RAIRO-Oper. Res. 49 (2015) 689–716
DOI: 10.1051/ro/2014064

RAIRO Operations Research
www.rairo-ro.org

VEHICLE ROUTING PROBLEM WITH LIMITED REFUELING HALTS USING PARTICLE SWARM OPTIMIZATION WITH GREEDY MUTATION OPERATOR

Ganesan Poonthalir[1], Rethnaswamy Nadarajan[1] and Shanmugam Geetha[2]

Abstract. Route planning and goods distribution are a major component of any logistics. Vehicle Routing Problem is a class of problems addressing the issues of logistics. Vehicle Routing Problem with Limited Refueling Halts is introduced in this paper. The objective is to plan a route with an emphasis on the time and cost involved in refueling vehicles. The method is tailored to find optimal routes with minimal halts at the refueling stations. The problem is modeled as a bi objective optimization problem and is solved using particle swarm optimization. A new mutation operator called greedy mutation operator is introduced. Experiments are conducted with available data sets and MATLABR2011a is used for implementation.

Keywords. Logistics, Vehicle Routing Problem, particle swarm optimization.

Mathematics Subject Classification. 90B06, 90C27, 90C59.

1. Introduction

Logistics plays a vital role for any distribution management. Many variants that deal with the real life problems in logistics are addressed under the topic Vehicle Routing Problem (VRP). It is a NP-hard Combinatorial Optimization Problem.

Received August 27, 2014. Accepted December 10, 2014.

[1] Department of Applied Mathematics and Computational Sciences, PSG College of Technology, 641004 Coimbatore, India. poonthalirk@gmail.com; nadarajan_psg@yahoo.co.in

[2] Department of Computer Applications, PSG College of Technology, 641004 Coimbatore, India. geet_shan@yahoo.com

Article published by EDP Sciences

Hence many heuristics are developed to solve the problem. It is an important area of research in Operations Research Community. Of the many variants Capacitated VRP (CVRP) and Vehicle Routing problem with time windows (VRPTW) are the major topics of discussion. A general VRP aims to find low cost tour for geographically distributed customers by vehicles. The route begins and ends with the central distribution center serving each customer once and returning to the depot. As VRP provides real time solution for many problems, many researchers have contributed different solutions for solving the problem.

Different variants on VRP like Stochastic VRP, Multi Depot Vehicle Routing Problem, VRP with Pickup and Delivery *etc.* are developed using additional constraints along with VRP. A detailed review on classical and advanced methods on VRP can be found in the works of Golden *et al.* [12], Laporte *et al.* [18] and Toth *et al.* [28]. Classes of exact and heuristic methods are developed to handle the problems on VRP. Many heuristics are developed to solve the problem and can be observed in the works of Gendreau *et al.* [11], Pisinger *et al.* [26], *etc.*

Along with the traditional variants, a category of VRP evolves that deal with a variety of real time issues with the introduction of new objectives and constraints. These include problems that arise with environmental issues along with routing, harmful gas emissions and their hazard, hazardous materials transportation, pollution VRP to name a few. Sbihi *et al.* [27] demonstrated the impact of environmental issues with logistics in their work. They also emphasized special concern for transporting hazardous materials. Xiao *et al.* [31] in their work stated that, fuel cost accounts for significant increase in the overall total cost of the tour. They developed a model that tries to reduce the fuel consumption by vehicles. Pollution VRP aims at finding a low cost tour with minimal CO_2 emissions. Bektas *et al.* [4] addressed the pollution routing problems and the impact of harmful gases on the environment. Kuo *et al.* [17] described the fuel consumption as an important criterion for vehicle routing. They proposed a model for calculating fuel consumption for time dependent VRP and solved using simulated annealing.

Since most organization is working towards reducing the consumption of fuel or looking for alternate fuel usage, designing a VRP with fuel usage as a constraint has become an important consideration for designing Vehicle Routing Problem with Limited Refueling Halts (VRPLHS). It is a subset of VRP and is also NP-hard.

VRPLHS has a set of geographically distributed customers served by a limited fleet of vehicles stationed at a depot. Each vehicle has a limited fuel capacity. This limited fuel level is considered for redirecting vehicle to nearby Refueling Stations (RF), when a vehicle is engaged to make long tours. Along with the route cost, this additional cost is also accumulated. This work concentrates in minimizing the overall cost of tour and number of halts made to RF for refueling. This will aid organization to plan the tour, which minimizes the cost of tour and the number of refueling halts.

There are few papers that address the issue of fuel usage as a constraint. Some papers are found towards fuel consumption and its impact on the environment. Green VRP (GVRP), VRP with battery capacity, flow refueling location

based model for VRP are some of the VRP variants on fuel usage constraints. Wang *et al.* [30] proposed Vehicle Scheduling Problem with route and fueling time constraints for routing electric buses. It concentrates in finding an economic solution for locating a refueling station. Aviral *et al.* [10] in their work addressed the usage of alternative fuel and optimum erection of alternative fuelling stations and modeled the problem as flow interception facility location problem. Artmeier *et al.* [2], proposed an extension of shortest route problem to find an energy optimal routing for electric cars. A multi depot unmanned vehicle with fuel constraints was proposed by Levy *et al.* [19] The unmanned vehicles have heterogeneous fuel capacity and the objective is to route the vehicle, so that each target is visited at least once and refueled in the fueling stations. Erdogan *et al.* [10] formulated a Green VRP (GVRP) with alternative fuel powered vehicle, with the impact of reducing the pollution due to combustion. They concentrated in routing vehicles with alternate fuel to reduce environment pollution employed with Alternate Fuel Stations (AFS) to route the vehicle. They used Modified Clarke and Wright Savings Algorithm (MCWS) and Density Based Clustering Algorithm (DBCA) to find the total cost of the route with trips to AFS.

In contrast to the model GVRP proposed by Erdogan *et al.* [10] where the main objective is to find low cost tour along with trips to AFS, the proposed VRPLHS aims to calculate low cost tour with trips to refueling stations and also strives to reduce the total number of halts made at the refueling station. G-VRP does not specify any constraint on the upper bound of the number of refueling halts. In VRPLHS, it is included as one of the objective functions. The objectives of VRPLHS are, to minimize cost of tour and minimize the number of halts made at *RF*. Hence VRPLHS is modeled as a Bi-objective Optimization Problem (BOP).

The problem of single objective optimization with all constraints is a direct method that yields a single optimal solution. Since BOP is also a Multi objective Problem (MOP), there is not a single optimal solution that optimizes the objective function. A compromising solution that satisfies both objectives is said to be a Pareto optimal solution. Pareto optimality (efficient, non-dominated) is defined as, to arrive at a solution with an objective that cannot be improved by worsening the other objective. It is not possible for a single solution that optimizes both the objectives hence a trade-off is established to arrive at an optimal solution. Recently, emphasis is built on solving MOP with evolutionary methods. Traditional methods like genetic algorithms are already used for solving MOP.

In this work, the problem is solved using Particle Swarm optimization (PSO). PSO is a stochastic search algorithm and is well suited for MOP as stated by Kennedy and Eberhart [16]. The advantages of PSO are, it has better exploration capabilities, it can retain best solutions achieved so far and it has too few parameters to tune for better performance. Moreover, all particles will interact with each other exhibiting the co-operative behavior that enhances the sharing among the particles, eventually leading to the solution improvement. Also the exploratory behavior of PSO is improved, by introducing a new mutation operator called Greedy mutation operator (GMO).

When deciding the evolutionary algorithm for MOP two goals are to be addressed: to guide the search towards Pareto efficient set and to keep track of the set of non-dominated solutions. The first goal can be achieved using velocity that guides the swarm towards better positions and to achieve the second goal an archive of best solutions found is maintained using PSO.

The contribution of this paper is threefold that is, it aids those organizations with limited vehicle fleet to plan a route that limits the number of halts made for refueling along with minimum cost of tour. Second, modeling the problem as a BOP and the use of PSO as an efficient method of solving the problem and third, the introduction of Greedy Mutation Operator (GMO) that makes PSO escape from local minima.

2. Bi Objective optimization (BOP)

Multi objective optimization problems are defined with more than one objective function. As specified in Caramia *et al.* [5], it is defined mathematically as,

$$\min(f_1(x), f_2(x), \ldots, f_n(x)),$$

$$x \in S, \tag{2.1}$$

where $n > 1$, S is the set of constraints defined as $S = x \in R^m / h(x) = 0$ and $g(x) \geqslant 0$

A bi-objective optimization problem is a MOP with only two objectives and it is defined as,

$$\min[f_1(x), f_2(x)],$$

$$x \in S, \tag{2.2}$$

where f, a scalar and S is the feasible region. Since there are two objectives, minimizing an objective will affect the other objective and hence a trustable trade off should be established to arrive at a compromising solution that minimizes both the objective functions. There is not a single optimal solution found, that optimizes all the objective functions, hence the concept of Pareto optimality as given by Pareto [23] is to be introduced.

The concept of Pareto optimum or Pareto efficiency is defined for arriving at these compromising solutions as the optimum solution obtained for a MOP is different from the Single objective optimization.

Definition 2.1 (Pareto Optimum or Pareto efficiency)**.** A point X is said to be Pareto optimum or an efficient solution for a multi objective problem if and only if there is no $x \in S$ (S is the set of all constraints) such that $f_i(x) < f_i(X)$ for all $i = 1, 2, \ldots, n$ for a minimization problem.

Definition 2.2 (Pareto Front)**.** Pareto front are the set of all solutions that are Pareto efficient.

The Pareto front has all Pareto efficient solutions *i.e.* the set of all non dominated points in the objective space. There are many approaches available for solving MOP to find these Pareto efficient solutions. Hwang *et al.* [15], mentioned three approaches for solving any multi objective problems, they are *a priori* method, interactive method and *posteriori*/generation method. In *a priori* method, a goal is set before the solution process and the method strives to arrive at the goal. In interactive method, a solution is achieved through interaction with the program, and hence it makes the convergence to a specified goal easier. In *posteriori*/generation methods, all efficient solutions to the problem are generated and the user decides one among them. In general, *posteriori* methods are preferred over all other methods, since it finds all solutions for the problem and then derives the best among them.

There are two well-known methods available under *posteriori*/generation methods as specified in Mavrotas [20]. They are weighted sum method that was initially found in the works of Zadeh [32] and ε constraint method. Both the methods convert the MOP into single objective problem and work towards it. Weighted sum method designates scalar weights and optimizes the set of objective function given as,

$$\sum_{i=1}^{n} w_i(f_i x_i), \text{ where } \sum_{i=1}^{n} w_i = 1 \text{ and } w_i > 0. \tag{2.3}$$

There are many ways for determining the suitable weights to optimize the solution. By varying the weights all non dominated points can be obtained. The disadvantage of the method is the selection of suitable weights.

In ε constraint method, one objective is optimized using the other objective function as constraints. According to Chankong *et al.* [6], and Cohon [9], the model in (2.1) can be written as,

$$\begin{gathered} \min(f_1(x)) \\ \text{Such that} \quad f_2(x) \leqslant \varepsilon_2 \\ f_3(x) \leqslant \varepsilon_3 \\ \cdots \\ f_n(x) \leqslant \varepsilon_n, \\ \text{and} \quad x \in S. \end{gathered} \tag{2.4}$$

Given a BOP as in (2.2) with two objectives, out of the defined objectives one is taken for optimization and the remaining objective is taken as a constraint converting (2.2) as,

$$\min f_1(x).$$

$$\text{Such that } f_2(x) \leqslant \varepsilon,$$

$$\text{and } x \in S. \tag{2.5}$$

The ε constraint method has several advantages over weighted sum method. As stated in the work of Mavrotas [20], for a weighted sum method appropriate choice of weights are needed for a better solution range. An advantage with ε constraint method is its ability to achieve efficient solution in a non convex region Pareto curve. Moreover, it is well-suited for bi objective optimization problems. A disadvantage is that the solution to the problem is dependent on the ε value. A proper choice of ε is required for a better Pareto optimal solutions.

3. Vehicle routing problem with limited refueling halts (VRPLHS)

VRPLHS is defined on a graph $G = (V, E)$ as an undirected connected graph with vertices V, named as customer locations, and edges E connecting the customers. The vertex set V is defined as $V = C \cup R \cup V_0$ where $C = \{C_1, C_2, \ldots, C_N\}$ are the set of customers, $R = \{R_1, R_2, \ldots, R_M\}$ are the set of *RFs* and V_0 is the depot. The depot has a set of homogenous vehicles K for serving the customers. Each vehicle should start from depot, serve the customer and return to the depot within the specified maximum time T. Customers are visited only once for servicing and RFs can be visited any number of times by a vehicle. The vertex set is distributed in 2-D space with X and Y coordinates. The vertex set includes N customer vertices, M refueling vertices and a single depot V_0.

Let t_i be the time taken to reach ith vertex. p_i is the service time taken at i, which is the time for serving a customer or fueling a vehicle and is kept a constant. The edge set is associated with c_{ij}, the distance or cost of travel from i to j and time tt_{ij}, the time taken to travel from i to j, where $i, j \in V$. Let L denote the total fuel level of a vehicle. Let z_i be the fuel level available on reaching the ith vertex. Let r be the constant fuel consumption rate along the distance travelled.

Let us assume that the vehicle starts its tour with full fuel capacity from the depot for serving a set of customers and the fuel level decreases at a constant fuel rate as it travels. Also it is assumed that, a minimum fuel level is maintained to reach the *RF*s/depot for refueling, so no vehicle is left stranded in the middle. When it is not possible to serve the next customer, the vehicle needs to be refueled. The tank is filled to its entire capacity either in *RF*s or in depot. Along the tour the vehicle is expected to halt for refuel a limited number of times given by ε.

VRPLHS is defined with the following objective and constraints.

Objective

- Minimize the total route cost.
- Minimize the refueling halts for the entire tour.

Constraints

- Each customer is serviced exactly once.
- Each vehicle route starts and ends at depot.
- The vehicle should visit the customer and return to the depot within a specified time.
- Vehicle is directed to the nearby fueling station/depot and not left stranded.

A decision variable x_{ijk} is introduced to know the customers who are served by a particular vehicle.

$$x_{ijk} = \begin{cases} 1, \text{ if } \textit{vehicle}\, k \text{ travels from } i \text{ to } j, i, j \in V \\ 0, \text{ otherwise.} \end{cases}$$

The mathematical formulation of the problem is,

$$\min \sum_{i,j \in V, k \in K, i \neq j} c_{ij} x_{ijk}, \tag{3.1}$$

$$\min \sum_{i \in V} y_{ir} \leqslant \varepsilon \; \forall\, r \in R, \tag{3.2}$$

where

$$y_{ir} = \begin{cases} 1 \text{ if vehicle travels from } i \text{ to } r, \\ 0 \text{ otherwise.} \end{cases}$$

Subject to

$$\sum_{i \in V,\, i \neq j} x_{ijk} = 1, \forall j \in V, k \in K, \tag{3.3}$$

$$\sum_{j \in V \setminus \{R\}, i \neq j} x_{ijk} = 1, \; \forall\, i \in R, \; k \in K, \tag{3.4}$$

$$\sum_{i,h \in V, h \neq i} x_{ihk} - \sum_{j,h \in V, h \neq j} x_{hjk} = 0, \forall \; k \in K, \tag{3.5}$$

$$\sum_{j \in V \setminus \{V_0\}} x_{V_0 jk} \leqslant K, \tag{3.6}$$

$$\sum_{j \in V \setminus \{V_0\}} x_{j V_0 k} \leqslant K, \tag{3.7}$$

$$t_i + (tt_{ij} + p_i) x_{ijk} - T(1 - x_{ijk}) \leqslant t_j, \; \forall\, i \in V, j \in V \setminus \{V_0\}, i \neq j, \tag{3.8}$$

$$t_j \leqslant T - (t_{j0} + p_j), \; \forall\, j \in V \setminus \{V_0\}, \tag{3.9}$$

$$t_{V_0} = p_{V_0} = 0, \tag{3.10}$$

$$0 \leqslant z_j \leqslant z_i - r.(c_{ij}x_{ijk}) + L(1 - x_{ijk}), \forall j \in C, i \in V, i \neq j, \tag{3.11}$$

$$z_j = L, \forall j \in V_0, j \in R, \tag{3.12}$$

$$z_j \geqslant L - r.(c_{ij}x_{ijk}), \forall i \in C, j \in V_0, j \in Ri \neq j. \tag{3.13}$$

The objective function of VRPLHS is given in (3.1) and (3.2). The objective of (3.1) is used to minimize the overall tour cost, (3.2) is used to minimize the number of refueling halts taken in a tour. The vehicle can travel from any vertex to any other vertex in V is given by (3.3). From a RF, either a customer or depot can be visited is specified in constraint (3.4). Constraint (3.5) is flow constraint to check that the same vehicle enters and leaves a vertex. The number of vehicles entering and leaving the depot is given in (3.6) and (3.7), and it should be within the specified maximum number of vehicle K. The total time taken to reach any $j \in V\backslash\{V_0\}$ is given through constraint (3.8). The total time taken to reach the depot from j is within the maximum time T, is checked with constraint (3.9). Initially, the time taken and the service time is kept as 0 in depot as given in constraint (3.10). Constraint (3.11) tracks the fuel level to reach vertex j that gets reduced at a constant rate r from vertex i. The refueling level upon reaching the RF or depot on route is specified through (3.12) and finally there is enough fuel to reach the depot or RF is guaranteed by (3.13) that prevent the vehicle being stuck in the middle of the tour. Some of the constraints of the model follow the model proposed by Erdogan *et al.* [10].

4. Particle swarm optimization (PSO)

PSO is an inspirational algorithm derived from the social behavior of birds, fish, *etc.* It is an evolutionary computation technique. It is a collective and iterative method with the emphasis on cooperation. Each potential solution (particle) to the problem is identified as a point in the search space of the PSO. Each point strives to achieve the best point obtained. Each particle has a velocity that can accelerate or decelerate depending on the current position of the point to reach the best position. Each point updates its position with its own experience (p_{best}) and experience gained by its neighbors (g_{best}). Each point is a particle and a vector and has dimension d. Let the ith particle X is defined as $(x_{i1}, x_{i2}, \ldots, x_{id})$ and is updated by velocity as,

$$v_{id}(t+1) = \omega.v_{id}(t) + C_p r_1[p_{id}(t) - x_{id}(t)] + C_g r_2[p_{gd}(t) - x_{id}(t)], \tag{4.1}$$

$$x_{id}(t+1) = x_{id}(t) + V_{id}(t+1), \tag{4.2}$$

where $v_{id}(t+1)$ is the velocity of the particle at time $(t+1)$ and $v_{id}(t)$ is velocity of particle at time t. If previous velocity is not updated, the particle will not move.

ω is the inertia factor and is used to balance the exploration and exploitation capability of a particle. C_p and C_g are cognitive and social (learning) parameters and are also called as trust parameters. C_p express the confidence of a particle on itself, C_g express the confidence of a particle on its neighbors. These parameters push the particle towards the best position. To have the stochastic behavior two random numbers r_1 and r_2 in the interval [0, 1] are generated. $p_id(t)$ and $p_{gd}(t)$ are the particle's personal (p_{best}) and global best (g_{best}), respectively. (4.1) is used to find the velocity of the particle and (4.2) finds the new position of the particle as a result of the velocity update.

The potential of PSO make it suitable for solving many MOP. A detailed review of PSO for MOP can be seen in the works of Parsopoulos *et al.* [25] and Coello *et al.* [8]. Vector Evaluate Particle Swarm Optimization (VEPSO) was proposed by Parsopoulos *et al.* [24] in line with multi objective genetic algorithm. Generally, there are two approaches available for handling MOP using PSO. In the first approach, each objective is taken separately and a solution is found, then a proper choice is made to select a Pareto optimal solution from among them. In the second approach, all the objectives are taken together and a solution is obtained. But, the determination of the best particle is difficult and hence calls for an external archive for storing the non dominated solutions. Several approaches in literature describe the proper selection and use of archive for storing non dominated solutions. Coello *et al.* [7] in their work MOPSO (Multi Objective Particle Swarm Optimization) introduced an external archive to store the Pareto optimal solutions, and for particle's diversification, mutation operator is used. They were the first to use the Pareto based PSO approaches. Agarwal *et al.* [1] described a fuzzy clustering based PSO, where a fuzzy clustering was used to maintain the archive and a mutation operator was used to diversify the population. A Multi Objective Comprehensive Learning PSO (MOCLPSO) was proposed by Huang *et al.* [14], where all the best particles are stored and used for velocity update, guiding the particle towards the best. Mousa *et al.* [21] suggested a local search method to improve the Pareto solution generation, by concentrating in less crowded areas. They are used for the archive update. A Time variant multi objective PSO was given by Tripathi *et al.* [29], where a time variant inertia and acceleration was used to maintain the exploration and exploitation capabilities and a mutation operator is used for divergence of the particle. The archive is updated with the non dominated solution using the combined population of swarm and the archive. If archive exceeds the maximum limit, then it is truncated and the more sparsely spread solution are retained.

4.1. PSO for BOP

A swarm of particles are initialized along with ε and a feasible solution is found. The fitness of each solution is calculated, and p_{best}, g_{best} is updated. The swarm has S members and the fitness function is ff_{ij} where $i = 1, 2, \ldots, S$ the size of swarm, and $j = 1, 2, \ldots, K$ are the number of ε values. For each ε, the best fitness

is taken and stored in an archive and it is the leader for that ε. If a new solution ff'_{ij} dominates ff_{ij} then the new solution is updated in the archive. Every other particle in the swarm follows the leader. Hence there will be K swarm equivalent to ε that contributes a fitness value. The set of all non dominated solution found for each ε make up the Pareto front.

4.2. PSO-GMO procedure for VRPLHS

The overall procedure of the Particle Swarm Optimization with Greedy Mutation Operator (PSO-GMO) algorithm for solving VRPLHS is described below.

Step 1. *Particle Representation:* PSO is basically a continuous optimization method. To make it to adapt to discrete methods certain modifications are necessary. One of the key issues is the successful representation of the particle of the problem and their corresponding mapping. Each particle is a customer sequence $(C_1, C_2, \ldots, C_N)$ but finding a solution to an optimization problem consists in finding a feasible solution that optimizes all the constraints of the problem. Since constraint satisfaction is necessary for finding the solution, refueling halts are included within the customer sequence obtained and the fitness is calculated. So a solution sequence has N customers $(C_1, C_2, \ldots, C_N)$, a Depot $C_{N+1}(V_0)$ and M $RF(C_{N+2}, C_{N+3}, \ldots, C_{N+M+1})$, where $(C_{N+2} = R_1, C_{N+3} = R_2, \ldots, C_{N+M+1} = R_M)$. This solution sequence forms the base for introducing few refueling halts. Hence only feasible solutions are available in the swarm. The introduction of refueling halts after the customer sequence formation also aid in using limited refueling halts.

For example, consider a set of 5 customers {1, 2, 3, 4, 5} with a depot {6} and 2 *RF* {7, 8} with 2 vehicles. A particle is a customer sequence {2, 4, 1, 5, 3}. A solution sequence is the introduction of depot and *RF*s as {6, 2, 4, 7, 1, 6, 5, 8, 3, 6} where 6 represents the depot and 7, 8 are *RF*s. The routes formed from the solution sequence are Route 1: {6, 2, 4, 7, 1, 6} and Route 2: {6, 5, 8, 3, 6}. The sequence with *RF* and depot is not taken for velocity update.

Step 2. *Swarm Initialization*: The initialization of swarm is a crucial step for faster convergence. Initializing the swarm with some preprocessing towards promising positions in the search guides to global best position. Around 75% of particles are generated using nearest neighbor heuristic (NNH). To select a seed customer three approaches are used namely (1) customer nearer to the depot and (2) customer farthest from the depot and (3) random selection. To have diversification of particles a random customer sequence is generated for the remaining 25% of swarm.

Step 3. *Transforming the particle to solution*: The swarm has S particles. Each particle is a customer sequence of dimension N. If the $Fuel_{\text{remaining}}$ is not sufficient in the vehicle to serve another customer then it is refueled. $Fuel_{\text{remaining}}$ is calculated using (4.3). Then a RF/depot is included in the solution sequence that is closer to the customer.

$$Fuel_{\text{remaining}} = Fuel_{\text{total}} - (distance\_travelled * Fuel\_consumption\_rate),$$

Algorithm 1: Nearest neighborhood heuristic

Initialize the seeds
for $i = 1$ to N
Calculate Euclidean distance $dist_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$,
where (x_i, y_i) is the co-ordinates of customer i and (x_j, y_j) is the co-ordinate of the seed.
choose a customer with minimum distance from the seed and mark as visited $C_{visited}$
Repeat
Find the customer $C_k$nearest to $C_{visited}$ using $dist_{visited,k}$
Mark C_k as visited and update $C_{visited} = C_k$
until all the customers are visited

where

$$Fuel_{\text{remaining}} \geqslant 0. \tag{4.3}$$

To introduce a new vehicle to the sequence, the travel time is checked. If the time is not sufficient to serve other customer, then the vehicle is redirected to the depot and a new vehicle is introduced. Time taken by a vehicle is calculated using (4.4)

$$Time\_taken_{\text{vehicle}} = Time\_taken_{\text{so far}} + (distance\_travelled/Average\_speed) + P. \tag{4.4}$$

P is the service time. Initially, time taken by the vehicle is zero at depot. $Time\_taken_{\text{so far}}$ is the time taken by the vehicle starting from the depot till it reaches a specified customer or RF. A new vehicle is introduced, if $(T - Time\_taken_{\text{vehicle}})$ is not sufficient to serve another customer, where T is the total time taken by a vehicle.

Step 4. *Fitness Function*: With the introduction of vehicle and RF, a solution sequence is obtained using Step 3. As stated earlier, a solution sequence is the route that is followed to serve the customer with trips to refueling stations. The fitness function for a PSO is the total cost of the route. To calculate the cost of route (fitness), the total cost incurred by all the vehicles to serve every customer along the tour, with the RF is considered. The cost of travel for a vehicle is calculated using (4.5), and the overall cost for the entire route for $k \in K$ vehicles is calculated using (4.6).

$$Total\_Cost_{\text{route}} = Cost_{\text{depot},\,i} + \sum_{i,j \in V/\{V_0\}} Cost_{i,j} + Cost_{i,\text{depot}} \,\forall\, i \in V \backslash \{V_0\} \tag{4.5}$$

$$Total\_Cost = \sum_{k \in K} Total\_Cost_{\text{route}} \tag{4.6}$$

Step 5. p_{best} and g_{best} update: Initially, p_{best} of a particle x_i in iteration $i = 1$ is, its calculated fitness value obtained using Step 4. During subsequent iterations, if the particle moves to an improved solution space, then the previous best position

available in p_{best} is updated, with the new best position of the particle, as specified in (4.7). Here, i is the iteration number.

$$p_{\text{best}} = \begin{cases} Total\_Cost(x_i), \ i = 1 \\ Total\_Cost(x_i), \ \text{if } Total\_Cost(x_i) < p_{\text{best}} \text{ for } i > 1. \end{cases} \tag{4.7}$$

Each particle has its own p_{best}. To obtain the global best fitness $g_{\text{best}}$for the entire swarm, the best of p_{best} is chosen, and stored as given in (4.8). Over the iterations, if the new g_{best} obtained is better than the current g_{best}, then it is updated.

$$g_{\text{best}} = \min(p_{\text{best}}) \, \forall \, i, \ 1 \leqslant i \leqslant S \tag{4.8}$$

Step 6. *Particle Encoding and Decoding*: The solution sequence obtained with the inclusion of *RF* and depot is in the feasible region and is used for fitness calculation. Each solution sequence has varied dimension with the inclusion *RF*. The inclusion of *RF*, in the solution sequence is dependent on the customer sequence under consideration. Hence, velocity update is made only for the customer sequence, which influences the addition of *RF* en route. Also, it eliminates the complexity that arises, with the varied dimension sequence. Since the problem is of discrete optimization, a conversion of the particle to particle's position in Multi Dimensional (*MD*) space is necessary, so each particle is converted to particle position using (4.9),

$$x_{ij} = x_{\min} + (x_{\max} - x_{\min})/n * ((y_{ij} - 1) + \text{rand}()), \tag{4.9}$$

where rand() is a random number in the interval [0, 1]. y_{ij} is the ith particle (customer sequence) with jth dimension that is to be converted to particle's position in *MD* space. n, is the total number of customers. $x_{\min}$ and $x_{\max}$ are the minimum and maximum boundary values, to make particles fall into a specified range. x_{ij} is the jth dimension of the ith particle encoded from particle y_{ij}. This procedure encodes the particle from a given customer sequence to particle's positions in the *MD* space.

To decode the particle position back to customer sequence, Rank of Value (ROV) method is used. It is used to rank the positions obtained through encoding. ROV uses smallest position value of a particle as rank 1, and incrementally ranks the successive position values. For example, for the 5 customer particle {2, 4, 1, 5, 3}, the encoded position value as calculated using (4.9) is {0.76, 1.25, 0.37, 1.85, 0.84}, where $x_{\min}$ and $x_{\max}$ are assumed as 0.0 and 2.0, respectively. Using *ROV* the position values are decoded back to customer sequence as {2, 4, 1, 5, 3}.

Step 7. *Velocity Update*: Each particle is made to search the solution space with the position update. It is achieved using velocity update. Each particle's new position is found by adding velocity as specified in (4.1) and (4.2). Generally, emphasis is given in varying the parameter of PSO, to make it work for challenging problems. In that perspective, inertia coefficient ω is added to PSO, to improve the performance. Inertia can be of two types, fixed and varying inertia. By the

study made by Han *et al.* [13] it is established that linearly decreasing inertia weight along with simulated annealing are found to be the best choice. Hence for ease and effectiveness, varying inertia weight is used. Generally, if varying inertia is used exploitation will be good. For a particle, an initial exploration guides a promising search space, and a local exploitation guide faster convergence. Usually, small values of inertia encourages local exploitation and large value increases global exploration. Hence inertia is kept a large value at the start and decreases during the iterative process to encompass a refined search in promising areas as given in equation (4.10)

$$\omega(t) = \omega_{\max} + (((t - I)/(1 - I)) * (\omega_{\min} - \omega_{\max})), \tag{4.10}$$

where t is the current iteration number, i is the total number of iterations. $\omega_{\min}$ and $\omega_{\max}$ are the initial and final inertia values.

For each particle, the feasibility is checked, if infeasible, a new particle is generated by controlling the velocity and inertia to make particle fall in the feasibility region.

Step 8. *Archive Update*: An external archive (A) is maintained to store the best of g_{best} values and is updated. The archive is used to store non dominated solutions. For each ε the archive gets updated with non-dominated solutions. If the current solution is better than the previous solution the archive is updated.

Algorithm 2: Archive Update

Let x_{current} be the solution in A and x_{new} be the new solution obtained
If $x_{\text{current}} \in A$ such that $x_{\text{current}} < x_{\text{new}}$ then A is not updated
Else if $x_{\text{current}} \in A$ and $x_{\text{current}} > x_{\text{new}}$ then $A = A - x_{\text{current}} \cup x_{\text{new}}$
Else if no x_{current} with $x_{\text{current}} < x_{\text{new}}$ in A then $A = A \cup x_{\text{new}}$

Step 9. *Greedy mutation operator* (*GMO*): One of the main problems with any evolutionary algorithms is premature convergences. In Genetic Algorithm (GA), it can be avoided by introducing the genetic operators. Since PSO does not have any genetic operators, to reach an optimum value PSO should interact with the particles. Since the problem to be solved involve many constraints, conventional approach of including an extra parameter will not work towards better optimum. To avoid local minima, a trajectory is expected that requires a transition of the particles from the current state to the next state. To achieve this, new mutation operator is introduced called as Greedy Mutation Operator (GMO).

To strengthen the search ability, many variations are defined in a PSO. In general, these variations made to PSO are problem dependent. Some work concentrates on introducing the crossover and mutation operator in PSO. To explore new areas in the search space, mutation operator of Genetic Algorithm (GA) can be used. The objective of mutation is, to increase the diversity of the swarm and to escape from local minima.

A Greedy Mutation Operator (GMO) is introduced that works as a local search technique. GMO is used to transform a particle X to another particle X^*, which tries to minimize the objective function *i.e.* $f(X^*) < f(X)$. Let $\{t \in T/t \leqslant N\}$, where t is the number of transitions required to transform from $X \to X^*$ and there can be N such transitions possible. A transition is defined as the change of position of a customer from positions i to j. The transition from $X \to X^*$ is successful, if X moves to an improved search space *IS*. To effect this transition, there should be an interaction with the particles yielding the solution. The transition Probability $X \to X^*$ is defined as,

$$\text{probability } (X \to X^*) \quad = \quad X^*(p_m)X(1-p_m).$$

To be more problem specific, if meaningful transitions are high there is a better chance of a particle to move to the improved search space. To get the maximum effect on the transitions, a trajectory with the nature of the problem is desired. According to GMO, two random cut points σ_1 and σ_2 are chosen. Let the particle P be represented as $(x_1, x_2, \ldots, x_n)$, where it is cut at x_l and x_k, let $(x'_1, x'_2, \ldots, x'_g)$ be the string within x_l and x_k. From x_l, nearest neighbor is chosen from the list and it is repeated till x'_g. If the number of transitions is more, then there is much chance of a particle to get into *IS*. For the probability of the particle $X \in IS$ is, $X^j(p_m)X^{N-j}(1-p_m)$, where $j < N$, where p_m is the mutation probability. If the search falls into a local minimum with j transitions resulting in $X \to X$, then a transition q is used with $q > j$ to enable $X \to X^*$ where $X^* \in IS$. The probability of the transition is,

$$\text{probability } (X \to X^*) = p_m^q(1-p_m^{N-q}), \text{ where } q > j$$

This transition made on the particle using this is more meaningful than a swap or inverse mutations.

The choice of the mutation parameter p_m plays a crucial role to escape from local minimum. Generally it is kept a constant. Since mutation is considered as a repairing technique a time varying mutation probability is introduced. The mutation rate is defined as $p_m = i/I$, where i is the current iteration number and I is the total number of iterations. During the initial iterations, mutation is required to have good transitions than in the later part, as all particles converges to the optimum.

Then, the obtained particles are crossed over with g_{best}, using a single point cross over operation. This further improves the solution.

After GMO and cross over, the particles are ready to be converted to solutions by inserting the *RF*s en route. This enhances the quality of the solution as *RF*s are introduced only if necessary achieving the second objective. Hence it is possible to reduce the number of refueling halts.

Algorithm 3: Greedy mutation operator (GMO)

Let the customer sequence be $(x_1, x_2, \ldots, x_n)$
Set $p_m = i/I$
If rand() $> p_m$ then
  Find the cut points σ_1, σ_2 using $\lceil no\_of\_customers * \text{rand}() \rceil$
  Let σ_1 and σ_1 cut $(x_1, x_2, \ldots, x_n)$ at x_l and x_k, respectively
  Let $(x'_1, x'_2, \ldots, x'_g)$ be the string within x_l and x_k
  Repeat
    Call NNH from x_k to each of $(x'_1, x'_2, \ldots, x'_g)$ and replace the string
  Until no further change
  Get the new sequence $(x''_1, x''_2, \ldots, x''_n)$
End
End

Algorithm 4: PSO-GMO for VRPLHS

Initialization Step
Initialize swarm S
Initialize the position of particle $X = (X_1, X_2, \ldots, X_n)$
Initialize velocity to zero and set ε values
Insert *RFs* according to (4.3) and append depot
Calculate fitness of particles
  Set $p_{\text{best}_1} = X_1, p_{\text{best}_2} = X_2, \ldots, p_{\text{best}_m} = X_m$
  Set g_{best} as best of p_{best}
  Update the archive with g_{best}
Iterative Step
Repeat till max_iterations
  For $i = 1$ to S
    Reframe the solution and eliminate the refueling halts
    Encode each solution to particle's position values using ROV method
    Update the velocity of each particle X_i using (4.1) and (4.2)
    If rand() $< p_m$ then
      Apply Greedy Mutation Operator
  End
  If rand() $< p_c$ then
      Perform single point cross over with g_{best}
    End
  Decode the particle's position to solution sequence
  Insert the refueling halts and depot
  Calculate fitness
  If $X_i > p_{best_i}$ then
    Update p_{best_i} as X_i
  End if
    If $p_{best_i} > g_{\text{best}}$ then
      Update g_{best} as p_{best}
  End if
  Update the archive
End
End repeat

TABLE 1. Parameter setting.

Parameter	Value
Cognitive acceleration parameter C_p	2.0
Social acceleration parameter C_g	2.0
Initial inertia weight $\omega_{\min}$	0.9
Final inertia weight $\omega_{\max}$	0.2
Particle's position $x_{\min}$ and $x_{\max}$	0.0 and 2.0
Swarm size S	30–45
Cross Over probability p_c	0.9
Number of iterations	1000–5000

5. Experimental results

The proposed PSO-GMO algorithm is executed on a computer with Intel Core i5-2400 processor with 3.10 GHz and 1.94 GB of RAM. The algorithm is implemented in MatlabR2011a. The obtained results are compared with the data sets of Erdogan *et al.* [10] used for solving Green Vehicle Routing Problem (GVRP).

The data set of Erdogan *et al.* [10] is divided into four categories. There are 4 groups of 10 different data sets created with 20 customers each. First 10 data sets have uniformly distributed customers (S1), next 10 for clustered customers (S2); Third data set is a combination of randomly distributed customers and clustered customers (S3). Fourth data set is to study the impact of fueling stations (S4). First two data sets have 3 fixed fueling stations, whereas third has 6 randomly generated fueling stations and fourth data set increments the fueling stations from 2 to 10. In their work they implemented two well-known algorithms namely the Modified Clark and Wright Savings Algorithm (MCWS) and Density Based Clustering Algorithm (DBCA). Now the proposed algorithm is targeted to solve the problem as a bi-objective optimization problem with two objectives using PSO-GMO. The parameter setting of the algorithm is given in Table 1.

These are the parameters that are influential to guide the solution in the search space. When the value of cognitive acceleration C_p increases, particles are attracted to p_{best}, similarly, when social acceleration C_g increases the particles are attracted to g_{best}. A legitimate choice followed in literature is to prefer $C_p = C_g = 2.0$ as specified in Ozcan *et al.* [22]. A linearly decreasing inertia weight is used with the initial inertia weight $\omega_{\min} = 0.9$ and final inertia weight $\omega_{\max} = 0.2$. $x_{\min}$ and $x_{\max}$ are the boundary values of the particle's position. The swarm size is kept at 30 for smaller size instances and based on size of instances proportionately incremented till 45. The crossover probability of particle is set at 0.9. As specified in Table 1, the number of iterations is from 1000 to 5000. For smaller instances the maximum number of iterations is kept as 2000 and for large instances the maximum iteration is kept as 5000 iterations.

As specified in Erdogan *et al.* [10], the fuel tank capacity is fixed as 60 liters and the fuel consumption rate is assumed to be 0.2. The average vehicle speed is

40 miles per hour; the total duration is 11 h. Service time at customer location and at refueling station is 30 min and 15 min respectively. In each instance, all the locations are given with their latitude and longitude values. For calculation they are converted to points in Euclidean space. Each instance specifies the distribution, namely uniform (U) or Clustered (C), total number of refueling stations available and the total number of customers. For example, in 20c3sU1 it is 20 customers, 3 fueling stations with uniformly distributed customers and instance number 1. S1_10i6s indicates S1 data set, 10th instance with 6 fueling stations.

The results obtained by PSO-GMO with a sample of 10 runs are shown in Tables 2 to 6. These tables depict the values got for the Number of Routes (NR), Number of Customers (NC), Cost of route (Cost), the Number of Refueling Halts (NRFH) and the Run Time (RT) in seconds obtained for the maximum number of iterations for an average of 10 sample runs. The data set is run with traditional PSO and with PSO-GMO. The results show that, when PSO is aligned with GMO, the results are good. With GMO, the particles are triggered to move into a new search space and hence there is more possibility of obtaining an optimal solution. It is observed that the results obtained for the small data sets produce better cost of tour as compared to the published results using MCWS and DBCA algorithms. Also, with a restriction placed on the number of times a visit is made to the RFs using ε, a few RFs are visited by the vehicle over the entire tour. In most cases, ε is taken as the number of fueling stations as specified in the data sets. In the following tables, the effect of a reduced number of refueling halts, and their impact on the overall tour cost is concentrated and the results are tabulated. The results in bold indicates the same or best solution got by PSO-GMO than MCWS/DBCA and the (**) is used to indicate the best solution got using PSO-GMO.

Some of the observations from the results show that, almost 4 instances did not stop for refueling. About 17 instances have used refueling stations only once. For 19 instances new best solution is produced using the PSO-GMO. About 5 of the instances have reduced the usage of the number of vehicles. The algorithm is stringent in producing the feasible solution with the limited number of refueling halts. As observed from Table 2, in most of the instances the vehicle used 1 or 2 refueling halts. In cases where the vehicle stopped once, stresses the necessity of only one refueling station. In case of Table 5 though the number of fueling stations are incremented from 2 to 10, the tour completed with at most 2 refueling halts and produced good results. In some cases, with the restriction placed on the constraints, it is not possible to serve all customers.

Figures 1 to 4 show the routes obtained for some data sets. Some of the instances show positive correlation *i.e.* as the refueling halts are increased the cost also increases. Some data set has shown the negative correlation, as observed in 20c3sU9, S1_4i4s, as the number of halt increases the cost decreases. For most of the instances, a single best solution is obtained.

Table 6 shows the comparison obtained on the average total cost by each instance using the PSO-GMO algorithm, and is an indicative of the performance improvement obtained by the proposed method PSO-GMO.

TABLE 2. Comparison of PSO-GMO for uniformly distributed customer data sets.

Dataset	CPLEX			MCWS			DBCA			PSO				PSO-GMO				
	NR	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*RT*(secs.)
20c3sU1	5	20	1797.51	6	20	1818.35	6	20	1797.51	6	20	2	1834.12	6	20	2	**1756.30****	118.65
20c3sU2	6	20	1574.82	6	20	1614.15	6	20	1613.53	6	20	1	1599.43	6	20	1	**1595.79**	115.56
20c3sU3	7	20	1765.9	7	20	1969.64	7	20	1964.57	7	20	1	**1703.25**	7	20	1	**1703.25****	116.79
20c3sU4	5	20	1482.00	5	20	1508.41	5	20	1487.15	5	20	2	1546.26	5	20	2	**1482.00**	114.63
20c3sU5	6	20	1689.35	6	20	1752.73	6	20	1752.73	6	20	2	1779.55	6	20	2	**1696.99**	115.90
20c3sU6	6	20	1643.05	6	20	1668.16	6	20	1668.16	6	20	1	**1666.98**	6	20	1	**1633.67****	112.54
20c3sU7	6	20	1715.13	6	20	1730.45	6	20	1730.45	6	20	0	1771.45	6	20	0	**1713.66**	112.86
20c3sU8	6	20	1709.43	6	20	1766.36	6	20	1718.43	6	20	1	1857.12	6	20	1	**1718.00**	118.39
20c3sU9	6	20	1708.84	6	20	1714.43	6	20	1714.43	6	20	1	1719.46	6	20	3	**1708.10****	121.59
																1	**1718.24**	122.50
20c3U10	8	20	1261.15	5	20	1309.52	5	20	1309.52	5	20	1	1321.57	5	20	1	**1205.4****	85.906

TABLE 3. Comparison of PSO-GMO for clustered customer data sets.

Dataset	CPLEX			MCWS			DBCA			PSO				PSO-GMO				
	NR	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*RT* (secs.)
20c3sC1	5	20	1235.21	5	20	1300.36	5	20	1300.62	6	20	0	**1295.21**	6	20	0	**1177.57****	100.46
20c3sC2	5	19	1539.94	5	19	1553.53	5	19	1553.53	6	19	1	**1553.34**	6	19	1	**1539.97**	109.22
20c3sC3	4	12	985.4	4	12	1083.12	4	12	1083.12	7	12	1	**900.33**	7	12	1	**879.20****	78.06
20c3sC4	5	18	1080.16	5	18	1135.90	5	18	1091.78	5	18	1	1128.67	5	18	1	1090.29	102.57
20c3sC5	7	19	2190.68	7	19	2190.68	7	19	2190.68	6	19	3	2261.46	6	19	3	2209.87	123.50
20c3sC6	9	17	2785.86	9	17	2883.71	9	17	2883.71	6	17	5	**2411.66**	6	17	5	**2411.66****	123.50
20c3sC7	5	6	1393.98	5	6	1701.40	5	6	1701.40	6	6	4	**1393.98**	6	6	4	**1393.98****	122.42
20c3sC8	10	18	3319.71	10	18	3319.74	10	18	3319.74	6	18	7	**3139.76**	6	18	7	**3129.76****	73.21
20c3sC9	6	19	1799.95	6	19	1811.05	6	19	1811.05	6	19	2	**1811.45**	6	19	2	**1728.29****	125.92
20c3sC10	8	15	2583.42	8	15	2648.84	8	15	2644.11	5	15	5	**2533.07**	5	15	5	**2531.07****	114.76

TABLE 4. Comparison of PSO-GMO for both uniform and clustered customer data sets.

Dataset	CPLEX			MCWS			DBCA			PSO				PSO-GMO				
	NR	*NC*	*Cost*	*NR*	*NCFH*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*NR*	*NC*	*NRFS*	*Cost*	*RT* (secs.)
S1_2i6s	6	20	1578.15	6	20	1614.15	6	20	1614.15	6	20	1	**1614.37**	6	20	1	**1566.21****	135.07
S1_4i6s	5	20	1438.89	5	20	1567.30	5	20	1541.46	6	20	2	**1503.68**	6	20	2	**1438.89**	129.81
S1_6i6s	6	20	1571.28	6	20	1616.20	6	20	1616.20	6	20	2	1751.88	6	20	2	1616.16	121.03
S1_8i6s	6	20	1692.34	6	20	1902.51	6	20	1882.54	7	20	3	1979.67	7	20	3	**1831.55****	118.78
S1_10i6s	5	20	1253.32	5	20	1309.52	5	20	1309.52	5	20	0	1389.34	5	20	0	1321.77	119.14
S2_2i6s	6	20	1645.8	6	20	1645.80	6	20	1645.80	6	20	2	1684.66	6	20	2	1669.88	103.01
S2_4i6s	6	19	1505.06	6	19	1505.06	6	19	1505.06	6	19	1	1536.87	6	19	1	**1504.98****	116.65
S2_6i6s	10	20	2842.08	10	20	3115.10	10	20	3115.10	10	20	7	**2640.65**	10	20	7	**2640.65****	169.23
S2_8i6s	9	16	2549.98	9	16	2722.55	9	16	2722.55	7	16	4	**2251.46**	7	16	4	**2251.46****	107.96
S2_10i6s	6	16	1605.65	6	16	1995.62	6	16	1995.62	7	17	5	2174.04	7	17	5	2004.80	112.14

TABLE 5. Comparison of PSO-GMO for both uniform and clustered customer data sets.

Dataset	CPLEX			MCWS			DBCA			PSO				PSO-GMO				
	NR	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*RT* (secs.)
S1_4i2s	6	20	1582.22	6	20	1582.2	6	20	1582.2	6	20	1	1602.66	6	20	1	**1582.58**	125.32
S1_4i4s	6	20	1504.1	6	20	1580.52	6	20	1580.52	6	20	1	1598.89	6	20	1	**1486.85****	101.82
																2	**1475.92****	99.67
S1_4i6s	5	20	1397.98	5	20	1561.29	5	20	1541.46	6	20	2	1517.13	6	20	2	**1475.92**	122.98
S1_4i8s	6	20	1376.98	6	20	1561.29	6	20	1561.29	6	20	2	1510.22	6	20	2	**1473.93**	126.09
S1_4i10s	5	20	1397.28	5	20	1536.04	5	20	1529.73	5	20	2	1486.75	5	20	2	**1486.75**	125.09
S2_4i2s	5	18	1080.16	5	18	1135.89	5	18	1117.32	5	18	0	1117.32	5	18	0	**1109.26**	100.78
S2_4i4s	6	19	1466.9	6	19	1522.72	6	19	1522.72	6	19	1	1533.67	6	19	1	**1466.91**	118.03
S2_4i6s	6	20	1454.96	6	20	1786.21	6	20	1730.47	6	20	1	1521.45	6	20	1	**1353.12****	126.07
S2_4i8s	6	20	1454.96	6	20	1786.21	6	20	1786.21	6	20	1	1521.45	6	20	1	**1353.12****	115.23
S2_4i10s	6	20	1454.93	6	20	1783.63	6	20	1729.51	6	20	1	1521.45	6	20	1	**1353.12****	120.64

TABLE 6. Comparison of the average total cost of the proposed method with MCWS and DBCA.

Data Set	MCWS	DBCA	PSO	PSO-GMO
1	1685.2	1675.6	1679.91	**1621.31**
2	1962.8	1957.9	1841.69	**1809.16**
3	1898.8	1894.8	1852.62	**1804.83**
4	1583.6	1583.6	1493.09	**1413.06**

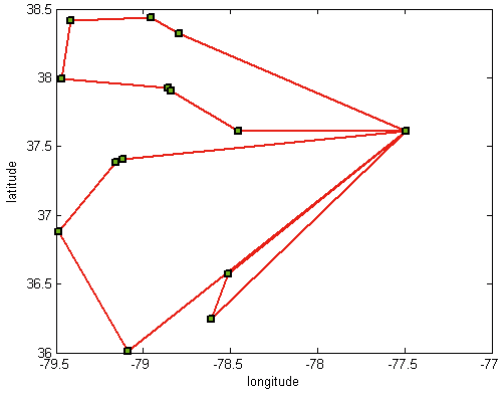

FIGURE 1. Route for 20c3sC3.

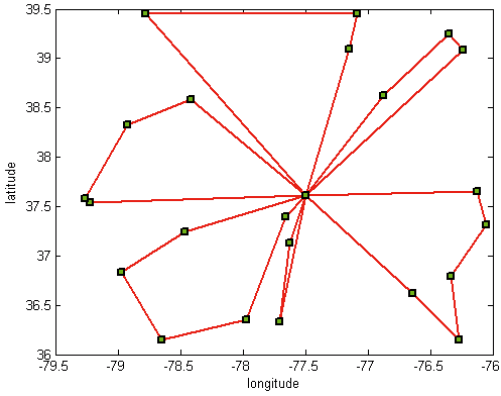

FIGURE 2. Route for 20c3sU1.

The proposed PSO-GMO algorithm is applied on large data sets with customers from 100 to 500 and its performance is compared against PSO and MCWS and DBCA algorithms. Each data set has 21 *RF*s. The results are tabulated in Table 7. A new best solution was obtained for instance with 500 customers. In about 4 data sets there is an increase in the number of customers being served.

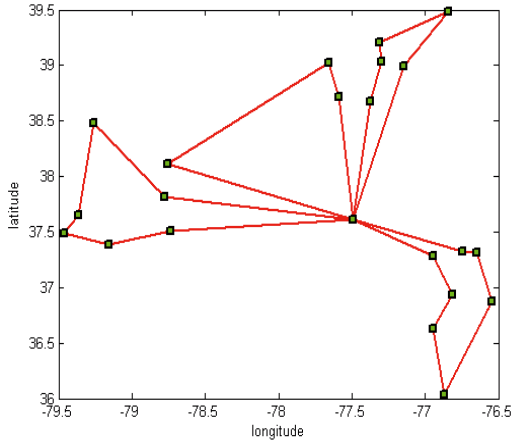

FIGURE 3. Route for 20c3sC1.

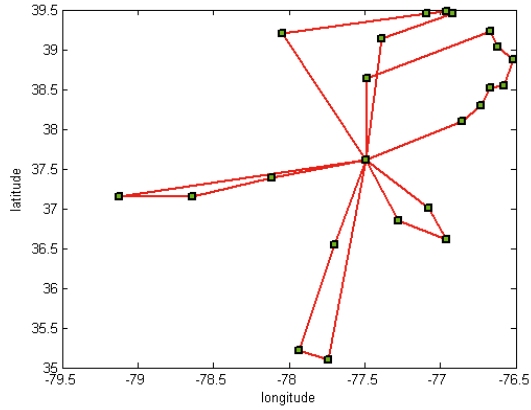

FIGURE 4. Route for s2_4i10s.

Compared to the small data sets where most instances obtained a single best solution, some of the large data sets show the contradictory behavior. For example for data set with 200, 400 and 450 customers, the cost and the refueling stops show a negative correlation. The remaining data sets show a positive correlation and hence a single best solution is reported. The observed Pareto front for data sets with 300 and 400 customer data sets is shown in Figure 5 and 6, respectively. New best solution was arrived at for data set with 500 customers. For the other data sets there is an increase in the number of customers served. More number of customers is served using PSO-GMO than with the existing algorithm.

The problem is concentrated on reducing the number of refueling halts and arrived at a solution. Though there are 21 fueling stations for large instance,

TABLE 7. Comparison of PSO-GMO with other algorithms for large data sets.

Large sample	MCWS			DBCA			PSO				PSO-GMO				
	NR	*NC*	*Cost*	*NR*	*NC*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*NR*	*NC*	*NRFH*	*Cost*	*RT* (secs.)
111 customers	20	109	5626.64	20	109	5626.64	19	109	6	5876.89	19	109	6	**5626.64**	3648.24
200 customers	35	190	10428.59	36	191	10413.59	38	192	13	11115.87	38	**192**	13	**10428.87**	9048.24
													12	11256.50	9768.24
300 customers	49	281	14242.56	49	281	14229.92	54	284	18	15124.69	54	**284**	18	14230.8	20424.24
350 customers	57	329	16471.79	57	329	16460.30	62	330	23	18139.28	62	**330**	21	**16471.20**	23453.23
400 customers	67	378	19472.10	66	373	19099.04	72	378	26	20877.46	72	**378**	28	**19472.10**	27566.64
													27	20752.34	26486.64
450 customers	75	424	21854.17	75	424	21854.19	75	428	28	22982.21	78	**428**	28	22165.17	33686.64
													29	21854.19	34406.64
500 customers	83	471	24527.46	84	471	24517.08	84	471	33	25077.56	84	471	33	**24431.25****	34954.64

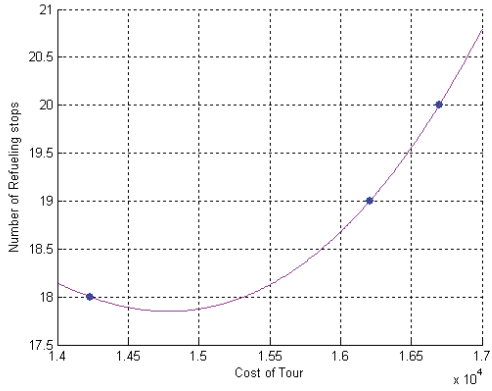

FIGURE 5. Pareto front of 300 customer instance.

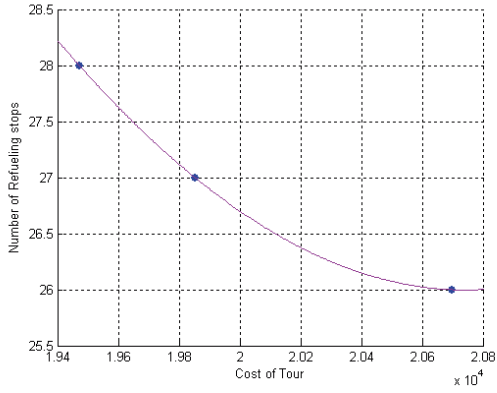

FIGURE 6. Pareto front of 400 customer instance.

the instance 111 customers has stopped only 6 times for refueling, where some stations are visited more than once. The solution trace obtained by PSO-GMO is depicted in Figure 7 for different data sets. The route obtained for instance with 111 customers and 500 customers is shown in Figures 8 and 9, respectively.

The results obtained by the proposed method are promising and better than the published results. This paper addresses the importance of the refueling stations from the perspective of cost and time involved, and in the number of halts taken by the vehicles for refueling. Additional information obtained with the implementation is, some *RFs* are visited frequently and some are never visited. Taking this as a cue, the work can be further extended to include optimal placement of warehouse/service station at that location of *RFs* that are frequently visited.

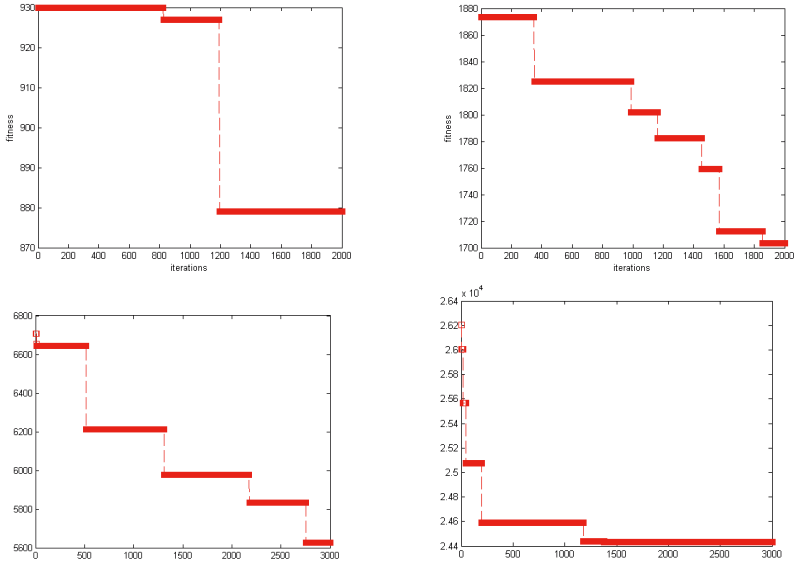

FIGURE 7. Trace of PSO_GMO algorithm for different data sets. (a) Instance 20c3sC3, (b) instance 20c3sU3, (c) instance 111 customer, (d) instance 500 customers.

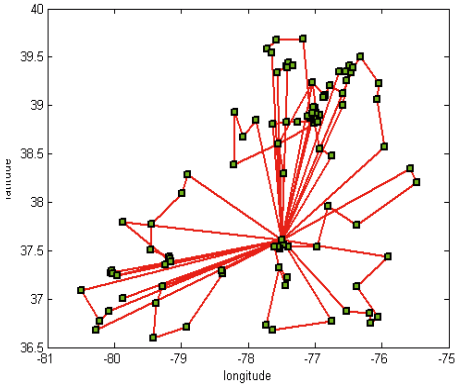

FIGURE 8. Route for 111 customers.

6. CONCLUSION

VRPLHS is introduced in this paper. The problem is modeled as a ε constraint bi objective problem and solved using PSO-GMO. The trajectory expected by PSO to escape from local minima is solved through a devised mutation operator GMO. The results obtained by PSO and PSO-GMO depict the influence of GMO in the search space for a good solution. The algorithm is compared with the available data

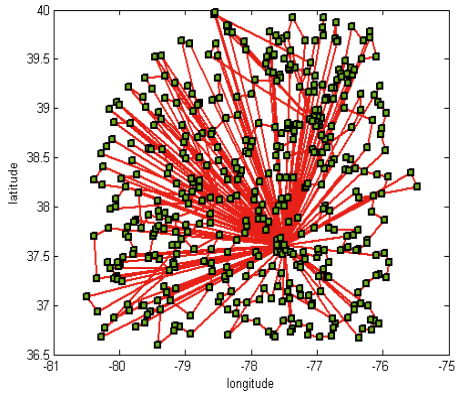

FIGURE 9. Route for 500 customers.

sets and proved to work better than the published results. Any organization that plan to route vehicle with refueling option should consider a tour with minimum refueling halts, as the cost can be substantially reduced. There are many insights observed. It opens up many areas of research in VRP with refueling as a constraint. This work can be extended for heterogeneous vehicles, vehicles with different fuel types, fueling stations with limited fuel capacity level. As the support towards green environment is adverse it may provide a different arena of research.

REFERENCES

[1] S. Agarwal, B.K. Panigrahi and M.K. Tiwari, Multi objective particle swarm algorithm with fuzzy clustering for electrical power dispatch. *IEEE Trans. Evol. Comput.* **12** (2008) 529–541.

[2] A. Artmeier, J. Haselmayr, M. Leucker and M. Sachenbacher, The shortest path problem revisited: Optimal routing for electric vehicles, in *Advances in Artificial Intelligence*. Springer, Berlin, Heidelberg (2010) 309–316.

[3] S. Aviral, P. Joseph and V. Venkatasubramanian, An optimization framework for cost effective design of refueling station infrastructure for alternative fuel vehicles. *Comput. Chem. Eng.* **35** (2011) 1431–1438.

[4] T. Bektas and G. Laporte, The pollution-routing problem. *Transp. Res. Part B* **45** (2011) 1232–1250.

[5] M. Caramia and P. Dell'Olmo, *Multi-objective management in freight logistics*. Springer (2008).

[6] V. Chankong and Y.Y. Haimes, *Multiobjective decision making: theory and methodology*. Elsevier Science, New York (1983).

[7] C.A.C. Coello and M.S. Lechuga, MOPSO: A proposal for multiple objective particle swarm optimization, in *Proc. of 2002 Congress on Evolutionary Computation part of the 2002 IEEE World Congress on Computational Intelligence*. IEEE Press, Hawaii (2002) 1051–1056.

[8] C.A.C. Coello and G.T. Pulido, Lechuga MSHandling multiple objectives with particle swarm optimization. *IEEE Trans. Evol. Comput.* (2004) **8** 256–279.

[9] J.L. Cohon, *Multiobjective Programming and Planning*. Academic Press, New York (1978).

[10] S. Erdogan and E. Miller-Hooks, A green vehicle routing problem. *Transp. Res. Part E* **48** (2012) 100–114.

[11] M. Gendreau, G. Laporte and J.Y. Potvin, *Metaheuristics for the vehicle routing problem*, Technical Report CRT-963. Centre de Recherche sur les Transports, Université de Montréal (1999).
[12] B.L. Golden and A.A. Assad, *Vehicle routing: methods and studies.* North-Holland, Amsterdam (1988).
[13] W. Han, P. Yang, H. Ren and J. Sun, Comparison study of several kinds of inertia weight for PSO, in *Proc. of IEEE international conference on progress in informatics and computing.* IEEE Comput. Soc. (2010) 280–284.
[14] V.L. Huang, P.N. Suganthan and J.J. Liang, Comprehensive learning particle swarm optimizer for solving multi objective optimization problems. *Int. J. Intell. Syst.* **21** (2006) 209–226.
[15] C.L. Hwang, A.S.M. Masud, S.R. Paidy and K.P. Yoon, Multiple Objective Decision Making, Methods and Applications: a State of the Art Survey, *Lecture Notes in Economics and Mathematical Systems*, 164. Springer-Verlag, Berlin (1979).
[16] J. Kennedy and R.C. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, San Francisco (2001).
[17] Y. Kuo, Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem. *Comput. Ind. Eng.* **59** (2010) 157–165.
[18] G. Laporte and I.H. Osman, Routing problems: a bibliography. *Ann. Oper. Res.* **61** (1995) 227–262.
[19] D. Levy, K. Sundar and S. Rathinam, Heuristics for routing heterogenous unmanned vehicles with fuel constraints. *Math. Problems Eng.* (2014), DOI:10.1155/2014/131450.
[20] G. Mavrotas, Effective implementation of the ε constraint method in Multi-Objective Mathematical Programming problems. *Appl. Math. Comput.* **213** (2009) 455–465.
[21] A.A. Mousa, M.A. El-Shorbagy and W.F. Abd-El-Wahed, Local search based hybrid particle swarm optimization algorithm for multi objective optimization. *Swarm Evol. Comput.* **3** (2012) 1–14.
[22] E. Ozcan and C. Mohan, Particle swarm optimisation: Surfing the waves, in *Proc. of IEEE international congress on evolutionary computation*. IEEE Computer Society (1999) 1939–1944.
[23] V. Pareto, Manuale di economica politica, societa editrice Libraria, Milan, Translated into English by A.S. Schwier, as *Manual of political economy*, edited by A.S. Schwier and A.N. Page, A.M. Kelly. New York (1971).
[24] K.E. Parsopoulos and M.N. Vrahatis, Particle swarm optimization method in multi objective problems, in *Proc. of 2002 ACM Symposium on Applied Computing* (2002) 603–607.
[25] K.E. Parsopoulos, M.N.Vrahatis, in *Multiobjective particle swarm optimization approaches, multi-objective optimization in computational intelligence: Theory and practice*, by Lam Thu Bui, Sameer Alam (Eds.), Chapter 2, pp. 20–42. IGI Global (2008).
[26] D. Pisinger and S. Ropke, *A general heuristic for Vehicle routing problem*. Department of Computer Science, University of Copenhagen (2005).
[27] A. Sbihi and R.W. Eglese, Combinatorial optimization and green logistics. *4OR-Q J. Oper. Res.* **5** (2007) 99–116.
[28] P. Toth and D. Vigo, An overview of vehicle routing problems, in *The vehicle routing problem*. edited by P. Toth and D. Vigo. SIAM monographs on discrete mathematics and applications (2002) 1–26.
[29] P.K. Tripathi, S. Bandhopadhyay and S.K. Pal, Multi objective particle swarm optimization with time variant inertia and acceleration coefficients. *Inform. Sci.* **177** (2007) 5033–5049.
[30] Y.W. Wang and C.R. Wang Locating passenger vehicle refueling stations. *Transp. Res. E* **46** (2010) 791–801.
[31] Y. Xiao, Q. Zhao, I. Kaku and Y. Xu, Development of a fuel consumption optimization model for the capacitated vehicle routing problem. *Comput. Oper. Res.* **39** (2012) 1419–1431.
[32] L. Zadeh, Optimality and non-scalar-valued performance criteria. *IEEE Trans. Autom. Control* **8** (1963) 59–60.