-route search, and intra-cluster search. Both the inter-route search and intra-route search focus on the cluster level; whereas, the intra-cluster search focuses on the customer level. The neighborhood operators which are used at cluster level: shift, shift2, swap, swap (2,1), swap (2,2), and 2-opt in the inter-route search; and shift, or-opt2, or-opt3, 2-opt, and swap in the intra-route search. The NL_c is the list of all inter-route neighborhood searches. The neighborhood operators that are adopted for intra-cluster search (customer level) are shift, 2-opt, and swap; these explore all moves within each cluster. The detail of the operators can be found in the literature [7,14,44]. The structure of each operator is shown in Figs. 1 and 2. The first move adoption strategy is adopted for all local search moves. In this strategy, the solution is updated whenever an improved solution is found. In all local searches, each neighborhood move is selected only once for possible improvement instead of iterative strategy. The overall structure of the VNS is shown in Algorithm 2.

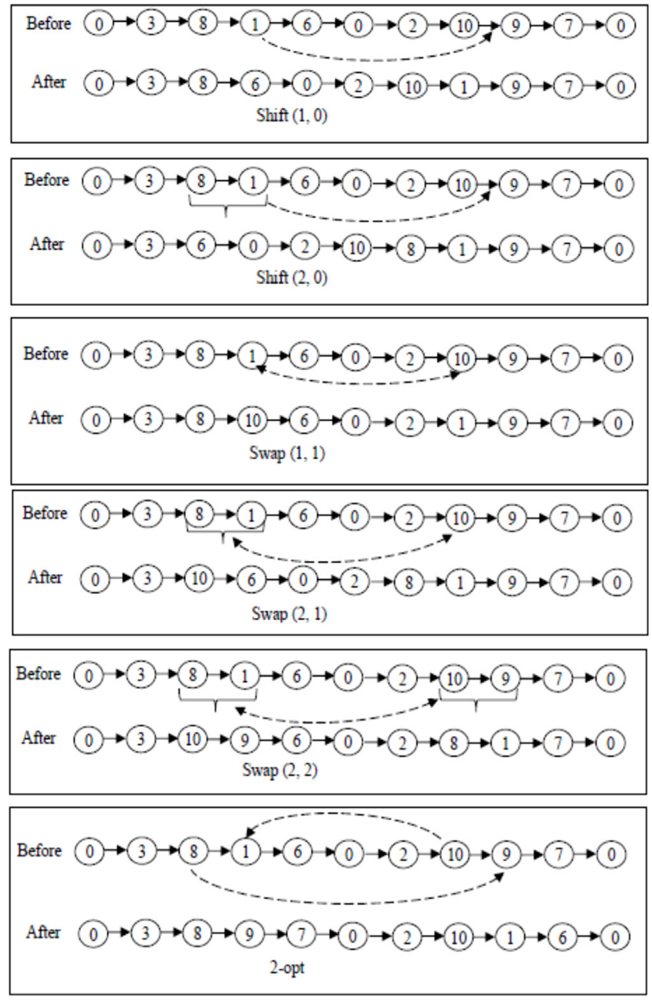

Fig. 1. Inter-route neighborhood search operators.

Algorithm 2: Variable neighborhood search (VNS)

Method VNS:
Initial solution, s;
Do
    Set previous solution, $s^{initial} = s$;
    List (NL_c) for the inter-route search;
        While $NL_c \neq \emptyset$
            Choose randomly a neighborhood from NL_c;
            Find best $s^{\neg}$ of $s \in$ neighbourhood;
            if $f(s^{\neg}) < f(s)$
                $s \leftarrow s^{\neg}$;
                $s \leftarrow$ Intra-route search(s)
                Update NL_c;
            else
                Remove neighbourhood from NL_c;
        end While
    $s \leftarrow$ Intra-cluster search (s);
While ($s < s^{initial}$)
return s;
end ***VNS;***

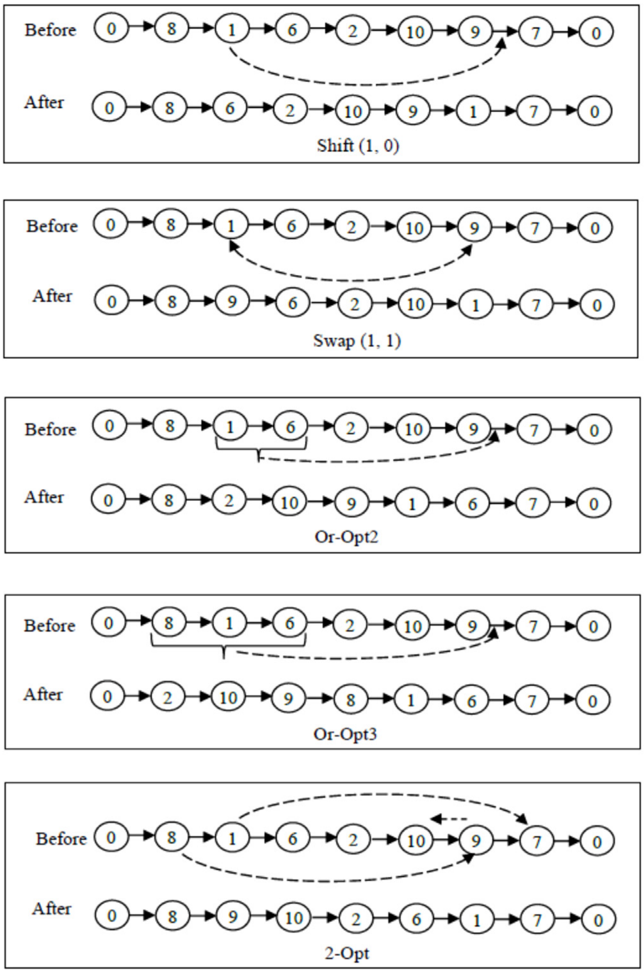

Fig. 2. Intra-route and inter-cluster neighborhood search operators.

4.4. Updating position and velocity vectors

The personal best position value for each particle is updated if the current solution obtained is better than the current personal best solution. Similarly, the global best value is updated if the new best solution is found better than the current global best value.

The velocity and position vectors are updated as follows:

$$\delta_{il} = w\delta_{il} + c_1 r_1 \left(\alpha_l^p - \alpha_{il}\right) + c_2 r_2 \left(\alpha_l^* - \alpha_{il}\right) \quad \forall i \in \{1, 2, \ldots, K\}, \forall l \{1, 2, \ldots, c\} \tag{16}$$

$$\beta_{il} = w\beta_{il} + c_1 r_1 \left(\gamma_j^p - \gamma_{il}\right) + c_2 r_2 \left(\gamma_j^* - \gamma_{il}\right) \quad \forall i \in \{1, 2, \ldots, K\}, \forall j \{1, 2, \ldots, n\} \tag{17}$$

$$\alpha_{il} = \alpha_{il} + \delta_{il} \qquad \forall i \in \{1, 2, \ldots, K\}, \forall l \{1, 2, \ldots, c\} \tag{18}$$

$$\gamma_{il} = \gamma_{il} + \beta_{il} \qquad \forall i \in \{1, 2, \ldots, K\}, \forall j \{1, 2, \ldots, n\} \tag{19}$$

4.5. Improvement scheme

The improvement scheme is used to improve the personal best solution. This is one of the new features of PSO used in this study. To our knowledge, this feature is not used in the existing literature of PSO. In the improvement scheme, at first, the solution is perturbed to generate a new solution. The perturbed solution is then optimized using the VNS scheme. A perturbation technique is implemented in both cluster and customer levels. In the perturbation scheme, firstly the Δ_1/Δ_2 number of clusters/customers are removed and then reinserting again using the cheapest insertion method. The structure of the improvement scheme is shown in Algorithm 3. The parameters Δ_1 and Δ_2 are randomly generated between [0.5c, 0.75c] and [0.5n_l, 0.75n_l] respectively.

Algorithm 3: Improvement scheme

```
Method Improvement scheme:
    Initial solution, s;
    s* ← Perturbation (s)
    s** ← VNS (s*)
    Update s
        if f(s**) < f(s)
            s = s**
return s;
end Improvement scheme;
```

4.6. Computational complexity of hybrid PSO

There are four main steps in the hybrid PSO algorithm- (1) sequence generation, (2) VNS method, (3) parameter update and (4) improvement scheme. The sequence generation step first creates route for clusters. The cluster route generation performs two sequential operations- (a) arranging clusters according to the position values, and (b) inserting clusters in partially generated routes. Both operations can be performed in $O\left(c^2\right)$ time, the complexity of the cluster route generation step remains $O\left(c^2\right)$. After generating cluster routes, the sequence generation step creates routes of the customer, which can be performed in $O\left(n^2\right)$ time. Since the cluster route generation and the customer route generation are performed sequentially, the complexity of sequence generation step becomes $O\left(c^2+n^2\right)$. Similarly, VNS method, parameter updating, and improvement scheme can be performed in $O\left(c^2+n^2\right)$ time. The four steps of the PSO are performed sequentially, therefore the complexity of one iteration of hybrid PSO remains $O\left(c^2+n^2\right)$.

5. Computational experiments

The proposed hybrid PSO algorithm is implemented using the C++ programming language to solve several benchmark datasets from the literature of CluVRP. The experiments are run on a Linux server with four 2.1 GHz processors with 16-core each and a total of 256 GB of RAM.

5.1. The benchmark CluVRP instances

The performance of the hybrid-PSO is tested on the CluVRP benchmark instances composed of 20 major customer groups named as, A, B, P, M, and Golden instances (Golden 1 to Golden 20) with a total of 298 individual instances. These CluVRP instances are originally adopted from the GVRP instances by Bektas et al. [45]. The characteristics of the benchmark dataset are summarized in Table 3. The algorithms and their notations used in this study for the results reporting purpose are shown in Table 4.

The PSO parameters are set by performing sensitivity analysis using the problem instances of sets A, B, M, and P. We use PSO solution without VNS and without improvement scheme for 100 iterations to set the parameters. We start the sensitivity analysis with the parameter values found in the literature [16,17,27]. The parameter values are set one by one in the order of w, c_1, c_2, r_1, r_2, and K. A number of different alternative values for each parameter are tested as w = {0.5, 2}; c_1 = {2, 5}; c_2 = {2, 5};

Table 3
The summary of the benchmark instances.

Instance type	No. of instances	No. of customers	No. of clusters	Vehicle capacity (No. of vehicles)	Source
A	27	31–79	11–27	100 (2–5)	Bektas et al. [45]
B	23	30–77	11–23	100 (2–5)	Bektas et al. [45]
M	4	100–261	34–76	200 (3–8)	Bektas et al. [45]
P	24	15–100	6–51	35–400 (1–8)	Bektas et al. [45]
Golden	220	201–483	17–97	550–1000 (4–12)	Battarra et al. [5]

Table 4
The algorithms and their notations used in this study.

Notations	Algorithms
BC	The branch and cut method of Battarra et al. [5]
UHGS	The unified hybrid genetic search approach of Vidal et al. [7]
Two-level	The two level algorithm results of Expósito-Izquierdo et al. [1]
Two-level VNS	The two level variable neighborhood search results of Defryn and Sorensen [8]
Decomposition-based method	The decomposition method of Horvat-Marc et al. [6]
Two-level optimization	The two-level optimization approach by Pop et al. [9]
LMNS	The large multiple neighborhood search result of Hintsch and Irnich [10]
Hybrid PSO	The algorithm proposed in this paper

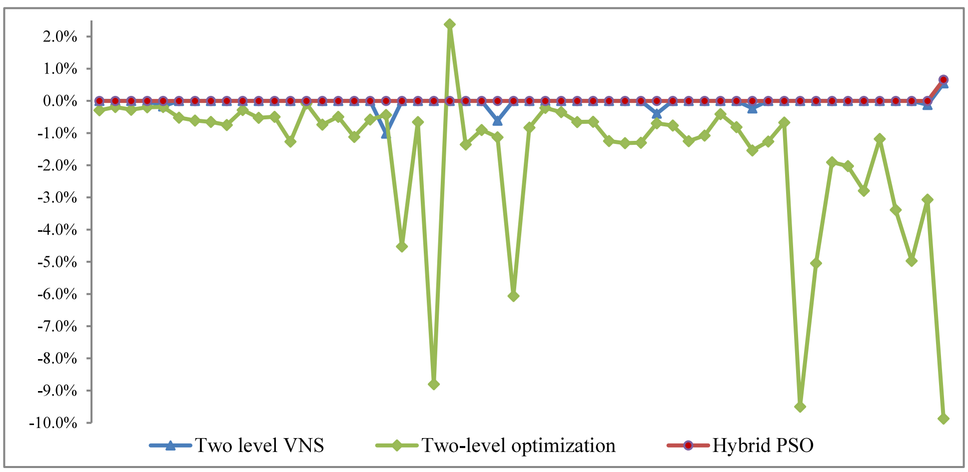

Fig. 3. Improvement% of the algorithms results for A, B, M instances.

r_1 ={0, 1}; r_2 = {0, 1}. Finally we set our best parameters as $w = 0.7$; $c_1 = c_2 = 2$; $r_1 = r_2 = 0.5$; $K = n/4$. We run the proposed hybrid PSO for each instance ten times with 100 iterations (i.e., algorithm termination criterion). The best result for each instance with average CPU time is obtained over ten runs. We observe that the improvement of results after 100 iterations is very marginal.

5.2. Performance evaluation of different algorithms

All the results in this study are evaluated by comparing the results reported by Battarra et al. [5] using the branch and cut (BC) algorithm to solve the CluVRP problem. They could not achieve the optimal solutions for all the problem instances but reported the best feasible upper bound solutions obtained during the execution of their algorithms. The solutions by Battarra et al. [5] are denoted by *UB*. Overall, the performance of the algorithms, including algorithms obtained from the literature, is evaluated by two criteria. The first criterion is that in how many instances does the algorithm finds a better solution than the upper bound, *UB* solution. It is reported in the tables under the "No. of improved *UB*". The second criterion is the improvement% of the algorithm compared to the *UB*. It is measured by Eq. (20), where *Sol* is used to denote the solutions found by the other algorithms. The improvement% of a group instance is reported as "improvement%" in the tables. In addition, the processing time (CPU time) is reported as t (s). The following formula is used to calculate improvement% from the *UB*.

$$Improvement\% = \frac{UB - Sol}{UB} \times 100 \quad (20)$$

Tables 5 and 6 show all the results of this study including reported results from the literature.

In the performance evaluation, the statistical tests, non-parametric Friedman test and post-hoc Bonferroni test are used to check any significant difference exists in the performance of algorithms. Friedman's test only reveals the difference among the results of different algorithms. The Bonferroni test is performed after Friedman's test to show which particular pair of algorithms is different from each other in comparison [46]. The statistical software IBM SPSS version 19 is used to run the Friedman and post-hoc Bonferroni test using $\alpha = 0.05$ as the level of significance.

5.2.1. Performance evaluation for A, B, M and P instances

Table 5 reports the results for the instances groups A, B, M, and P. The two-level VNS algorithm, decomposition-based method, two-level optimization, and the hybrid PSO are evaluated in the table. The comparison shows that all of the two-level VNS, the decomposition-based method, and the two-level optimization obtain the improved *UB* solution for one instance out of 75 instances; whereas, the hybrid PSO is capable of obtaining the improved *UB* solution for a total of 2 instances out of 78. In addition, the overall improvements obtained are −0.03%, −5.00%, and −1.7% respectively in the two-level VNS, decomposition-based method, and two-level optimization, which shows that all the two-level VNS, the decomposition-based method, and two-level optimization are inferior to BC solutions. In the case of the hybrid PSO solution, the overall improvement is found to be 0.05% compared to the BC solution, which also indicates that the hybrid PSO solution is superior to the two-level VNS by 0.08% (from −0.03% to 0.05%), decomposition-based method by 5.05% (from −5.00% to 0.05%), and to two-level optimization approach by 1.12% (from −1.7% to 0.05%). The average CPU time of hybrid PSO is 0.22 s, which is almost equal to the CPU time of the competitive algorithm two-level VNS (0.23 s). In the Friedman test, a significant statistical difference is found in comparing the performance of hybrid PSO with all algorithms (p values = 0.000). We also performed post-hoc Bonferroni tests between algorithms to check whether their results are statistically different or not. The p value of the pair-wise comparison Bonferroni test between hybrid PSO and decomposition-based method is 0.0020; between hybrid PSO and two-level optimization is 0.00007. These p values are less than the Bonferroni adjustment significant level 0.0125 and thus the null hypotheses are rejected. The rejection of the null hypotheses shows that the results of hybrid PSO are statistically different than the results of the decomposition-based method and two-level optimization. The test further reveals that the results of hybrid PSO and two-level VNS are not statistically significantly different, because the test p value is 0.0528.

Based on the statistical test and average improvement%, it can be concluded that the proposed hybrid PSO is better than the decomposition-based method and two-level optimization algorithms. The analysis also indicates that the proposed hybrid PSO is competitive with the two-level VNS for A, B, M, and P instances.

Fig. 3 reveals that the two-level optimization algorithm obtains negatively dispersed results from the *UB* for most of the instances. The algorithm, two-level VNS, achieves nearly closer results to the *UB* but the proposed hybrid PSO achieves more nearest results to the *UB*. The decomposition-based method is omitted in Fig. 3 because the results of the algorithm are far away from the *UB* for the instances.

5.2.2. *Performance evaluation for Golden instances*

Table 6 reports the result for the Golden instances. This set includes a total of 220 instances. The results by the UHGS, the two-level, two-level VNS, LMNS, and the hybrid PSO are evaluated in the table. In the comparison study (in Table 6), we omit 5 instances out of 220 instances (2 instances from instance group $n = 360$ and 3 instances from group $n = 420$), because the results produced by the proposed PSO are found to be exceptionally better than other algorithms. Although we report all the results in the appendix tables (Tables 8 to 14). The comparison shows that the LMNS improves the *UB* solution for 114 instances and the UHGS improves for 4 instances; whereas, the hybrid PSO improves a total of 136 instances. The two-level algorithm and two-level VNS algorithm obtain no improved *UB* solution of the Golden instances.

The overall average improvement for Golden instances using LMNS, UHGS, the two-level, two-level VNS is −0.18%, −0.03%, −2.40%, and −1.08% respectively. The hybrid PSO obtains an overall average improvement of 0.40%, which is better than all existing algorithms. In terms of solution quality, our nearest competitor is LMNS and UHGS. The CPU time for the LMNS and UHGS is as 9.5 s and 626.70 s respectively; whereas, the CPU time for the hybrid PSO is 9.44 s only. The hybrid PSO uses a Linux server with four 2.1 GHz processors with 16-core each and a total of 256 GB of RAM. The UHGS uses a Xeon CPU with 3.07 GHz

Table 5
Summarized results of A, B, M, and P instances.

Instances in BC			Two-level VNS			Decomposition-based method			Two-level optimization			Hybrid PSO		
Group	No. of instances	No. of Customer	No. of improved UB	Improvement %	t (s)	No. of improved UB	Improvement %	t (s)	No. of improved UB	Improvement %	t (s)	No. of improved UB	Improvement %	t (s)
A	27	31–79	0/24	−0.07%	0.05	1	−2.6%	…	1	−1. 21%	…	0	0.00%	0.06
B	23	30–77	0	−0.03%	0.04	0	−3.0%	…	0	−1.63%	…	0	0.00%	0.04
M	4	100–261	1	0.11%	3.48	0	−32.3%	…	0	−5.32%	…	1	0.09%	2.09
P	24	15–100	0	−0.01%	0.07	…	…	…	…	…	…	1	0.13%	0.27
Total	78	…	1/75	…	…	1/78	…	…	1/78	…	…	2/78	…	…
Avg	…	…	…	−0.03%	0.23	…	−5.00%	…	…	−1.7%	…	…	0.05%	0.22

Table 6
Summarized results of Golden instances.

Golden instance		UHGS			Two-level			Two-level VNS			LMNS			Hybrid PSO		
n	No of instances	No of improved UB	Improvement %	t (s)	No of improved UB	Improvement %	t (s)	No of improved UB	Improvement %	t (s)	No of improved UB	Improvement %	t (s)	No of improved UB	Improvement %	t (s)
200 201	11	0	0.00%	2866.56	…	−4.61%	10	0	−0.07%	10	8	−0.10%	9.9	8	1.12%	2.50
240	22	0	0.00%	154.93	…	−2.39%	10	0	−0.44%	10	17	−0.05%	3.7	15	1.09%	3.02
252	11	0	−0.01%	127.15	…	−0.50%	10	0	−0.53%	10	8	−0.10%	1.4	11	1.03%	3.15
255	11	0	−0.02%	135.45	…	−3.69%	10	0	−1.33%	10	8	−0.09%	2.1	11	0.92%	3.32
280	11	0	0.00%	3848.31	…	−2.94%	10	0	−0.71%	10	7	−0.05%	20.1	8	0.83%	6.07
300	11	0	0.00%	197.93	…	−1.04%	10	0	−0.93%	10	8	−0.06%	6.3	11	1.28%	4.77
320	22	0	−0.02%	202.49	…	−1.26%	10	0	−0.85%	10	13	−0.10%	4.9	20	0.71%	5.69
323	11	0	−0.08%	175.74	…	−4.94%	10	0	−0.93%	10	6	−0.26%	2.6	0	−0.88%	6.54
360	20	0	0.00%	1250.15	…	−2.87%	10	0	−1.02%	10	17/22	−0.09% −5	17.9	14	0.49%	10.53
396	11	0	−0.05%	292.26	…	−1.54%	10	0	−1.37%	10	1	−0.41%	2.4	1	−0.81%	10.29
399	11	0	−0.06%	225.26	…	−4.96%	10	0	−2.15%	10	4	−0.32%	2.8	3	−0.16%	11.69
400	11	0	−0.01%	1384.18	…	−2.56%	10	0	−1.26%	10	3	−0.15%	19.5	10	0.52%	12.24
420	8	0	0.00%	361.86	…	−2.60%	10	0	−1.11%	10	8/11	−0.12%	15.4	8	0.36%	19.90
440	11	0	−0.02%	1017.64	…	−3.67%	10	0	−1.32%	10	2	−0.21%	19.9	7	0.20%	15.82
480	22	0	−0.01%	1434.94	…	−3.42%	10	0/21	−1.49%	10	6	−0.33%	15.6	6	−0.18%	19.29
483	11	4	−0.07%	405.87	…	−4.93%	10	0	−2.23%	10	1	−0.33%	2.9	3	−0.62%	18.55
Total	215	4/220	…	…	…	…	…	0/219	…	…	114/220	….		136/215	…	…
Avg.		…	−0.03%	626.70	…	−2.40%	10	…	−1.08%	10	….	−0.18%	9.5	…	0.40%	9.44

with 16 GB of RAM running under Oracle Linux Server 6.4, two-level VNS uses CPU with Intel(R) Core(TM) i7-4790 with 3.60 GHz with 16 GB of RAM, and LMNS uses a personal computer with MS Windows 7 with an Intel(R) Core(TM) i7- 5930K CPU with 3.5 GHz with 64 GB RAM to perform their computations. In terms of speed, these computers are comparable. Therefore, it can be concluded that the hybrid PSO is superior to all algorithms stated here in terms of both solution quality and CPU time.

The statistical analysis reveals that there are significant differences in the comparison of the performance of hybrid PSO to all algorithms in the Friedman test (p $values = 0.000$). The pairwise Bonferroni test shows that the results of PSO are statistically different than the results of UHGS, two-level, and LMNS. The test between hybrid PSO and two-level VNS algorithms shows that the results of these two algorithms are not statistically different.

Based on the statistical test and average improvement%, it can be concluded that the proposed hybrid PSO is better than the UHGS, two-level, and LMNS algorithms. The analysis also indicates that the proposed hybrid PSO is competitive with the two-level VNS for the Golden instances.

As it can be noted from Fig. 4, the hybrid PSO improves the solution for most instances group. The two-level algorithm obtains relatively worse results followed by the two-level VNS algorithm. The LMNS algorithm generates comparatively better results but not as good as UHGS algorithm results. The UHGS finds the results nearly close to the *UB* for most of the instances.

5.2.3. Effect of hybridizing and improvement scheme on PSO's performance

The effect of hybridizing the proposed PSO on solution quality is presented in Table 7. The performance of the hybridization of the PSO is evaluated for the 20 major customers groups with a total of 298 instances under three settings: PSO without VNS and without improvement scheme; PSO with VNS and without improvement scheme; and the proposed PSO (i.e., PSO with VNS and with improvement scheme). The number of iterations for each setting is changed to maintain approximately the same computational time. All other parameters in the PSO framework are the same for all settings.

Table 7 shows that hybridizing the PSO with VNS and without improvement scheme improves the solution quality of the PSO without VNS and without improvement scheme by 82.54% (from −83.59% to −1.05%). The solution quality of the PSO with VNS and without improvement scheme is further improved by 1.06% (from −1.05% to 0.01%) by hybridizing the PSO with VNS and improvement scheme. Thus, the table denotes that the performance of PSO is enhanced if hybridization with VNS and with improvement scheme. These results justify the hybridization of the PSO with the VNS and with the inclusion of the improvement scheme in PSO.

In the pure improvement scheme, we implement the improvement scheme on the randomly generated initial solution for a specified number of iterations. In this scheme, the initial solution is perturbed, and then local searches of the improvement scheme are implemented. The process is repeated until the specified number of iterations is reached. Thus, the pure improvement scheme without PSO can be considered as an iterative local search (ILS) [7,47]. The result of the pure improvement scheme is found as the improvement% of 0.01% and it improves the *UB* solution for 94 instances with CPU time of 9.67 s. The total iterations for pure improvement are 14,000. The result of the proposed hybrid PSO is found as the improvement% of 0.31%, which is 0.30% (from 0.01% to 0.31%) superior to the pure improvement scheme result. The result of the improvement scheme is close to the proposed PSO algorithm. This observation brings an interesting fact about the potential of ILS. A further investigation is needed to design an efficient ILS for solving the clustered vehicle routing problem.

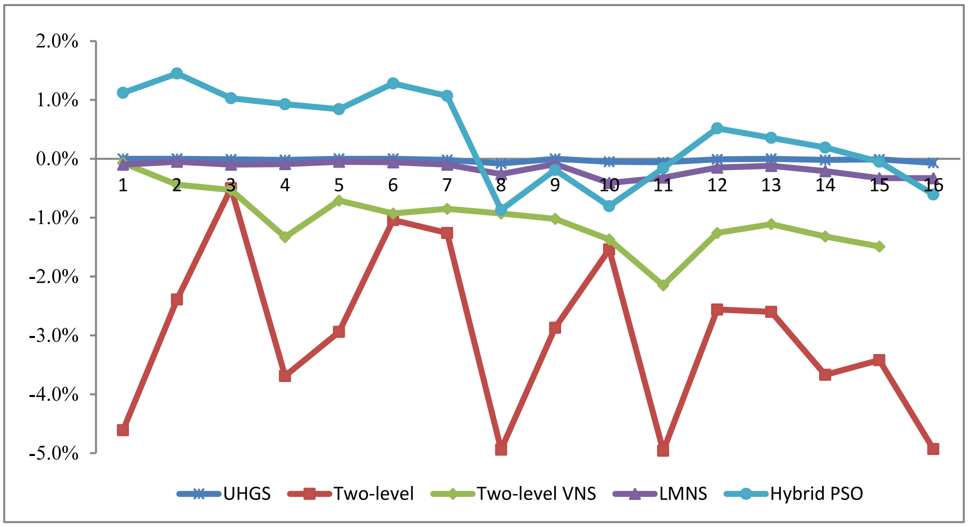

Fig. 4. Improvement% of the algorithms results for 16 groups of Golden instances.

Table 7
Effect of hybridization on solution quality.

Degree of hybridization	Number of iterations	No. of improved UB	Improvement %	t (s)
PSO without VNS and without improvement scheme	3000	0	−83.59%	11.09
PSO with VNS and without improvement scheme	350	0	−1.05%	10.44
Pure improvement scheme	14000	94	0.01%	9.67
Proposed PSO	100	138	0.31%	6.99

Table 8
Results for the instances A, B.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
A	31	11	2	522	522	0.02	0.00%
A	32	11	2	472	472	0.04	0.00%
A	32	11	2	562	562	0.02	0.00%
A	33	12	2	547	547	0.03	0.00%
A	35	12	2	588	588	0.04	0.00%
A	36	13	2	569	569	0.04	0.00%
A	36	13	2	615	615	0.04	0.00%
A	37	13	2	507	507	0.04	0.00%
A	38	13	2	610	610	0.05	0.00%
A	38	13	2	613	613	0.06	0.00%
A	43	15	2	714	714	0.08	0.00%
A	44	15	3	712	712	0.07	0.00%
A	44	15	3	664	664	0.05	0.00%
A	45	16	3	664	664	0.08	0.00%
A	47	16	3	683	683	0.08	0.00%
A	52	18	3	651	651	0.09	0.00%
A	53	18	3	724	724	0.09	0.00%
A	54	19	3	653	653	0.08	0.00%
A	59	20	3	787	787	0.09	0.00%
A	60	21	4	682	682	0.08	0.00%
A	61	21	3	778	778	0.09	0.00%
A	62	21	4	801	801	0.08	0.00%
A	62	21	3	865	865	0.08	0.00%
A	63	22	3	773	773	0.07	0.00%
A	64	22	3	725	725	0.07	0.00%
A	68	23	3	814	814	0.08	0.00%
A	79	27	4	972	972	0.09	0.00%
B	30	11	2	375	375	0.02	0.00%
B	33	12	2	416	416	0.16	0.00%
B	34	12	2	562	562	0.01	0.00%
B	37	13	2	431	431	0.01	0.00%
B	38	13	2	321	321	0.01	0.00%
B	40	14	2	476	476	0.01	0.00%
B	42	15	2	415	415	0.01	0.00%
B	43	15	3	447	447	0.01	0.00%
B	44	15	2	506	506	0.01	0.00%
B	44	15	2	391	391	0.03	0.00%
B	49	17	3	467	467	0.02	0.00%
B	49	17	3	666	666	0.02	0.00%
B	50	17	3	585	585	0.03	0.00%
B	51	18	3	427	427	0.05	0.00%
B	55	19	3	433	433	0.03	0.00%
B	56	19	3	634	634	0.05	0.00%
B	56	19	3	753	753	0.04	0.00%
B	62	21	3	685	685	0.04	0.00%
B	63	22	4	526	526	0.04	0.00%
B	65	22	3	687	687	0.05	0.00%
B	66	23	4	626	626	0.08	0.00%
B	67	23	3	588	588	0.09	0.00%
B	77	26	4	721	721	0.11	0.00%

Table 9
Results for the instances M, P.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
M	100	34	4	607	607	0.54	0.00%
M	120	41	3	691	693	0.67	−0.29%
M	150	51	4	804	804	2.52	0.00%
M	199	67	6	914	908	4.61	+0.66%
P	100	51	5	679	669	1.95	+3.18%
P	15	6	4	253	253	0.01	0.00%
P	18	10	2	186	186	0.01	0.00%
P	19	7	1	200	200	0.01	0.00%
P	20	7	1	190	190	0.01	0.00%
P	21	8	1	202	202	0.01	0.00%
P	21	8	4	365	365	0.03	0.00%
P	22	8	3	279	279	0.02	0.00%
P	39	14	2	396	396	0.06	0.00%
P	44	15	2	440	440	0.09	0.00%
P	49	17	4	491	491	0.10	0.00%
P	49	17	3	447	447	0.10	0.00%
P	49	17	3	460	460	0.10	0.00%
P	50	17	4	537	537	0.13	0.00%
P	54	19	4	500	500	0.13	0.00%
P	54	19	6	595	471	0.23	0.00%
P	54	19	3	462	462	0.26	0.00%
P	54	19	3	471	595	0.17	0.00%
P	59	20	4	552	552	0.35	0.00%
P	59	20	5	611	611	0.21	0.00%
P	64	22	4	619	619	0.40	0.00%
P	69	24	4	643	643	0.47	0.00%
P	75	26	2	581	581	0.84	0.00%
P	75	26	2	581	581	0.84	0.00%

6. Conclusion

The combinatorial optimization problem, the CluVRP, is considered in this paper. In the CluVRP, customers are partitioned into predefined clusters. The same vehicle is assigned to serve all customers consecutively under a cluster before it moves to another cluster or returns to the depot. All customers and clusters must be served only once. The objective of the problem is to find the optimal distribution costs for the logistic network serving all customers by using the available vehicles. In this paper, a hybrid PSO algorithm is proposed to solve the CluVRP. With the complementary nature of both algorithms, the hybrid PSO combines the local optimal improvement capabilities of VNS with the swarm based diversification abilities of the PSO. The

Table 10
Results for the Golden instances 1–4.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
Golden 1	240	17	4	4831	4751	3.66	1.66%
Golden 1	240	18	4	4847	4757	2.42	1.86%
Golden 1	240	19	4	4872	4789	2.45	1.70%
Golden 1	240	21	4	4889	4790	2.57	2.02%
Golden 1	240	22	4	4908	4826	2.58	1.67%
Golden 1	240	25	4	4899	4818	2.61	1.65%
Golden 1	240	27	4	4934	4862	2.60	1.46%
Golden 1	240	31	4	5050	4953	2.68	1.92%
Golden 1	240	35	4	5102	5047	2.98	1.08%
Golden 1	240	41	4	5097	5058	3.64	0.77%
Golden 1	240	49	3	5000	4953	4.38	0.94%
Golden 2	320	22	4	7716	7622	6.10	1.22%
Golden 2	320	23	4	7693	7578	6.04	1.49%
Golden 2	320	25	4	7668	7571	6.14	1.26%
Golden 2	320	27	4	7638	7527	5.27	1.45%
Golden 2	320	30	4	7617	7552	4.55	0.85%
Golden 2	320	33	4	7640	7548	4.12	1.20%
Golden 2	320	36	4	7643	7550	4.71	1.22%
Golden 2	320	41	4	7738	7644	4.80	1.21%
Golden 2	320	46	4	7861	7795	5.59	0.84%
Golden 2	320	54	4	7920	7830	7.27	1.14%
Golden 2	320	65	4	7892	7841	10.32	0.65%
Golden 3	400	27	4	10540	10489	17.15	0.48%
Golden 3	400	29	4	10504	10393	11.23	1.06%
Golden 3	400	31	4	10486	10395	8.33	0.87%
Golden 3	400	34	4	10465	10408	8.56	0.54%
Golden 3	400	37	4	10482	10415	8.50	0.64%
Golden 3	400	41	4	10501	10426	10.03	0.71%
Golden 3	400	45	4	10485	10405	9.66	0.76%
Golden 3	400	51	4	10583	10538	10.70	0.43%
Golden 3	400	58	4	10776	10751	12.38	0.23%
Golden 3	400	67	4	10797	10785	15.36	0.11%
Golden 3	400	81	4	10614	10627	22.75	−0.12%
Golden 4	480	33	4	13598	13567	19.24	0.23%
Golden 4	480	35	4	13643	13635	19.17	0.06%
Golden 4	480	37	4	13520	13498	16.29	0.16%
Golden 4	480	41	4	13460	13473	16.55	−0.10%
Golden 4	480	44	4	13568	13540	16.65	0.21%
Golden 4	480	49	4	13758	13772	17.88	−0.10%
Golden 4	480	54	4	13760	13767	19.11	−0.05%
Golden 4	480	61	4	13791	13796	20.86	−0.04%
Golden 4	480	69	4	13966	13975	20.77	−0.06%
Golden 4	480	81	4	13975	14001	27.50	−0.19%
Golden 4	480	97	4	13775	13833	36.26	−0.42%

Table 11
Results for the Golden instances 5–8.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
Golden 5	200	14	4	7622	7462	3.08	2.10%
Golden 5	200	15	3	7424	7424	2.94	0.00%
Golden 5	200	16	3	7491	7491	2.92	0.00%
Golden 5	200	17	3	7434	7434	2.83	0.00%
Golden 5	200	19	4	7576	7484	2.11	1.21%
Golden 5	200	21	4	7596	7489	1.98	1.41%
Golden 5	200	23	4	7643	7532	2.02	1.45%
Golden 5	200	26	4	7560	7436	2.15	1.64%
Golden 5	200	29	4	7410	7299	2.28	1.50%
Golden 5	200	34	4	7429	7321	2.52	1.45%
Golden 5	200	41	4	7241	7130	2.69	1.53%
Golden 6	280	19	3	8624	8624	8.87	0.00%
Golden 6	280	21	3	8628	8633	7.97	−0.06%
Golden 6	280	22	3	8646	8655	6.14	−0.10%
Golden 6	280	24	4	8853	8728	5.46	1.41%
Golden 6	280	26	4	8910	8777	5.57	1.49%
Golden 6	280	29	4	8936	8846	4.51	1.01%
Golden 6	280	32	4	8891	8799	4.37	1.03%
Golden 6	280	36	4	8969	8862	4.79	1.19%
Golden 6	280	41	4	9028	8920	5.30	1.20%
Golden 6	280	47	4	8923	8823	6.08	1.12%
Golden 6	280	57	4	9028	8948	7.77	0.89%
Golden 7	360	25	3	9904	9978	12.34	−0.75%
Golden 7	360	26	3	9888	9946	10.85	−0.59%
Golden 7	360	28	3	9917	9963	10.67	−0.46%
Golden 7	360	31	4	10021	9989	10.00	0.32%
Golden 7	360	33	4	10029	9937	9.42	0.92%
Golden 7	360	37	4	10131	10034	9.93	0.96%
Golden 7	360	41	4	10052	9975	10.57	0.77%
Golden 7	360	46	4	10080	10010	9.70	0.69%
Golden 7	360	52	4	10095	10010	10.15	0.84%
Golden 7	360	61	4	10096	10061	12.83	0.35%
Golden 7	360	73	4	10014	9985	17.67	0.29%
Golden 8	440	30	4	10866	10797	13.57	0.64%
Golden 8	440	32	4	10831	10744	13.48	0.80%
Golden 8	440	34	4	10847	10787	13.54	0.55%
Golden 8	440	37	4	10859	10792	13.09	0.62%
Golden 8	440	41	4	10934	10898	13.50	0.33%
Golden 8	440	45	4	10960	10947	13.65	0.12%
Golden 8	440	49	4	11042	11045	11.84	−0.03%
Golden 8	440	56	4	11194	11224	13.35	−0.27%
Golden 8	440	63	4	11252	11279	15.74	−0.24%
Golden 8	440	74	4	11321	11314	21.45	0.06%
Golden 8	440	89	4	11209	11256	30.78	−0.42%

Table 12
Results for the Golden instances 9–12.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
Golden 9	255	18	4	300	296	3.17	1.33%
Golden 9	255	19	4	299	295	3.05	1.34%
Golden 9	255	20	4	296	293	2.98	1.01%
Golden 9	255	22	4	290	289	2.91	0.34%
Golden 9	255	24	4	290	289	2.85	0.34%
Golden 9	255	26	4	288	285	2.76	1.04%
Golden 9	255	29	4	292	291	2.78	0.34%
Golden 9	255	32	4	297	293	3.03	1.35%
Golden 9	255	37	4	294	290	3.43	1.36%
Golden 9	255	43	4	295	292	4.09	1.02%
Golden 9	255	52	4	296	294	5.52	0.68%
Golden 10	323	22	4	367	373	5.60	−1.63%
Golden 10	323	24	4	361	365	5.28	−1.11%
Golden 10	323	25	4	359	361	5.20	−0.56%
Golden 10	323	27	4	361	365	5.27	−1.11%
Golden 10	323	30	4	367	370	5.43	−0.82%
Golden 10	323	33	4	373	379	5.40	−1.61%
Golden 10	323	36	4	385	389	5.61	−1.04%
Golden 10	323	41	4	400	402	6.18	−0.50%
Golden 10	323	47	4	398	399	7.10	−0.25%
Golden 10	323	54	4	393	395	8.77	−0.51%
Golden 10	323	65	4	387	389	12.11	−0.52%
Golden 11	399	27	5	457	452	8.42	1.09%
Golden 11	399	29	5	455	456	8.40	−0.22%
Golden 11	399	31	5	455	457	8.44	−0.44%
Golden 11	399	34	5	455	456	8.50	−0.22%
Golden 11	399	37	5	459	461	8.70	−0.44%
Golden 11	399	40	5	461	462	9.07	−0.22%
Golden 11	399	45	5	462	461	9.76	0.22%
Golden 11	399	50	5	458	456	10.98	0.44%
Golden 11	399	58	5	456	458	13.55	−0.44%
Golden 11	399	67	5	454	458	17.44	−0.88%
Golden 11	399	80	5	451	454	25.36	−0.67%
Golden 12	483	33	5	535	541	11.56	−1.12%
Golden 12	483	35	5	537	542	11.55	−0.93%
Golden 12	483	38	5	535	542	11.64	−1.31%
Golden 12	483	41	5	537	541	11.59	−0.74%
Golden 12	483	44	5	535	545	11.93	−1.87%
Golden 12	483	49	5	533	540	13.69	−1.31%
Golden 12	483	54	5	535	545	15.33	−1.87%
Golden 12	483	61	5	538	542	18.42	−0.74%
Golden 12	483	70	5	546	539	23.37	1.28%
Golden 12	483	81	5	546	545	30.51	0.18%
Golden 12	483	97	5	560	551	44.42	1.61%

Table 13
Results for the Golden instances 13–16.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
Golden 13	252	17	4	552	549	2.85	0.54%
Golden 13	252	19	4	549	544	2.67	0.91%
Golden 13	252	20	4	548	540	2.70	1.46%
Golden 13	252	22	4	548	540	2.65	1.46%
Golden 13	252	23	4	548	540	2.65	1.46%
Golden 13	252	26	4	542	535	2.68	1.29%
Golden 13	252	29	4	540	534	2.77	1.11%
Golden 13	252	32	4	543	538	3.01	0.92%
Golden 13	252	37	4	545	543	3.41	0.37%
Golden 13	252	43	4	553	549	4.08	0.72%
Golden 13	252	51	4	560	554	5.17	1.07%
Golden 14	320	22	4	692	690	4.87	0.29%
Golden 14	320	23	4	688	685	4.64	0.44%
Golden 14	320	25	4	678	676	4.41	0.29%
Golden 14	320	27	4	676	676	4.38	0.00%
Golden 14	320	30	4	678	680	4.42	−0.29%
Golden 14	320	33	4	682	681	4.45	0.15%
Golden 14	320	36	4	687	685	4.58	0.29%
Golden 14	320	41	4	690	688	5.04	0.29%
Golden 14	320	46	4	694	691	5.84	0.43%
Golden 14	320	54	4	699	697	7.46	0.29%
Golden 14	320	65	4	703	697	10.13	0.85%
Golden 15	396	27	4	842	854	6.86	−1.43%
Golden 15	396	29	4	843	852	6.89	−1.07%
Golden 15	396	31	4	837	851	6.69	−1.67%
Golden 15	396	34	4	838	852	6.85	−1.67%
Golden 15	396	37	4	845	857	6.98	−1.42%
Golden 15	396	40	4	849	856	7.40	−0.82%
Golden 15	396	45	5	853	852	7.36	0.12%
Golden 15	396	50	5	851	853	10.27	−0.24%
Golden 15	396	57	5	850	853	12.35	−0.35%
Golden 15	396	67	5	855	857	16.69	−0.23%
Golden 15	396	80	5	857	858	24.89	−0.12%
Golden 16	480	33	5	1030	1029	11.45	0.10%
Golden 16	480	35	5	1028	1026	10.97	0.19%
Golden 16	480	37	5	1028	1028	10.83	−0.00%
Golden 16	480	41	5	1032	1035	11.50	−0.29%
Golden 16	480	44	5	1028	1031	11.93	−0.29%
Golden 16	480	49	5	1031	1034	12.51	−0.29%
Golden 16	480	54	5	1022	1027	14.74	−0.49%
Golden 16	480	61	5	1013	1022	17.35	−0.89%
Golden 16	480	69	5	1012	1017	21.60	−0.49%
Golden 16	480	81	5	1018	1024	30.20	−0.59%
Golden 16	480	97	5	1018	1025	40.92	−0.69%

Table 14
Results for the Golden instances 17–20.

Instance				BC	Hybrid PSO		
Group	n	c	m	UB	Solution	CPU t (s)	Improvement %
Golden 17	240	17	3	418	420	3.17	−0.48%
Golden 17	240	18	3	419	422	3.07	−0.72%
Golden 17	240	19	3	422	422	2.94	−0.00%
Golden 17	240	21	3	425	426	2.89	−0.24%
Golden 17	240	22	3	424	426	2.92	−0.47%
Golden 17	240	25	3	418	419	2.68	−0.24%
Golden 17	240	27	3	414	415	2.67	−0.24%
Golden 17	240	31	4	421	411	2.77	2.38%
Golden 17	240	35	4	417	406	2.97	2.64%
Golden 17	240	41	4	412	403	3.61	2.18%
Golden 17	240	49	4	414	404	4.13	2.42%
Golden 18	300	21	4	592	587	5.01	0.84%
Golden 18	300	22	4	594	590	4.98	0.67%
Golden 18	300	24	4	592	587	4.05	0.84%
Golden 18	300	26	4	590	580	4.17	1.69%
Golden 18	300	28	4	577	569	4.04	1.39%
Golden 18	300	31	4	578	572	3.63	1.04%
Golden 18	300	34	4	582	574	3.69	1.37%
Golden 18	300	38	4	586	580	4.34	1.02%
Golden 18	300	43	4	594	584	4.70	1.68%
Golden 18	300	51	4	601	591	5.81	1.66%
Golden 18	300	61	4	599	588	8.09	1.84%
Golden 19	360	25	10	925	807	54.79	12.76%
Golden 19	360	26	10	924	807	52.66	12.60%
Golden 19	360	28	4	808	813	9.26	−0.62%
Golden 19	360	31	4	811	815	7.84	−0.49%
Golden 19	360	33	4	797	802	7.07	−0.63%
Golden 19	360	37	5	799	790	7.00	1.13%
Golden 19	360	41	5	789	776	7.30	1.65%
Golden 19	360	46	5	788	775	8.27	1.65%
Golden 19	360	52	5	800	788	9.68	1.50%
Golden 19	360	61	5	807	798	12.27	1.12%
Golden 19	360	73	5	810	801	17.74	1.11%
Golden 20	420	29	11	1220	1081	99.2	11.39%
Golden 20	420	31	12	1232	1072	84.19	12.99%
Golden 20	420	33	12	1208	1060	78.68	12.25%
Golden 20	420	36	5	1059	1056	10.93	0.28%
Golden 20	420	39	5	1052	1047	10.14	0.48%
Golden 20	420	43	5	1052	1048	9.58	0.38%
Golden 20	420	47	5	1053	1052	17.35	0.09%
Golden 20	420	53	5	1058	1053	18.52	0.47%
Golden 20	420	61	5	1058	1055	23.47	0.28%
Golden 20	420	71	5	1049	1054	30.54	0.47%
Golden 20	420	85	5	1049	1045	38.64	0.38%

algorithm is tested on the benchmark instances found in CluVRP literature. The major contributions of the study include designing a new hybrid PSO metaheuristic algorithm to solve the CluVRP and finding the new best-known solutions for a total of 138 instances out of 293 benchmark instances with an average CPU time of 6.99 s. It is also contributed in this study by adding new features in the PSO algorithm such as the use of two types of particles and improvement scheme for the personal best solution. In the improvement scheme, the personal best solutions of the swarm are further improved by adopting the perturbation and VNS method. Hence, the proposed algorithm has great potential for solving instances of other variants of VRP. With the capability of a quality solution on relatively acceptable CPU time, the algorithm has the perspective to use in many practical scenarios such as distribution logistics with CO_2 emission cap leading to a penalty, the problem of perishable items, and transportation problems in military operations, etc. Like all research works, this work also has some limitations and future research directions. Many attributes of VRPs such as time windows, carbon emissions, backhaul, and multi-depot can be added with CluVRP to capture real-world scenarios. Although the proposed algorithm is designed to solve CluVRP solely, it can be easily extended to solve other variants of VRP. Future research works can also explore the possibility of combining PSO with other metaheuristics such as genetic algorithm, tabu search, simulated annealing etc.

CRediT authorship contribution statement

Md. Anisul Islam: Analyzed the problem, Designed the methodology, Analyzed and interpreted the data, Conceptualized the solution techniques, Performed the experiments, Wrote the paper. **Yuvraj Gajpal:** Designed the methodology, Analyzed and interpreted the data, Conceptualized the solution techniques, Reviewed and edited the paper. **Tarek Y. ElMekkawy:** Designed the methodology, Analyzed and interpreted the data, Conceptualized the solution techniques, Reviewed and edited the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors acknowledge the funding received from the Manitoba Government through the Department of Education and Advanced Learning, and the University of Manitoba to support this research. The authors thank the anonymous referees for their valuable comments.

Appendix. The detailed computational results

See Tables 8–14.

References

[1] C. Expósito-Izquierdo, A. Rossi, M. Sevaux, A two-level solution approach to solve the clustered capacitated vehicle routing problem, Comput. Ind. Eng. 91 (2016) 274–289.

[2] P. Toth, D. Vigo, Models, relaxations and exact approaches for the capacitated vehicle routing problem, Discrete Appl. Math. 123 (2002) 487–512.

[3] T. Barthélemy, A. Rossi, M. Sevaux, K. Sörensen, Metaheuristic approach for the clustered VRP, in: EU/ME 2010—10th Anniversary of the Metaheuristic Community, Lorient, France, 2010.

[4] P.C. Pop, I. Kara, A.H. Marc, New mathematical models of the generalized vehicle routing problem and extensions, Appl. Math. Model. 36 (2012) 97–107.

[5] M. Battarra, G. Erdoğan, D. Vigo, Exact algorithms for the clustered vehicle routing problem, Oper. Res. 62 (2014) 58–71.

[6] A.H. Marc, L. Fuksz, P.C. Pop, D. Danciulescu, A novel hybrid algorithm for solving the clustered vehicle routing problem, in: E. Onieva, I. Santos, E. Osaba, E. Quintián (Eds.), Hybrid Artificial Intelligent Systems, Springer, 2015, pp. 679–689.

[7] T. Vidal, M. Battarra, A. Subramanian, G. Erdogan, Hybrid metaheuristics for the clustered vehicle routing problem, Comput. Oper. Res. 48 (2015) 87–99.

[8] C. Defryn, K. Sörensen, A fast two-level variable neighborhood search for the clustered vehicle routing problem, Comput. Oper. Res. 83 (2017) 78–94.

[9] P.C. Pop, L. Fuksz, A.H. Marc, C. Sabo, A novel two-level optimization approach for clustered vehicle routing problem, Comput. Ind. Eng. 115 (2018) 304–318.

[10] T. Hintsch, S. Irnich, Large multiple neighborhood search for the clustered vehicle routing problem, European J. Oper. Res. 270 (2018) 118–131.

[11] T. Hintsch, Large multiple neighborhood search for the soft-clustered vehicle-routing problem, Comput. Oper. Res. 129 (2021) 105132.

[12] M. Sevaux, K. Sörensen, Hamiltonian paths in large clustered routing problems, in: Proceedings of the EU/Meeting 2008 workshop on Metaheuristics for Logistics and Vehicle Routing, EU/ME, Vol. 8, 2008, pp. 411-417.

[13] V. Schmid, K.F. Doerner, G. Laporte, Rich routing problems arising in supply chain management, European J. Oper. Res. 224 (2013) 435–448.

[14] A. Subramanian, Heuristic, Exact and Hybrid Approaches for Vehicle Routing Problems (Ph.D. thesis), University Federal Fluminense, Niteroi (Brazil, 2012.

[15] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings Of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39-43.

[16] T.J. Ai, V. Kachitvichyanukul, A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery, Comput. Oper. Res. 36 (2009) 1693–1702.

[17] Y. Marinakis, G.R. Iordanidou, M. Marinaki, Particle swarm optimization for the vehicle routing problem with stochastic demands, Appl. Soft Comput. 13 (2013) 1693–1704.

[18] N. Norouzi, M. Sadegh-Amalnick, M. Alinaghiyan, Evaluating of the particle swarm optimization in a periodic vehicle routing problem, Measurement 62 (2015) 162–169.

[19] D.C. Hop, N. Van Hop, T.T.M. Anh, Adaptive particle swarm optimization for integrated quay crane and yard truck scheduling problem, Comput. Ind. Eng. 153 (2021) 107075.
[20] I.H. Dridi, E.B. Alaïa, P. Borne, H, Bouchriha, Optimisation of the multi-depots pick-up and delivery problems with time windows and multi-vehicles using PSO algorithm, Int. J. Prod. Res. 58 (14) (2020) 1–14.
[21] Q. Nie, S. Liu, Q. Qian, Z. Tan, H. Wang, Optimization of the Sino-Europe transport networks under uncertain demand, Asia-Pac. J. Oper. Res. (2021).
[22] S.N. Sahu, Y. Gajpal, S. Debbarma, Two-agent-based single-machine scheduling with switchover time to minimize total weighted completion time and makespan objectives, Ann. Oper. Res. 269 (2018) 623–640.
[23] J. Li, Y. Gajpal, A.K. Bhardwaj, H. Chen, Y. Liu, Two-agent single machine order acceptance scheduling problem to maximize net revenue, Complexity (2021) http://dx.doi.org/10.1155/2021/6627081.
[24] N. Mladenovic, P. Hansen, Variable neighborhood search, Comput. Oper. Res. 24 (11) (1997) 1097–1100.
[25] M.A. Islam, Y. Gajpal, Optimization of conventional and green vehicles composition under carbon emission cap, Sustainability 13 (12) (2021) 6940, http://dx.doi.org/10.3390/su13126940.
[26] P. Hansen, N. Mladenovic, Variable neighborhood search, in: F. Glover, G. Kochenberge (Eds.), Handbook of Metaheuristics, Boston, Kluwer, 2003, pp. 145–184.
[27] Y. Marinakis, M. Marinaki, G. Dounias, A hybrid particle swarm optimization algorithm for the vehicle routing problem, Eng. Appl. Artif. Intell. 23 (2010) 463–472.
[28] F.P. Goksal, I. Karaoglan, F. Altiparmak, A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery, Comput. Ind. Eng. 65 (2013) 39–53.
[29] Marinakis Y., Marinaki M., Migdalas A., A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows, Info. Sci. 8 (10) (2019) 2583–2589.
[30] S. Zou, Jin. Li, X. Li, A hybrid particle swarm optimization algorithm for multi-objective pickup and delivery problem with time windows, J. Comput. 8 (10) (2013) 2583–2589.
[31] S. Zhang, M. Chen, W. Zhang, A novel location-routing problem in electric vehicle transportation with stochastic demands, J. Cleaner Prod. 221 (2019) 567–581.
[32] P. Hansen, N. Mladenovic, Variable neighborhood search: principles and applications, European J. Oper. Res. 130 (3) (2001) 449–467.
[33] B.F. Moghaddam, R. Ruiz, S.J. Sadjadi, Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm, Comput. Ind. Eng. 62 (2012) 306–317.
[34] G. Moslehi, M. Mahnam, A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search, Int. J. Prod. Econ. 129 (1) (2011) 14–22.
[35] H. Liu, A. Abraham, O. Choi, S.H. Moon, Variable neighborhood particle swarm optimization for multi-objective flexible job-shop scheduling problems, Lecture Notes Comput. Sci. 4247 (2006) 197–204.
[36] P. Pongchairerks, V. Kachitvichyanuku, A comparison between algorithms VNS with PSO and VNS without PSO for job-shop scheduling problems, Int. J. Comput. Sci. 1 (2) (2007) 179–191.
[37] A.F. Ali, A.E. Hassanien, V. Snasel, M.F. Tolba, A new hybrid particle swarm optimization with variable neighborhood search for solving unconstrained global optimization problems, in: P. Kromer, et al. (Eds.), Proceedings of the Fifth Intern. Conf. on Innov. in 151 Bio-Inspired Comput. and Appl. IBICA 2014, in: Advances in Intelligent Systems and Computing, 303, Springer International Publishing Switzerland, 2014, http://dx.doi.org/10.1007/978-3-319-08156-4_16c.
[38] L. Zhang, J. Wu, A pso-based hybrid metaheuristic for permutation flow shop scheduling problems, Sci. World J. (2014) 1–8.
[39] B.F. Gumaida, J. Luo, A hybrid particle swarm optimization with a variable neighborhood search for the localization enhancement in wireless sensor networks, Appl. Intell. 49 (2019) 3539–3557.
[40] Y. Marinakis, A. Migdalas, A. Sifaleras, A hybrid particle swarm optimization –variable neighborhood search algorithm for constrained shortest path problems, European J. Oper. Res. 261 (2017) 819–834.
[41] L. Cai, W. Lv, L. Xiao, Z. Xu, Total carbon emissions minimization in connected and automated vehicle routing problem with speed variables, Expert Syst. Appl. 165 (2021) 113910.
[42] M. Ranjbar, R.G. Saber, A variable neighborhood search algorithm for transshipment scheduling of multi products at a single station, Appl. Soft Comput. 98 (2021) 106736.
[43] M.A. Islam, Y. Gajpal, T.Y. ElMekkawy, Mixed fleet based green clustered logistics problem under carbon emission cap, Sustainable Cities Soc. 72 (2021) 103074.
[44] A. Subramanian, L.M.A. Drummond, C. Bentes, L.S. Ochi, R. Farias, A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery, Comput. Oper. Res. 37 (11) (2010) 1899–1911.
[45] T. Bektas, G. Erdogan, S. Ropke, Formulations and branch-and-cut algorithms for the generalized vehicle routing problem, Transp. Sci. 45 (2011) 299–316.
[46] A.E. Ezugwu, A.O. Adewumi, M.E. Frîncu, Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem, Expert Syst. Appl. 77 (2017) 189–210.
[47] G. Macrina, L.D.P. Pugliese, F. Guerriero, G. Laporte, The green mixed fleet vehicle routing with partial battery recharging and time windows, Comput. Oper. Res. 101 (2019) 183–199.

Md. Anisul Islam is a Ph.D. student of the Department of Mechanical Engineering at the University of Manitoba, Winnipeg, Canada. He received his M.Sc. in Industrial Engineering. His research interests are in green vehicle routing problems, optimization techniques for transportation, distribution, and logistic problems.

Yuvraj Gajpal is an Assistant Professor of Supply Chain Management at Asper School of Business, University of Manitoba Winnipeg, Canada. He holds a Ph.D. in Management Science from DeGroote School of Business at McMaster University Hamilton, Canada and Master in Industrial Management from Indian Institute of Technology (IIT) Madras, India. His research interests mainly lie on the application of heuristics and meta-heuristics on transportation and logistics management. He has published papers in leading research journals such as Computers and Operations Research, European Journal of Operations Research, International Journal of Production Economics, Annals of Operations Research, Reliability Engineering and Systems Safety, Construction Management and Economics and Journal of the Operational Research Society. He is a reviewer of many international journals such as Computers and Operations Research, European Journal of Operations Research, Computers and Industrial Engineering, Journal of Heuristics and Transportation Research Part E.

Dr. T. Y. ElMekkawy is a Professor at Qatar University. One of his research focuses is related to scheduling and performance optimization of manufacturing systems. His research is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). He has obtained research funding from Canadian Industry and healthcare organizations such as Motor Coach (Manitoba), Winnipeg Regional Health Authority (WRHA), Manitoba Patient Access Network (MPAN), Manitoba Health, and CancerCare Manitoba. His publications appeared in high impact journals such as Journal of Renewable Energy, International Journal for Energy Research, ASME Transactions, AMJ, IJPR, IJPE, IJCIM, IJAMT, IJOR, and EJIE.