Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Two solution representations for solving multi-depot vehicle routing problem with multiple pickup and delivery requests via PSO

Voratas Kachitvichyanukul [b,*], Pandhapon Sombuntham [a], Siwaporn Kunnapapdeelert [b]

[a] Accenture Solution Company Limited, Thailand
[b] Industrial and Manufacturing Engineering, Asian Institute of Technology, Klongluang, Pathumthani 12120, Thailand

ARTICLE INFO

Article history:
Available online 20 April 2015

Keywords:
Vehicle routing
Multi-depot
Multiple pickup and delivery requests
Particle swarm optimization
Solution representation

ABSTRACT

Two solution representations for solving the generalized multi-depot vehicle routing problem with multiple pickup and delivery requests (GVRP-MDMPDR) is presented in this paper. The representations are used in conjunction with GLNPSO, a variant of PSO with multiple social learning terms. The computational experiments are carried out using benchmark test instances for pickup and delivery problem with time windows (PDPTW) and the generalized vehicle routing problem for multi-depot with multiple pickup and delivery requests (GVRP-MDMPDR). The preliminary results illustrate that the proposed method is capable of providing good solutions for most of the test problems.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

The vehicle routing problem (VRP) was first proposed by Dantzig and Ramser (1959). The key decision is to determine the optimal sequence of customers to be visited by each vehicle, which satisfied such criteria as travel distance, time, and cost involved in the operation. Several variants of VRP have been studied to address the wide variety of conditions in real world applications. For example, capacitated VPR (CVRP) addressed the VRP under the limitations of vehicle's capacity. Heterogeneous fleet VRP (HVRP), an extension of CVRP, has similar characteristics as CVRP but allows the vehicles to have different capacities. VRP with time windows (VRPTW) requires that the customer must be served within a specific time interval. VRP with simultaneous pickup and delivery (VRPSPD) deals with the customer requirement that involves simultaneous pickups and deliveries, and multi depot VRP (MDVRP) contains more than one depot.

The vehicle routing problem with pickup and delivery (VRPPD) is a problem where the service required by customers may be both the pickups and deliveries of commodities. It can be further subdivided as delivery-first and pickups-seconds, mixed pickups and deliveries, and simultaneous pickups and deliveries. Berbeglia, Cordeau, and Laporte (2009) classified the pickup and delivery problem (PDP) into three groups according to the pickup–delivery relation. The first group is many-to-many problem which any node can be a source or destination for any commodity. A commodity may be picked up from one of many locations, and delivered to one of several locations as well. The second group is one-to-many-to-one problem where commodities are originally available at the depot for delivery to customers. The commodities available at the customers must be first picked up and delivered back to the depot. Such characteristic is normally found in many previous studies of VRPPD. The last one, one-to-one problem, each commodity can be called a request which has a given origin and destination.

Variants of VRP including PDs with various constraints are known to be the NP-hard problems which may require excessively long computational time to find the optimal solutions for large problems. Consequently, numerous metaheuristic approaches have been developed for solving large VRP problems. Neural Networks (NNs), Genetic Algorithm (GA), Simulated Annealing (SA), Tabu Search (TS), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO) had all been tried as solution methods for the problems although these methodologies do not guarantee to find optimal solutions but they are quite capable of providing good solutions or near optimal solutions within the reasonable time.

Particle Swarm Optimization (PSO) was first proposed by Kennedy and Eberhart (1995) and it has been demonstrated to be very effective as a solution method for many intractable combinatorial problems. Several real-valued PSO algorithms were developed for solving variants of VRP such as CVRP, VRPSPD, and VRPTW; (Ai & Kachitvichyanukul (2009a, 2009b, 2009c)); and the results confirmed that PSO can be very effective for solving generalized vehicle routing problems.

Sombuntham and Kachitvichyanukul (2010) proposed a particle swarm optimization algorithm for multi-depot vehicle routing

* Corresponding author.
E-mail addresses: voratas@ait.ac.th (V. Kachitvichyanukul), Pandhapon.Sombuntham@accenture.com (P. Sombuntham), siwaporn.kunnapapdeelert@ait.ac.th (S. Kunnapapdeelert).

http://dx.doi.org/10.1016/j.cie.2015.04.011
0360-8352/© 2015 Elsevier Ltd. All rights reserved.

problem with multiple pickup and delivery requests. The solution representation SD1 is a general one and the algorithm is not restricted only to PDPTW problem. However, the computational experiments are only reported for the test instances from Li and Lim (2001) for the PDPTW.

This paper extends the SD1 solution representation from Sombuntham and Kachitvichyanukul (2010) and proposes two new solution representations, named SD2 and SD3, for solving both PDPTW and GVRP-MDMPDR. Computational experiments are carried out using the test instances for the PDPTW from Li and Lim (2001) and test problem instances for GVRP-MDMPDR from Sombuntham and Kunnapapdeelert (2012). The preliminary results show that the algorithm is able to provide good solutions to most of the test problems.

2. Problem description

A *customer* is generally defined as a place which must be served by a vehicle from a depot in many previous studies of vehicle routing problem. In this paper, the term customer is replaced by a more general term, *location*, to reflect that a *location* may have both pickup and delivery requirements. For pickup and delivery problem, a "request" is a commodity that requires a vehicle to collect from an origin and deliver to a destination. Although direct shipment is allowed, each location can play only one of these three roles: pickup location, delivery location, and vehicle station or depot. However, in real situation, a direct shipment may be required and each location can play multiple roles. Each location may have several items to be picked up and delivered to several other locations. This VRP that allows a location to play multiple roles and may have various pickup items delivered to different locations is denoted as the generalized vehicle routing problem for multiple depots with multiple pickup and delivery requests (GVRP-MDMPDR). This situation can be depicted as shown in Fig. 1.

In this paper, vehicle routes are formed in such a way that.

(1) the total routing cost is minimized;
(2) the number of vehicle is minimized;
(3) the fulfilled demand is maximized.
In addition, the following restrictions must be met:
(4) each request is served exactly once by a vehicle;
(5) the load of a vehicle never exceeds its capacity;
(6) each route starts and ends at the indicated terminals, which may or may not be a depot;
(7) the number of vehicles used do not exceed maximum number of available vehicles;
(8) the total duration of each route (including travel and service time) does not exceed a preset limit.

3. Mathematical model formulation

The mathematical model for the problem is an extension of the model found in Ropke and Pisinger (2006). The objective function in this study considers the total distance, the number of vehicles used, and the number of fulfilled request and all vehicles are allowed to serve any requests if the assigned request does not exceed the vehicle's capacity. Moreover, a location can have multiple requests, i.e., a location can simultaneously be a pickup node, a delivery node, and a depot. The model was first reported in Sombuntham and Kachitvichyanukul (2010) and it is included here for ease of reference. The definitions of sets, input parameters, and the main decision variables in the model are described below:

Sets	
P	Set of pickup nodes for request $P = \{1, 2, \ldots, R\}$ where R is the total number of requests
D	Set of delivery nodes for request $D\{1 + R, 2 + R, \ldots, 2R\}$ where $r + R$ denotes destination node of request r
N	Set of all pickup and delivery nodes $N = P \cup D$
N_k	Subset of nodes visited by vehicle k
K	Set of all vehicles
V	Set of all nodes. $V = N \cup \{\tau_1 \ldots \tau_m\} \cup \{\tau'_1 \ldots \tau'_m\}$ where τ_k represents node of the start station of vehicle $k, k \in K$ τ'_k represents node of end station of vehicle $k, k \in K$
A	Set of (i,j) which is an arc from node i to node j, where $i,j \in V$
Input parameters	
C_k	Capacity of vehicle $k \in K$
f_k	Fixed cost of vehicle $k \in K$ if it is used

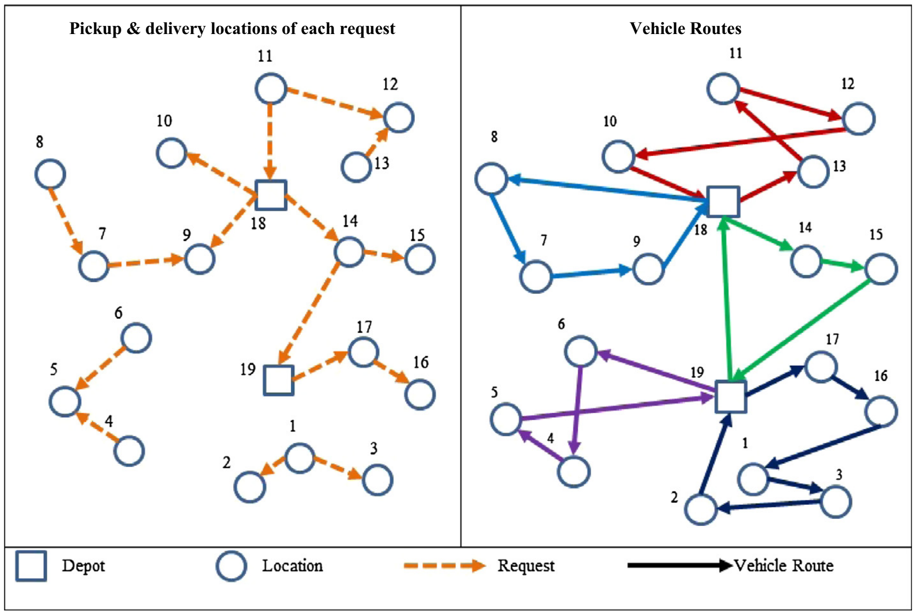


Fig. 1. The VRP with many-to-many requests.

g_k	Variable cost per distance unit of vehicle $k \in K$
H_i	Penalty cost if the request for pick up at node i is not served, $i \in P$.
d_{ij}, t_{ij}	Distance and travel time between node i and node j, for i and $j \in N$. Travel times satisfy the triangle inequality; $t_{ij} \leqslant t_{il} + t_{lj}$ for all $i, j, l \in V$; and are nonnegative

Since a visit to a node may be restricted to certain time interval and the service cannot occur instantaneously, the time window with the fixed and variable service times are included in the model

s_i	Fixed service time when visiting node i
e_i	Variable service time per item units of node i
$[a_i, b_i]$	Time window in which the visit at the particular location must start; a visit to node i can only take place between time a_i and b_i
l_i	Quantity of goods to be loaded onto the vehicle at node i for $i \in P$
$-u_j$	Quantity of goods to be unloaded from the vehicle at node j for $j \in D$

Decision variables

x_{ijk}	A binary variable with the value 1 if vehicle k travels from node i to node j and is zero otherwise, where $i, j \in V, k \in K$
S_{ik}	A nonnegative time that indicates when vehicle k starts the service at location $i, i \in V, k \in K$
L_{ik}	A nonnegative quantity that is an upper bound on the amount of goods on vehicle k arriving at node i, where $i \in V, k \in K$. S_{ik} and L_{ik} are defined only when vehicle k actually visits node i
z_i	A binary variable that indicates if a pickup request at node i, where $i \in P$, is placed on the request bank. The value is one if the request is placed in the request bank and zero otherwise

The detailed mathematical model can now be described. The objective is to minimize the weighted sum of the distance traveled, the sum of the fix cost for each used vehicle, and the penalty cost associated with number of requests that were not scheduled as given in Eq. (1). The constraints are given in Eqs. (2)–(16).

$$\text{Minimize} \quad \alpha \sum_{k \in K} g_k \sum_{(i,j) \in A} d_{ij} x_{ijk} + \beta \sum_{k \in K} \sum_{j \in P} f_k x_{\tau_k, j, k} + \gamma \sum_{i \in P} H_i z_i \tag{1}$$

$$\text{Subject to}: \sum_{k \in K} \sum_{j \in N_k} x_{ijk} - z_i \leqslant 1 \quad \forall i \in P \tag{2}$$

$$\sum_{j \in V} x_{ijk} - \sum_{j \in V} x_{j, j+r, k} = 0 \quad \forall k \in K, \forall i \in P \tag{3}$$

$$\sum_{j \in P \cup \{\tau'_k\}} x_{\tau_k, j, k} = 1 \quad \forall k \in K \tag{4}$$

$$\sum_{i \in D \cup \{\tau_k\}} x_{i, \tau'_k, k} = 1 \quad \forall k \in K \tag{5}$$

$$\sum_{i \in V} x_{ijk} - \sum_{i \in V} x_{j,i,k} = 0 \quad \forall k \in K, \forall j \in N \tag{6}$$

$$x_{ijk} = 1 \Rightarrow S_{ik} + s_i + l_i \times e_i + t_{ij} \leqslant S_{jk} \quad \forall k \in K, \forall (i,j) \in A \tag{7}$$

$$a_i \leqslant S_{ik} \leqslant b_i \quad \forall k \in K, \forall i \in V \tag{8}$$

$$S_{ik} \leqslant S_{i+r,k} \quad \forall k \in K, \forall i \in P \tag{9}$$

$$x_{ijk} = 1 \Rightarrow L_{ik} - u_i + l_i = L_{jk} \quad \forall k \in K, \forall (i,j) \in A \tag{10}$$

$$L_{ik} \leqslant C_k \quad \forall k \in K, \forall i \in V \tag{11}$$

$$L_{\tau_k k} = L_{\tau'_k k} = 0 \quad \forall k \in K \tag{12}$$

$$x_{ijk} \in \{0, 1\} \quad \forall k \in K, \forall (i,j) \in A \tag{13}$$

$$z_i \in \{0, 1\} \quad \forall i \in P \tag{14}$$

$$S_{ik} \geqslant 0 \quad \forall k \in K, \forall i \in V \tag{15}$$

$$L_{ik} \geqslant 0 \quad \forall k \in K, \forall i \in V \tag{16}$$

In the objective function, Eq. (1), α, β, and γ are the weights that reflect the relative importance of each cost component. Eq. (2) ensures that each pickup location is visited or that the corresponding request is placed in the request bank. Eq. (3) ensures that the delivery location is visited if the pickup location is visited and that the visit is performed by the same vehicle. Eqs. (4) and (5) ensure that a vehicle leaves every start terminal and a vehicle enters every end terminal. Together with Eq. (6) this ensures that consecutive paths between τ_k and τ'_k are formed for each vehicle $k \in K$. Eqs. (7) and (8) ensure that S_{ik} is set correctly along the paths and that the time windows are obeyed. These constraints also make sub tours impossible. Eq. (9) ensures that each pickup occurs before the corresponding delivery. Eqs. (10)–(12) ensure that the load variables are set correctly along the paths and that the capacity constraints of the vehicles are enforced.

This study focuses on the case when one location has many associated items that must be shipped to several different locations. The pickup and delivery nodes of each request in the mathematical model are modeled separately and the constraints are not in integer-linear form. This makes it much easier to understand the problem description and the formulation. Since the problem will be solved via metaheuristic, the integer-linear form of the model is not required. Moreover, the model did not require the picked up items to be shipped back to the depot and can support many variants of VRP such as VRPTW, CVRP, HVRP, VRPPD, and PDPTW.

4. Solution representations and decoding procedure for GVRP-MDMPDR

Solution representation and decoding procedure are the key elements in the design of effective particle swarm optimization algorithm. Sombuntham and Kachitvichyanukul (2010) presented a PSO-based algorithm with solution representation SD1 for solving GVRP-MDMPDR problem using a variant of PSO with multiple social learning terms, GLNPSO by Pongchairerks and Kachitvichyanukul (2009). The solution representation SD1 ignores the information of the origin–destination pairs of the requests and only uses the customer priority list and vehicle reference coordinates to construct the solutions. They reported that the algorithm provided consistent solution quality. However, the route construction procedure used in SD1 is quite complex with excessively high computational time. To improve the solution quality and solution time, two new solution representations, SD2 and SD3, are proposed here along with their corresponding decoding procedures.

5. Solution representation SD2 and decoding procedure

The solution representation SD2 utilizes the position of a particle to generate a priority list of pickup and delivery locations from the requests along with vehicle assignment. Suppose that r represents the total number of requests, the total number of dimensions of each particle is $3r$ *(2r for the pick-up and delivery locations and 1r for the vehicle assignment)*. Each dimension of the particle is filled with a randomly generated number. The ranking of the random number is used to prioritize the requests. The first $2r$ dimensions indicate the priorities of pickup and delivery operations of each request. The next r dimensions are used to find the assigned vehicle number for each request. The schematic representation of the particle is explained in Fig. 2 below.

The decoding procedure is performed in 3 steps: (1) construction of pickup–delivery priority list, (2) assignment of vehicle, and (3) construction of vehicle route. Figs. 3–5 illustrate the decoding procedure SD2 for the problem with 5 requests and 3 vehicles. As shown in Fig. 3, the first step of the decoding procedure is to construct a priority list for pickup and delivery by first extracting

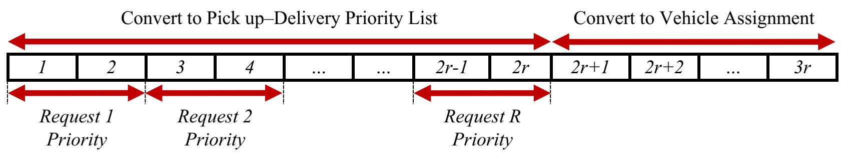
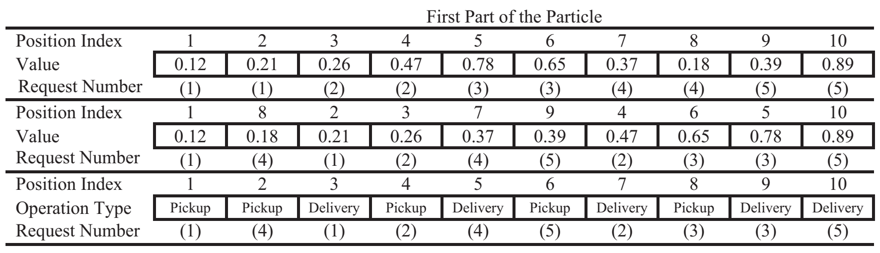


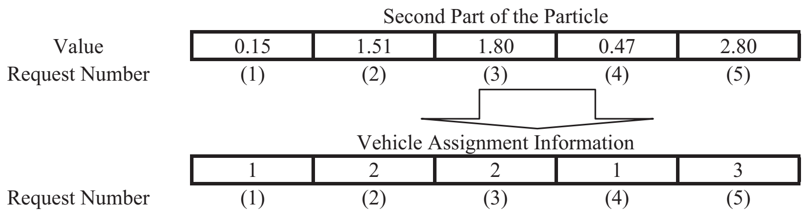
Fig. 2. Illustration of SD2 Solution representation.



First Part of the Particle

Position Index	1	2	3	4	5	6	7	8	9	10
Value	0.12	0.21	0.26	0.47	0.78	0.65	0.37	0.18	0.39	0.89
Request Number	(1)	(1)	(2)	(2)	(3)	(3)	(4)	(4)	(5)	(5)
Position Index	1	8	2	3	7	9	4	6	5	10
Value	0.12	0.18	0.21	0.26	0.37	0.39	0.47	0.65	0.78	0.89
Request Number	(1)	(4)	(1)	(2)	(4)	(5)	(2)	(3)	(3)	(5)
Position Index	1	2	3	4	5	6	7	8	9	10
Operation Type	Pickup	Pickup	Delivery	Pickup	Delivery	Pickup	Delivery	Pickup	Delivery	Delivery
Request Number	(1)	(4)	(1)	(2)	(4)	(5)	(2)	(3)	(3)	(5)

Fig. 3. Illustration of the construction of a Pickup–Delivery Priority List by SD2.



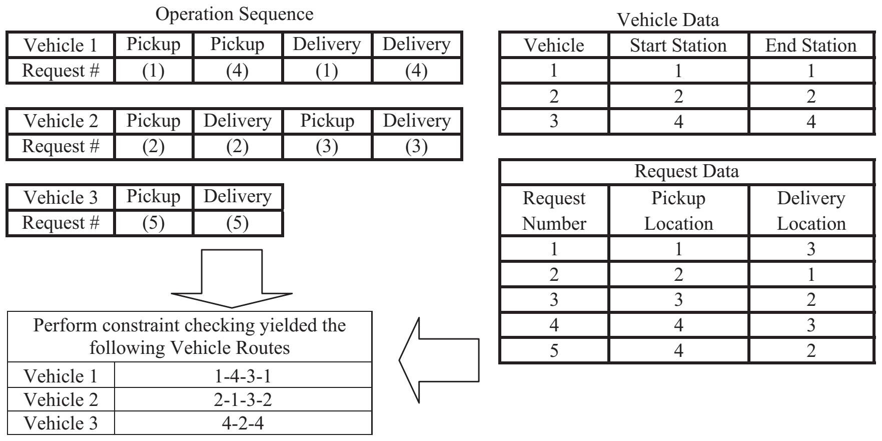
Second Part of the Particle

Value	0.15	1.51	1.80	0.47	2.80
Request Number	(1)	(2)	(3)	(4)	(5)

Vehicle Assignment Information

	1	2	2	1	3
Request Number	(1)	(2)	(3)	(4)	(5)

Fig. 4. Illustration of vehicle assignment by SD2.



Operation Sequence

Vehicle 1	Pickup	Pickup	Delivery	Delivery
Request #	(1)	(4)	(1)	(4)

Vehicle 2	Pickup	Delivery	Pickup	Delivery
Request #	(2)	(2)	(3)	(3)

Vehicle 3	Pickup	Delivery
Request #	(5)	(5)

Vehicle Data

Vehicle	Start Station	End Station
1	1	1
2	2	2
3	4	4

Request Data

Request Number	Pickup Location	Delivery Location
1	1	3
2	2	1
3	3	2
4	4	3
5	4	2

Perform constraint checking yielded the following Vehicle Routes

Vehicle 1	1-4-3-1
Vehicle 2	2-1-3-2
Vehicle 3	4-2-4

Fig. 5. Illustration of vehicle route construction by SD2.

all $2r$ values from a particle and rearrange the list in ascending order. All values in the list is then assigned a corresponding request number. Each request number must be presented twice in the list. The first occurrence is for pickup operation while the latter occurrence is for the delivery operation. For the second step, the vehicle assignment procedure is illustrated in Fig. 4 by converting the value in the second part of the particle from the $(2r+1)$th dimension to the $(3r)$th dimension into vehicle assignment for each request. As shown in Fig. 4, the value in each dimension is rounded up and the number indicates which vehicle number is being assigned while the position of such value represents its corresponding request number. The last step is the construction of the operation sequences and vehicle route as illustrated in Fig. 5 by using the data from Figs. 3 and 4. Each operation sequence of a request, pickup or delivery, is assigned to a vehicle based on vehicle assignment one by one. The problem constraints such as time window, vehicle capacity, and maximum route time are checked in this step. This step may generate routes that are different from the original vehicle assignment but the route construction procedure is deterministic and for the same set of data, the route generated by the decoding method is unique.

A formal algorithm steps for transforming the known data into the appropriate form for computing are explained in Algorithm 1. Although the assignment process is completed in step 3d of Algorithm 1, there might still be some unassigned requests and they must be added into the vehicle routes by using the feasibility improvement procedures named ARR1 and ARR2 in step 3e. ARR1 and ARR2 are discussed in later section.

Algorithm 1. Decoding Algorithm for SD2

The notation used in Algorithm 1 is given below.

Notations

θ_{lh}	Position of l^{th} particle at the h^{th} dimension
N	Set of locations, N_i is the location index with i^{th} priority. $\{N_1, N_2, \ldots, N_n\}$
L	Set of all requests, $\{L_1, L_2, \ldots, L_r\}$
W	Vehicle assignment list, where W_{ij} is vehicle index with j^{th} priority corresponding to location priority i^{th}. $\{W_{11}, W_{12}, \ldots, W_{nm}\}$
U_p	Set of unfulfilled requests which has $p$as a pickup location, $p = N_i$ where $N_i \in N$.
D_l	The delivery location of request$l$ where $l \in L$.
R_c	Route of vehicle $c, c = 1, 2, \ldots, m$.
M_c	Set of requests assigned to vehicle $c, c = 1, 2, \ldots, m$.
O_{lj}	Operation sequence (pickup–delivery sequences) of j^{th} vehicle corresponding to l^{th} particle

For a problem with m vehicles, and r requests, decoding Particle Position θ_{lh} (position of l^{th} particle at the h^{th} dimension) into Vehicle Operation sequences, O_{lj}, and Route, R_{lj}. (operation sequence and route of j^{th} vehicle corresponding to l^{th} particle). The route of vehicle is the sequence of nodes to visit while the operation sequence consists of pick-up and delivery of requests for each of the node visited.

```
1. Constructing Pickup–Delivery Priority List $(N)$
   a. Build set $S = \{1, 2, 3, \ldots, 2r - 1, 2r\}$ and $N = \emptyset$.
   b. Select $c$ from set $S$ where $\theta_{lc} = \min_{h \in S} \theta_{lh}$.
   c. Add $c$ to last position in set $N$.
   d. Remove $c$ from set $S$.
   e. Repeat step 1.b until $S = \emptyset$.
   f. For each $c$ in set $N$, set $c = \lceil \frac{c}{2} \rceil$.
   g. Set $i = 1$
      i. Start from earlier position of $N$ and set $m = k = 0$.
      ii. Consider $c_j$ (value at $m^{th}$ position of $N$).
      iii. If $c_j = i$, set $k = k + 1$.
      iv. If $k = 2$, set $c_j = c_j + r, i = i + 1$ and repeat from 1.g.i until
          $i = r + 1$.
          Otherwise, set $m = m + 1$ and repeat step1.g.ii.
2. Construct vehicle assignment list $(W)$
   a. Set $i = 1$, $W = \emptyset$.
   b. Set $c = \lceil \frac{\theta_{l,2n+i}}{2} \rceil$ (rounding up the value)
   c. Add $c$ to the last position in set $W$, and $i = i + 1$.
   d. Repeat 2.b until $i = r + 1$.
3. Construct vehicle route
   a. Set $k = 1$.
   b. Add operation, pickup or delivery, one by one to the
      operation sequence of a vehicle $W_k$.
      i. Set $c = N_k$ and $b = W_k$.
      ii. If $N_k \leqslant r$, make a candidate operation sequence of vehicle
          by inserting $c$ and $c + r$ into the last positions of $O_{lb}$.
      iii. Sort $O_{lb}$ based on the position value in $N$.
      iv. Convert $O_{lb}$ into $R_{lb}$.
      v. Check feasibility of the candidate route by evaluating all
         constraints: vehicle capacity, location time window,
         maximum, and route time restriction constraints.
      vi. If a feasible solution is found, keep the route $R_{lb}$ and $O_{lb}$
          and add $c$ to $M_c$. Otherwise remove $c$ and $c + r$ from $O_{lb}$.
   c. If $k = r$, go to step 3.d. Otherwise set $k = k + 1$, and go to
      step 3.b.
   d. Improve routes by eliminate repeating visited location in
      each route, for each vehicle $j = 1, 2, \ldots, m$.
      i. Set $S = \emptyset$.
      iii. Identify locations which the vehicle visits more than
           once, and add locations number to $S$.
      ii. For each $c$ in $S$, and for each repeating position of $c$ in $R_{lj}$,
          remove one position of location $c$ from $R_{lj}$ and move all
          operations which take place at the removed position to
          another position which is also location $c$ and still exists
          in $R_{lj}$. If it is feasible and total distance is reduced, save
          $R_{lj}$ and update $O_{lj}$.
   e. Add any remaining requests by using either procedures
      ARR1 or ARR2 (to be discussed later).
```

6. Solution representation SD3 and decoding procedure

The solution representation SD3 is different from SD2 mainly in the vehicle assignment procedure. In SD3, vehicle orientation points with radius coverage are used to generate vehicle assignment. The solution representation for SD3 is shown in Fig. 6. Note that the first part of the solution representation is the same as SD2 while the second part now consists of 3m dimension to represent (x, y) coordinate of the orientation point and the coverage radius. The concept is similar to the solution representation SR2 for VRP proposed by Ai and Kachitvichyanukul (2009b) where route construction is performed by using the information regarding vehicle orientation points and their coverage. The first part of the decoding procedure for SD3 is the same as that of SD2 that transforms the first 2r dimension of particle into the operation priority of requests at the first step as shown in Fig. 6 when r refers to number of requests and m represents number of vehicle.

The next step is to convert the next 3m dimension into the vehicle assignment by extracting the coordinates of vehicle orientation points including the vehicle coverage radius as depicted in Fig. 7. Such coordinates and radius are used for defining a coverage area of each vehicle when the coordinates are considered as the center. The virtual location of a request is defined as the middle point (Cartesian coordinates in two dimensions) between pickup and delivery locations of each request. Moreover, the "shipment point" is defined as either pickup or delivery location of the request. Once the middle point and shipment points of the request are within the coverage area of a vehicle, the request will be assigned to that vehicle. The explanation of a request's middle point and shipment points is shown in Fig. 8 and the example of middle points calculation is presented in Fig. 9.

Next step is the construction of vehicle routes based on the information obtained from previous section. Vehicle routes are constructed by considering a pickup operation of a request from the pickup–delivery list one by one. A candidate vehicle of a request must be the vehicle that its coverage area covers the middle point or at least one of shipment points. Each candidate is then assigned a numerical score which can be calculated based on 2 different methods depend on the situation of the vehicle. The first method is for the situation that the pickup location is the start station of the vehicle or the delivery location is the end station of the vehicle. The score for this situation is half of the distance between the other shipment point and the vehicle orientation point. Otherwise, the score is the Euclidean distance between the middle point of a request and the vehicle orientation point. The request is then assigned to a candidate vehicle based on ascending order of the scores as shown in Fig. 10.

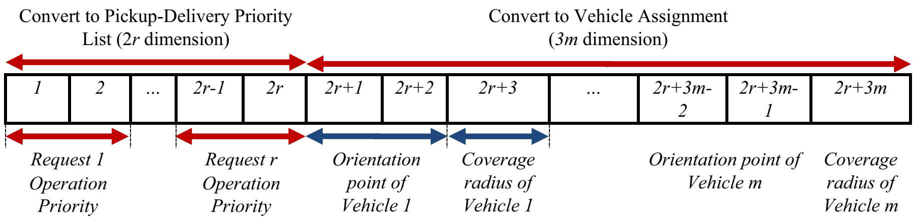

Convert to Pickup-Delivery Priority List (2r dimension)
Convert to Vehicle Assignment (3m dimension)
1
2
...
2r-1
2r
2r+1
2r+2
2r+3
...
2r+3m-2
2r+3m-1
2r+3m
Request 1 Operation Priority
Request r Operation Priority
Orientation point of Vehicle 1
Coverage radius of Vehicle 1
Orientation point of Vehicle m
Coverage radius of Vehicle m

Fig. 6. Illustration of SD3 solution representation.

Particle (Second part)

(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)
40	40	24	60	80	25	80	30	32

Vehicle Number	Orientation Points	Coverage Radius
1	(40, 40)	24
2	(60, 80)	25
3	(80, 30)	32

Fig. 7. Extracting the second part of a particle into vehicle orientation points and coverage radius.

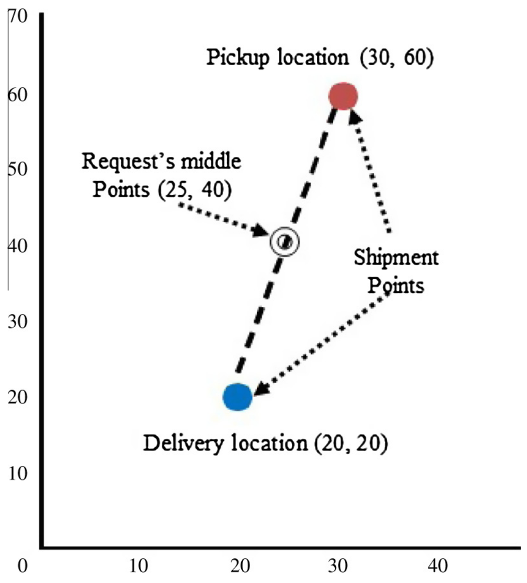


Fig. 8. Illustration of middle point and shipment points of a request.

Later, the vehicle operation sequence is converted into a vehicle route and the problem constraints are then checked for feasibility. If it is infeasible, the next candidate vehicle is considered. After that the assignment of the remaining requests is performed to reduce the number of unassigned requests. The route construction and the formal algorithm of SD3 are summarized in Fig. 11 and Algorithm 2, respectively.

Request Number	Score of Vehicle		
	1	2	3
1	0.5 x (28.28)	---	---
2	0.5 x (64.03)	0.5 x (22.37)	---
3	18.03	0.5 x (72.11)	---
4	25	---	0.5 x (60.83)
5	---	0.5 x (67.08)	0.5 x (60.00)

Request Number	Candidate Vehicles	
	1st Priority	2nd Priority
1	Vehicle 1	---
2	Vehicle 2	Vehicle 1
3	Vehicle 1	Vehicle 2
4	Vehicle 1	Vehicle 3
5	Vehicle 3	Vehicle 1

Fig. 10. Forming list of candidate vehicle of each request by using vehicle orientation points and coverage radius.

Algorithm 2. Decoding Algorithm for SD3.

The decoding step of SD3 for a problem with m vehicles, and r requests, decoding particle position θ_{lh} (position of l^{th} particle at the h^{th} dimension) into vehicle operation sequences O_{lj}, and Route R_{lj}(operation sequence and route of j^{th} vehicle corresponding to l^{th} particle) can be explained below.

1. Construct Pickup–Delivery Priority List (N)
   a. Build set $S = \{1, 2, 3, \ldots, 2r - 1, 2r\}$ and $N = \emptyset$.
   b. Select c from set S where $\theta_{lc} = \min_{h \in S} \theta_{lh}$.
   c. Add c to last position in set N.
   d. Remove c from set S.
   e. Repeat step 1.b until $S = \emptyset$.
   f. For each c in set N, set $c = \frac{c}{2}$.
   g. Set $i = 1$
      i. Start from earlier position of N and set $m = k = 0$.
      ii. Consider c_j (value at m^{th} position of N).
      iii. If $c_j = i$, set $k = k + 1$.
      iv. If $k = 2$, set $c_j = c_j + r, i = i + 1$ and repeat 1.g.i until $i = r + 1$. Otherwise, set $m = m + 1$ and repeat step1.g.ii.
2. Extract Vehicle Coverage Area information, for each vehicle $j = 1, 2 \ldots m$.

Location	X-Y coordinate
1	(30, 60)
2	(80, 90)
3	(20, 20)
4	(90, 20)

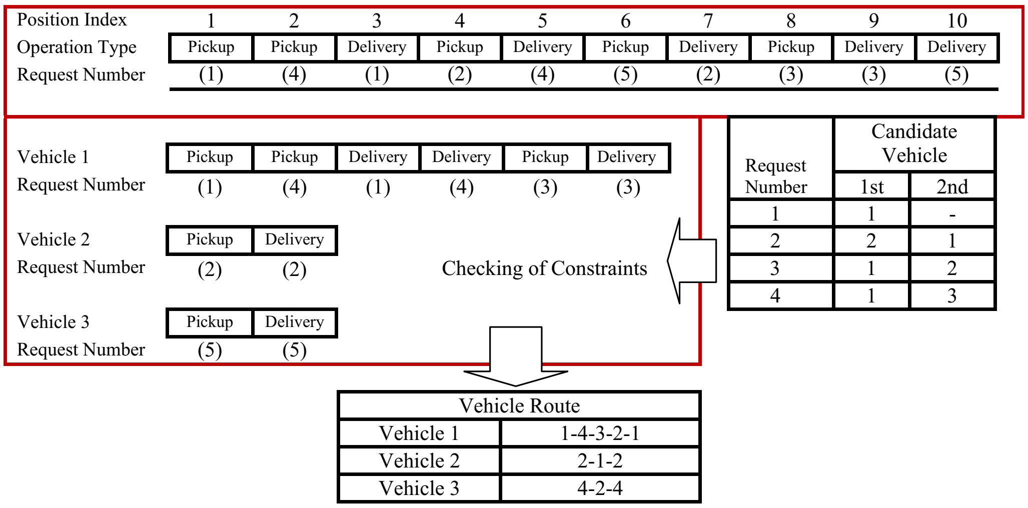
+

Request number	Pickup location	Delivery location
1	1	3
2	2	1
3	3	2
4	4	3
5	4	2

=

Request number	Middle Points
1	(25, 40)
2	(55, 75)
3	(50, 55)
4	(55, 20)
5	(85, 55)

Fig. 9. Calculation of middle point for each request.



Position Index	1	2	3	4	5	6	7	8	9	10
Operation Type	Pickup	Pickup	Delivery	Pickup	Delivery	Pickup	Delivery	Pickup	Delivery	Delivery
Request Number	(1)	(4)	(1)	(2)	(4)	(5)	(2)	(3)	(3)	(5)

Vehicle 1	Pickup	Pickup	Delivery	Delivery	Pickup	Delivery
Request Number	(1)	(4)	(1)	(4)	(3)	(3)
Vehicle 2	Pickup	Delivery				
Request Number	(2)	(2)				
Vehicle 3	Pickup	Delivery				
Request Number	(5)	(5)				

Request Number	Candidate Vehicle	
	1st	2nd
1	1	-
2	2	1
3	1	2
4	1	3

Vehicle Route	
Vehicle 1	1-4-3-2-1
Vehicle 2	2-1-2
Vehicle 3	4-2-4

Fig. 11. Route construction of SD3 based on the Pickup–Delivery Priority List and candidate vehicles list.

a. Set the coordinate $(xref_j, yref_j)$ of the orientation point, $xref_j = \theta_{l,2n+3j-2}$ and $yref_j = \theta_{l,2n+2j-1}$.
b. Set coverage radius, $g_j = \theta_{l,2n+3j}$.

3. Construct Candidates Vehicles, for each request i = 1, 2, ..., r. (W)
   a. Compute a request middle point,

$$repx_i = \frac{x_{P_i} + x_{D_i}}{2}, \tag{17}$$

$$repy_i = \frac{y_{P_i} + y_{D_i}}{2} \tag{18}$$

   ((x_{P_i}, y_{P_i}) and (x_{D_i}, y_{D_i}) are coordinates of pickup and delivery location of request i.)
   b. Build set S = {1, 2, ..., m}, and $W_i = \emptyset$.
   c. Assign score for each vehicle j = 1, 2, ..., m and set $score_{ij} = -1$.
      i. If pickup location is the same location as vehicle j start station,

$$score_{ij} = \sqrt{(x_{D_i} - xref_j)^2 + (y_{D_i} - yref_j)^2} \tag{19}$$

      ii. Else if delivery location is the same location as vehicle j end station,

$$score_{ij} = \sqrt{(x_{P_i} - xref_j)^2 + (y_{P_i} - yref_j)^2} \tag{20}$$

      iii. Otherwise, if the middle point of request i is within vehicle j's coverage area,

$$score_{ij} = \sqrt{(repx_i - xref_j)^2 + (repy_i - yref_j)^2} \tag{21}$$

      iv. Remove j from S. If $score_{ij} \geqslant 0$, add j to W_i.
   d. Sort W_i based on the score in ascending order.
4. Construct Vehicle Route
   a. For each request k and set of candidate vehicle W_k for request k.
   b. Add operation, pickup or delivery, one by one to the operation sequence of a vehicle
      i. $c = N_k$ where c is the (pickup or delivery) operation of request k.
      ii. $b = W_{kj}$ (b *is the candidate vehicle* j *of request* k *from set* W_k).
      iii. If $N_k \leqslant r$, make a candidate operation sequence of vehicle by inserting c and $c + r$ into the last positions of O_{lb}. Otherwise next request k, and go to step 4.b.
      iv. Sort O_{lb} based on the position value in N.
      v. Convert O_{lb} into R_{lb}.
      vi. Check feasibility of the candidate route by evaluating all constraints: vehicle capacity, location time window, maximum, and route time restriction constraints.
      vii. If a feasible solution is found, update the route R_{lb} and O_{lb} and add c to M_c. Otherwise remove c and $c + r$ from O_{lb} and remove vehicle b from W_k.
      viii. If $W_k = \emptyset$, set next request k, and go to step 4.b. Otherwise, set next vehicle j, and go to step 4.b.ii.
   c. If no more request, go to step 4.d. Otherwise set next request k, and go to step 4.b.
   d. Improve routes by eliminate repeating visited location in each route, for each vehicle j = 1, 2, ..., m.
      i. Set $S = \emptyset$.
      ii. Identify locations which the vehicle visits more than once, and add locations number to S.
      iii. For each c in S, and for each repeating position of c in R_{lj}, remove one position of location c from R_{lj} and move all operations which take place at the removed position to another position which is also location c and still exists in R_{lj}. If it is feasible and total distance is reduced, save R_{lj} and update O_{lj}.
   e. Add the remaining request by using procedures ARR1 or ARR2.

7. Feasibility improvement procedures ARR1 and ARR2

The proposed solution representations, SD2 and SD3 and the decoding methods are designed for solving VRPs using the GLNPSO algorithm. As the obtained results rely on random search which might lead to solutions with some unfulfilled requests. Therefore, a feasibility improvement procedure for determining the appropriate sequences and vehicles to assign the unfulfilled requests are required. This section describes 2 different feasibility improvement procedures called ARR1 and ARR2.

The basic idea of ARR1 is simply to insert a request into the route at the positions that lead to more number of unfulfilled requests assigned as a primary criterion with the least increase in distance assigned as a secondary criteria. For each of the unfulfilled requests, the distance between the pickup location and the nearest location exists in the route is used as the criteria for selecting of vehicle for insertion. After the admissible placement for pickup location is found, the procedure attempts to insert the associated delivery location to all possible positions of the vehicle route after the pickup location. This insertion idea that considers the pickup location first and follows by the delivery location is similar to the single pair insertion, SPI, by Nanry and Barnes (2000) and it is included in Algorithm 3.

The second feasibility improvement procedure, ARR2, is to insert pickup and delivery operations of unfulfilled requests into a vehicle operation sequence. For a given request, a vehicle is selected in the same way as ARR1 based on the shortest distance between the pickup location and existing location in the route. The pickup operation of the request is then inserted as the first operation sequence of the vehicle while the delivery operation is inserted into the last operation sequence. The better placement of pickup operation is determined by trying to move the placement to later positions. The best placement is found only when feasibility is met as well as the least increase in distance is found. For the placement for delivery operation, it can be determined by using the same idea as that of pickup operation but in reverse direction which is given in Algorithm 4.

Algorithm 3. Add Remaining Requests 1 (ARR1).

```
For each unfulfilled request $k$ in set $U$,

1. Evaluate all vehicles based on the distance between the pickup
   location of request $k$ and closest location existed in the route.
   Assign priority to the vehicle based on the distances starting
   from the first priority vehicle, $c$ = 1.
2. Set $R$ = the route of vehicle $c$, $Q$ = set of requests assigned to the
   vehicle $c$.
3. Start from first location existed in the route; insert the pickup
   location after it.
4. Set $R'$ = the route after insertion and $Q'$ = the request assignment list.
5. Determine possible request assignments based on the new
   candidate route $R'$ and add newly assigned requests to $Q'$ if
   feasibility is found.
6. For each position $l$ in $R'$ that follows the pickup location,
   a. Insert the delivery location of request $k$ after the position $l$ in
      $R'$.
   b. Add request $k$ to $Q'$ and check for feasibility.
   c. If feasible, determine possible request assignments based on
      the new candidate route $R'$ and add newly assigned requests
      to $Q'$ .
   d. If one of the following conditions is satisfied.
      • $\|Q'\| > \|Q\|$.
      • $\|Q'\| = \|Q\|$, and less total distance.

   Set $R = R'$ and $= Q'$.
7. Consider next location in route and repeat step 3 until all locations in the route is considered
8. If request $k \notin Q$, set $c = c + 1$ and repeat step 2. Otherwise,
   save $R$ and $Q$ to the route and request assignment of vehicle $c$,
   and update operation sequence of the vehicle and $U$.
```

Algorithm 4. Add Remaining Requests 2 (ARR2).

```
For each unfulfilled request $k$ in set $U$,

1. Evaluate all vehicles based on the distance between the pickup
   location of request $k$ and closest location ($B_k$) existed in the
   route., Assign priority to the vehicle based on the distances.
   Start from the first priority vehicle, $c$ = 1.
2. Set $O$ = the vehicle $c$'s operation sequence, $Q$ = set of requests
   assigned to the vehicle $c$.
3. Insert the pickup operation after the operation performed at $B_k$
   and delivery operation after the last operation sequence in $O$.
4. Set $O = O_b$ and $D_b$ = total distance if feasibility is met. Otherwise
   $O_b = Ø$ and $D_b$ = a very large value.
5. For each position $l$ later than the position of the pickup operation of request $k$
   a. Move the delivery operation of the position before position
      $l$.
   b. Set $O = O_b$ and $D_b$ = total distance if feasibility is met and
      new total distance $< D_b$.
6. If $O_b = Ø$, change position of pickup operation. Otherwise go to
   step 8.
7. If the new position of inserting results in less distance increment than the distance between the pickup location and the
   next closest vehicle, repeat step 5. Otherwise go to step 6.
8. Add $k$ to $Q$ and determine possible request assignments based
   on the new candidate route $R$ and add newly assigned requests
   to $Q$ if feasibility is found.
9. Save $O_b$, $R$ and $Q$ to operation sequence, request assignment, and
   route of vehicle $c$ then update $U$.
```

8. Computational experiments

The proposed algorithms are implemented using the ETLib object library from Nguyen, Ai, and Kachitvichyanukul (2010) using the class library from for GLNPSO, a variant of PSO with multiple social learning terms. According to the characteristics of the problem, the GVRP-MDMPDR model allows each location to serve more than one role, the test problem instances must contain the characteristics that each location can have multiple requests and each location can serve more than one role. Since the PDPs are the problems that each location can play only one of the three roles, pickup location, delivery location, and vehicle station, the PDP is only the special case of GVRP-MDMPDR where each location is playing only one role. Consequently, the test problem instances for PDPTW of Li and Lim (2001) can be used for initial test. Additional datasets that allow a location to play multiple roles are provided in Sombuntham and Kunnapapdeelert (2012). The algorithms are implemented with C# with Microsoft Visual Studio 2008. The program runs on the platform of Intel® Core TM, CPU 2.83 GHz with 2.96 GB RAM.

8.1. Pickup and delivery problem with time windows instances

The first set of benchmark problems tested is the 100-location PDPTW from Li and Lim (2001). The instances consist of three scenarios of location distribution, i.e. clustered locations, randomly distributed locations, and half-random-half-clustered locations. Each location can have multiple requests and can serve multiple roles such as vehicle station, pickup and delivery location.

The parameters are set as follows. Maximum number of iteration and number of particle are set as 1000 and 100, respectively. Constant values such as inertia weight and number of neighbors are set follow the work of Ai and Kachitvichyanukul (2009a). The number of neighbors is five and inertia weight is linearly decreasing from 0.9 to 0.4. Acceleration constants for global best, personal best, local best, and near neighbor best are set as 0.5, 0.5, 1.5, and 1.5, respectively. The GLNPSO algorithm with 3 solution representations, SD1, SD2, and SD3 are run 5 times for each test problem instance to ensure performance of the algorithm as presented in Tables 1a, 1b, and 1c where NV denotes number of vehicles required for the solution.

The results imply that the solution quality obtained from SD1 and SD3 are quite similar when applied for solving the problem

Table 1a
Results of PDPTW (Clustered locations).

Case	Best known solution		SD1		SD2		SD3	
			Average % deviation		Average % deviation		Average % deviation	
	NV	Distance	NV	Distance	NV	Distance	NV	Distance
lc101	10	828.94	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
lc102	10	828.94	**0.00**	**0.00**	**0.00**	**0.00**	0.00	0.73
lc103	9	1035.35	4.44	−5.59	11.11	−18.00	11.11	−19.37
lc104	9	860.01	**0.00**	**2.81**	0.00	14.83	0.00	8.42
lc105	10	828.94	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
lc106	10	828.94	2.00	7.43	0.00	0.29	**0.00**	**0.00**
lc107	10	828.94	**0.00**	**0.00**	0.00	0.32	**0.00**	**0.00**
lc108	10	826.44	2.00	1.57	0.00	0.46	**0.00**	**0.31**
lc109	9	1000.6	**11.11**	**−17.25**	11.11	−7.42	11.11	−16.55
lc201	3	591.56	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
lc202	3	591.56	**0.00**	**0.00**	13.33	3.82	**0.00**	**0.00**
lc203	3	585.56	**0.00**	**0.96**	0.00	4.62	0.00	1.02
lc204	3	590.6	0.00	4.34	0.00	9.02	**0.00**	**2.16**
lc205	3	588.88	0.00	0.26	0.00	0.04	**0.00**	**0.00**
lc206	3	588.49	**0.00**	**0.00**	0.00	2.01	**0.00**	**0.00**
lc207	3	588.29	**0.00**	**0.00**	0.00	1.97	**0.00**	**0.00**
lc208	3	588.32	**0.00**	**0.00**	0.00	0.97	**0.00**	**0.00**
Average			1.15	−0.32	2.09	0.76	1.31	−1.37

Bold number in the table indicates the minimum average percentage deviation in its row.

Table 1b
Results of PDPTW (Random locations).

Case	Best known solution		SD1		SD2		SD3	
			Average % deviation		Average % deviation		Average % deviation	
	NV	Distance	NV	Distance	NV	Distance	NV	Distance
lr101	19	1650.8	0.00	0.66	**0.00**	**0.00**	1.05	0.17
lr102	17	1487.57	0.00	4.85	**0.00**	**3.12**	0.00	3.37
lr103	13	1292.68	**0.00**	**5.21**	0.00	10.97	0.00	6.99
lr104	9	1013.39	15.56	9.33	13.33	11.13	15.56	9.14
lr105	14	1377.11	0.00	1.50	1.43	1.58	1.43	1.30
lr106	12	1252.62	3.33	3.24	13.33	12.50	**1.67**	**0.74**
lr107	10	1111.31	12.00	9.93	20.00	15.13	**10.00**	**8.26**
lr108	9	968.97	**2.22**	**1.30**	11.11	10.43	6.67	4.81
lr109	11	1208.96	16.36	11.42	23.64	17.99	**12.73**	**9.09**
lr110	10	1159.35	**16.00**	**7.20**	24.00	15.86	22.00	9.85
lr111	10	1108.9	10.00	7.54	14.00	9.46	**6.00**	**4.48**
lr112	9	1003.77	**22.22**	**18.13**	26.67	26.68	22.22	19.13
lr201	4	1253.23	0.00	1.79	10.00	4.70	**0.00**	**0.82**
lr202	3	1197.67	33.33	12.02	33.33	28.33	**20.00**	**10.48**
lr203	3	949.4	0.00	20.26	6.67	27.71	**0.00**	**16.95**
lr204	2	849.05	50.00	22.42	50.00	35.07	**20.00**	**7.85**
lr205	3	1054.02	26.67	23.40	33.33	28.48	**33.33**	**20.05**
lr206	3	931.63	0.00	17.48	13.33	45.29	**0.00**	**10.86**
lr207	2	903.06	90.00	31.25	50.00	50.12	**50.00**	**19.29**
lr208	2	734.85	20.00	21.27	50.00	36.63	**10.00**	**17.58**
lr209	3	930.59	20.00	16.92	33.33	30.86	**13.33**	**12.17**
lr210	3	964.22	33.33	36.69	20.00	52.76	**0.00**	37.52
lr211	2	911.52	50.00	13.37	50.00	32.84	**50.00**	**10.97**
Average			18.31	12.92	21.63	22.07	12.87	10.52

Bold number in the table indicates the minimum average percentage deviation in its row.

that the locations are clustered. The SD3 is more robust than SD1 and SD2 for the cases with random and half-random-half-clustered distributed with large maximum route time. However, SD1 is more effective than the others for the cases with half-random-half-clustered location with short planning.

As mentioned earlier, the final solutions based on SD1, SD2, and SD3 might still have some unfulfilled requests in the solutions. The improvement procedures, ARR1 and ARR2, are added to improve the solutions by reducing the unfulfilled requests. A preliminary experiment is first carried out to assess the effectiveness of ARR1 and ARR2 by using three different versions of SD2, i.e., SD2 without ARRs, SD2 with ARR1, and SD2 with ARR2. The test problems from Dataset A from Sombuntham and Kunnapapdeelert (2012) are used. The experiments are performed using the accelerated constants for global best, personal best, local best, and near neighbor best of 0, 1, 1, and 2 with 10 replications for each problem instance and the results are presented in Table 2.

Table 2 illustrates that the ARR1 provide the better solutions than that of ARR2. However, ARR1 consumes more computational time than the ARR2. Average percentage deviation of objective function from SD2 using ARR2 from those of ARR1 including the percentage of time improved from ARR1 when use ARR2 are computed to compare potential of improvement between these two

Table 1c
Results of PDPTW (Mixed cluster and random locations).

Case	Best known solution		SD1		SD2		SD3	
			Average % deviation		Average % deviation		Average % deviation	
	NV	Distance	NV	Distance	NV	Distance	NV	Distance
lrc101	14	1708.8	**0.00**	**0.64**	2.86	1.05	0.00	1.07
lrc102	12	1558.07	**3.33**	**2.11**	11.67	5.16	6.67	4.10
lrc103	11	1258.74	**0.00**	**3.17**	10.91	12.66	5.45	6.82
lrc104	10	1128.4	**2.00**	**4.84**	8.00	10.80	6.00	8.30
lrc105	13	1637.62	**3.08**	**1.31**	6.15	2.97	6.15	2.36
lrc106	11	1424.73	12.73	4.65	20.00	12.04	**10.91**	**5.31**
lrc107	11	1230.15	**0.00**	**4.76**	10.91	14.92	7.27	6.37
lrc108	10	1147.43	**10.00**	**3.27**	12.00	17.11	12.00	12.22
lrc201	4	1406.94	20.00	27.80	20.00	40.73	**15.00**	**31.54**
lrc202	3	1374.27	**20.00**	**10.03**	53.33	21.85	33.33	10.50
lrc203	3	1089.07	26.67	8.01	26.67	14.65	**0.00**	**0.71**
lrc204	3	818.66	**0.00**	**0.89**	0.00	18.31	0.00	2.98
lrc205	4	1302.2	15.00	28.41	20.00	39.58	**5.00**	**33.64**
lrc206	3	1159.03	**26.67**	**18.39**	53.33	39.38	33.33	24.86
lrc207	3	1062.05	33.33	37.96	33.33	47.10	**20.00**	**30.03**
lrc208	3	852.76	13.33	14.64	26.67	33.68	**0.00**	**5.93**
Average			11.63	10.68	19.74	20.75	10.07	11.67

Bold number in the table indicates the minimum average percentage deviation in its row.

Table 2
Comparison of feasibility improvement procedure using SD2.

Instance	SD2 without ARRs			SD2 with ARR1			SD2 with ARR2		
	Objective function		Time (s)	Objective function		Time (s)	Objective function		Time (s)
	Average	SD		Average	SD		Average	SD	
Aac1	1982.33	209.51	8.27	891.66	13	32.88	925.55	36.64	22.32
Aac2	2203.77	172.57	8.4	802.92	29.59	39.61	885.26	51.28	21.31
Aar1	3531.79	147.82	8.41	2102.92	80.53	41.81	2209.92	113.84	22.95
Aar2	[a]	[a]	8.08	2377.03	110.3	36.72	2502.89	81.05	27.61
Aarc1	2895.66	204.27	8.44	1554.12	74.54	42.88	1622.45	68.38	22.58
Aarc2	3079.77	188.85	8.37	1726.28	36.08	39	1801.59	58.96	23.93

[a] The value is large due to remained unfulfilled requests.

Table 3
Deviation of the results of ARR2 from ARR1.

Instances	Deviation of ARR2 from ARR1	
	% Average deviation of objective function	% Average time deviation
Aac1	3.80	−32.12
Aac2	10.25	−46.20
Aar1	5.09	−45.11
Aar2	5.29	−24.81
Aarc1	4.40	−47.34
Aarc2	4.36	−38.64

procedures as illustrated in Table 3. The results showed that ARR2 is preferable than the ARR1.

8.2. Experimental results on test problems for GVRP-MDMPDR

The performances of the two solution representations are further evaluated in this section by using test problem instances provided in Sombuntham and Kunnapapdeelert (2012). All of PSO parameters are the same as those used in previous experiment except that the number of iteration and number of particles are

Table 4
Results of the proposed algorithm on data set A.

Instance	Best solution so far	Objective function									Average computational time (s)		
		SD1			SD2 with ARR2			SD3 with ARR2					
		AVG	SD	%Dev	AVG	SD	%Dev	AVG	SD	%Dev	SD1	SD2	SD3
Aac1	864.22	887.59	0.00	0.03	928.23	78.90	0.07	886.10	16.87	**0.03**	244.82*	39.87	44.35
Aac2	782.54	865.75	6.27	0.11	906.28	63.66	0.16	795.05	13.00	**0.02**	263.47*	41.90	47.32
Aar1	2004.06	2244.04	73.31	0.12	2194.54	118.72	0.10	2043.59	28.57	**0.02**	246.46*	44.03	56.45
Aar2	2373.18	2513.50	4.39	0.06	2585.79	112.11	0.09	2434.54	39.40	**0.03**	220.68*	46.31	63.41
Aarc1	1436.74	1464.19	21.60	**0.02**	1603.38	92.98	0.12	1560.72	72.73	0.09	348.42*	41.17	55.36
Aarc2	1679.34	1697.77	13.91	**0.01**	1814.67	105.11	0.08	1741.77	34.79	0.04	284.98*	41.68	55.33

A bold number indicates the minimum average deviation of the entire row.
A number with asterisk (*) indicates that it takes the greatest computational time compare to other algorithms.

Table 5
Results of the SD2 and SD3 on data set B, C, D, and E.

Instance	Best solution so far	Objective function						Time (s)		%
		SD2			SD3					
		AVG	SD	% Dev	AVG	SD	% Dev	SD2	SD3	
Bac1	1805.82	2147.19	204.11	0.19	1856.58	36.75	**0.03**	128	155	22
Bac2	1897.47	2305.76	195.91	0.22	2007.80	67.45	**0.06**	120	150	25
Bar1	4196.45	4506.90	281.53	0.07	4274.32	85.19	**0.02**	127	193	52
Bar2	7848.43	8606.65	213.15	0.10	8314.79	300.90	**0.06**	403	546	35
Barc1	2838.10	3316.33	182.87	0.17	2942.79	83.52	**0.04**	128	174	36
Barc2	3123.40	3405.79	131.80	0.09	3200.26	65.24	**0.02**	127	187	47
Cac1	3379.24	3941.54	106.81	**0.04**	3950.34	124.20	**0.04**	386	400	3
Cac2	4297.98	4791.03	97.37	0.05	4705.29	132.25	**0.04**	368	400	9
Car1	6153.78	7023.89	113.00	0.05	6856.13	126.68	**0.02**	405	475	17
Car2	7643.92	8645.09	170.41	0.04	8461.05	212.92	**0.02**	420	530	26
Carc1	4406.78	4972.85	85.82	0.08	4964.94	263.89	**0.08**	364	427	17
Carc2	4320.85	5225.43	156.91	0.04	5125.09	110.68	**0.02**	362	474	31
Dc1	2715.44	3307.91	632.43	0.22	2914.33	214.21	**0.07**	419	490	17
Dc2	2670.12	3220.42	270.24	0.21	2754.97	61.23	**0.03**	398	488	23
Dr1	3424.96	3727.13	183.92	0.09	3539.90	152.87	**0.03**	483	527	9
Dr2	3484.56	3929.62	244.38	0.13	3805.01	299.78	**0.09**	479	607	27
Drc1	4388.53	4648.49	217.15	0.06	4636.59	209.36	**0.06**	497	600	21
Drc2	4865.68	5185.40	222.91	**0.07**	5405.75	181.78	0.11	509	642	26
Ec1	3100.78	4275.60	428.44	0.38	3653.50	378.50	**0.18**	412	471	15
Ec2	3165.58	3878.82	132.48	0.23	3337.96	196.99	**0.05**	384	434	13
Er1	3404.02	3979.25	446.22	0.17	3658.98	113.50	**0.07**	469	563	20
Er2	3260.52	4269.31	392.73	0.31	3593.54	224.56	**0.10**	465	619	33
Erc1	4157.00	5057.35	232.08	0.22	4245.51	137.43	**0.02**	412	493	20
Erc2	6882.52	7812.98	326.96	0.14	7062.69	175.38	**0.03**	426	570	34

set as 1000 and 50, respectively. In this experiment, 5 replications are applied on dataset A as presented in Table 4.

The results depict that SD3 provides the best quality of solution when the locations are clustered and randomly distributed whereas SD1 provides the best solution quality for the remaining instances. Considering the computational effort, it is clear that SD1 requires much longer computational time than SD3 and SD2, respectively. Moreover, computing time of SD1 is at least 3 times longer than those of SD2 and SD3. This demonstrated that SD2 and SD3 are preferable to SD1. The proposed algorithms, SD2 and SD3, are further tested with the remaining test problem instances, namely B, C, D, and E, and the results are shown in Table 5.

The result in Table 5 illustrates that SD3 provides the better solution quality than that of SD2 in most cases. Only 2 case instances, Cac1 and Drc2, are found that SD2 have better performance than SD3. The computational times of SD3 are slightly higher than those of SD2 for all cases.

9. Conclusions

Two solution representations, SD2 and SD3, and their associated decoding methods for solving multi depot vehicle routing problem with pickup and delivery request are proposed in this paper. The two solution representations and decoding methods were implemented using GLNPSO which is the variant of PSO with multiple social learning terms. The proposed algorithms were first evaluated by using test problem instances for PDPTW. The results showed that in most cases SD3 provided better solutions for PDPTW than those from SD1 and SD2, respectively.

Two feasibility improvement procedures, ARR1 and ARR2, are also proposed to reduce the number of unfulfilled requests that may remain in the solution. The experimental results indicate that the ARR2 is more appropriate to deal with such problem than the ARR1. The feasibility improvement procedure ARR2 is then added to both SD2 and SD3 and the experiments are further conducted for solving GVRP-MDMPDR by using test problem dataset from Sombuntham and Kunnapapdeelert (2012). The results confirmed that the PSO framework with solution representation, SD3, and feasibility improvement procedure, ARR2, is the best among the algorithms tested for solving variant of VRPs such as PDPTW and GVPR-MDMPDR.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments. The second author acknowledge the financial support of the Asian Institute of Technology during his study period at the Institute prior to his current affiliation. This work was supported by Thailand Research Fund under Grant MLSC525009. The development of the object library for evolutionary techniques (ETLib) received financial support from the 2008 Royal Thai Government Joint Research Program for Visiting Scholars.

References

Ai, T. J., & Kachitvichyanukul, V. (2009a). A particle swarm optimization for vehicle routing problem with time windows. *International Journal of Operational Research, 6*(4), 519–537.
Ai, T. J., & Kachitvichyanukul, V. (2009b). Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem. *Computers & Industrial Engineering, 56*(1), 380–387.
Ai, T. J., & Kachitvichyanukul, V. (2009c). A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research, 36*, 1693–1702.
Berbeglia, G., Cordeau, J. -F., & Laporte, G. (2009). Dynamic pickup and delivery problems. *European Journal of Operational Research*, in press, Corrected Proof (http://dx.doi.org/10.1016/j.ejor.2009.04.024).
Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science, 6*(1), 80–91.
Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE international conference on neural networks* (Vol. 4, pp. 1942–1948).
Li, H., & Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. In *Proceedings of the 13th international conference on tools with artificial intelligence, ICTAI-2001, Dallas, USA* (pp.160–170).

Nanry, W. P., & Barnes, J. W. (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research B, 34*, 107–121.

Nguyen, S., Ai, T. J., & Kachitvichyanukul, V. (2010). *Object library for evolutionary techniques ETLib: User's guide*. Thailand: Asian Institute of Technology.

Pongchairerks, P., & Kachitvichyanukul, V. (2009). Particle swarm optimization algorithm with multiple social learning structures. *International Journal Operational Research, 6*(2), 176–194.

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery with time windows. *Transportation Science, 40*, 455–472.

Sombuntham, P., & Kunnapapdeelert, S. (2012). Benchmark problem instances for generalized multi-depot vehicle routing problems with pickup and delivery requests. In *Proceedings of the 13th Asia pacific industrial engineering* & *management systems conference 2012, Thailand* (pp. 290–297).

Sombuntham, P., & Kachitvichyanukul, V. (2010). Multi-depot vehicle routing problem with pickup and delivery requests. *IAENG Transactions on Engineering Technologies, 5*, 71–85.