Computers & Industrial Engineering 56 (2009) 380–387

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem

The Jin Ai *, Voratas Kachitvichyanukul

Industrial Engineering and Management, School of Engineering and Technology, Asian Institute of Technology, P.O. Box 4, Klong Luang, Pathumtani 12120, Thailand

ARTICLE INFO

Article history:
Received 5 October 2007
Received in revised form 24 April 2008
Accepted 12 June 2008
Available online 20 June 2008

Keywords:
Capacitated vehicle routing problem
Particle swarm optimization
Solution representation
Metaheuristics

ABSTRACT

This paper presents two solution representations and the corresponding decoding methods for solving the capacitated vehicle routing problem (CVRP) using particle swarm optimization (PSO). The first solution representation (SR-1) is a $(n+2m)$-dimensional particle for CVRP with n customers and m vehicles. The decoding method for this representation starts with the transformation of particle into a priority list of customer to enter route and a priority matrix of vehicle to serve each customer. The vehicle routes are then constructed based on the customer priority list and vehicle priority matrix. The second representation (SR-2) is a $3m$-dimensional particle. The decoding method for this representation starts with the transformation of particle into the vehicle orientation points and the vehicle coverage radius. The vehicle routes are constructed based on these points and radius. The proposed representations are applied using GLNPSO, a PSO algorithm with multiple social learning structures, and tested using some benchmark problems. The computational result shows that representation SR-2 is better than representation SR-1 and also competitive with other methods for solving CVRP.

1. Introduction

The capacitated vehicle routing problem (CVRP) introduced by Dantzig and Ramser (1959), is a problem to design a set of vehicle routes in which a fixed fleet of delivery vehicles of uniform capacity must service known customer demands for a single commodity from a common depot at minimum cost. The CVRP can be formally defined as follows (Cordeau, Gendreau, Laporte, Potvin, & Semet, 2002; Lysgaard, Letchford, & Eglese, 2004; Prins, 2004). A set of n customers require a delivery service from a depot. Each customer i has a non-negative demand q_i and a service time s_i. A fleet of m identical vehicles of capacity Q and service time limit D is stationed at the depot. The depot and customers locations are known; therefore, the travel distance or travel cost between two locations (d_{ij}) and travel time between two locations (t_{ij}) are also known. The CVRP consists of designing a set of at most m delivery routes such that (1) each route starts and ends at the depot, (2) each customer is visited exactly once by exactly one vehicle, (3) the total demand of each route does not exceed Q, (4) the total duration of each route (including travel and service times) does not exceed a preset limit D, and (5) the total routing cost is minimized.

The CVRP is the key operational problem of the vehicle routing problems that must be solved in the daily operation of physical distribution and logistic. Hence, studying this basic problem and methods for finding solution of the problem is essential as the foundation to learn other advanced problem in this field and develop its solution methodology.

It is known that the CVRP is an NP-hard problem (Haimovich, Rinnooy Kan, & Stougie, 1988), in which finding the optimal solution of CVRP instance is very hard and usually requires very long computational time. As a consequence, evolutionary computing methods have been applied for CVRP to find a near optimal solution in a reasonable amount of time, for example: genetic algorithm (Baker & Ayechew, 2003; Berger & Barkaoui, 2003), ant colony optimization (Bullnheimer, Hartl, & Strauss, 1999; Doerner et al., 2002) and particle swarm optimization (Chen, Yang, & Wu, 2006; Ai & Kachitvichyanukul, 2007).

Particle swarm optimization (PSO), which first proposed by Kennedy and Eberhart (1995), is a population based search method that mimics the behavior of group organism as a searching method. In the PSO, a solution of a specific problem is being represented by multi-dimensional position of a particle and a swarm of particles is working together to search the best position which correspond to the best problem solution. In each PSO iteration, every particle moves from its original position to a new position based on its velocity, where particles' velocity is influenced by the cognitive and social information of the particles. The cognitive information of a particle is the best position

* Corresponding author. Fax: +66 2 524 5697.
E-mail addresses: thejin.ai@ait.ac.th, jinai@mail.uajy.ac.id (T.J. Ai), voratas@ait.ac.th (V. Kachitvichyanukul).

0360-8352/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2008.06.012

that has been visited by the particle, i.e. position that provides the best objective function, and the most common social information of the particles is called the global best position, the best position that has been visited by all particles in the swarm. A comprehensive survey on PSO mechanism, technique, and application is provided by Kennedy and Eberhart (2001) and also Clerc (2006).

Two previous researches on the application of PSO to CVRP had different features and characteristics, including the benchmark problems that had been used for testing the algorithms. Chen et al. (2006) applied the discrete version of PSO and combined the method with Simulated Annealing algorithm, while Ai and Kachitvichyanukul (2007) used the classical version of PSO without any hybridization. In term of computational result, Chen's PSO could provide high quality solution for some benchmark problems with number of customers less than 134. However, their method required significantly larger computational time and even almost reached half an hour for the slowest case. On the other hand, Ai and Kachitvichyanukul's PSO could provide solution within relatively fast computational time for some benchmark problems with number of customers less than 199. However, there were some variations on the solution quality.

In order to make PSO applicable to CVRP, the relationship between particle position and vehicle routes must be clearly defined. The definition of particle as an encoded solution is usually called a solution representation and the method to convert it to problem specific solution is usually called a decoding method. This paper proposes two specific solution representations, namely SR-1 and SR-2, and its corresponding decoding method to convert position in PSO into CVRP solution. The solution representation SR-1 is a direct extension of the work of Ai and Kachitvichyanukul (2007), in which a local improvement procedure is added to its decoding method in order to enhance solution quality. The solution representation SR-2 is a new proposed representation which expands the basic idea of SR-1. The decoding method for SR-2 is also incorporated some simple local improvement procedures for increasing solution quality. Both representations are designed for the classic variant of PSO, which is using real value of position. Hence, these representations are different with Chen's work which was based on a discrete-valued representation.

The remainder of this paper is organized as follow: Section 2 reviews PSO framework for solving CVRP. Section 3 explains the proposed solution representations and decoding methods. Section 4 discusses the computational experiment of PSO that applied the solution representations on benchmark data set. Finally, Section 5 summarizes the result of this study and suggests further direction in this research.

2. PSO framework for solving CVRP

The PSO framework for solving CVRP is based on GLNPSO, a PSO Algorithm with multiple social structures (Pongchairerks & Kachitvichyanukul, 2005). In this framework, the particles are initialized in step 1, evaluated its corresponding fitness value within steps 2–3, updated its cognitive and social information within steps 4–7, and moved by step 8. Step 9 is controlling step for repeating or stopping the iteration. Note that the adjustment of this framework from the original GLNPSO algorithm are in step 2, which is the conversion of the position of particle into vehicle routes, and step 3, which is to determine the performance measurement of the routes. Also, this algorithm is designed for the minimization problem, since the CVRP objective is to minimize total routing cost.

The notation and the description of the algorithm are given as follows.

Notation

t	Iteration index; $t = 1 \ldots T$
i	Particle index, $i = 1 \ldots I$
d	Dimension index, $d = 1 \ldots D$
u	Uniform random number in the interval [0,1]
$w(t)$	Inertia weight in the tth iteration
$v_{id}(t)$	Velocity of the ith particle at the dth dimension in the tth iteration
$x_{id}(t)$	Position of the ith particle at the dth dimension in the tth iteration
p_{id}	Personal best position (pbest) of the ith particle at the dth dimension
p_{gd}	Global best position (gbest) at the dth dimension
p^L_{id}	Local best position (lbest) of the ith particle at the dth dimension
p^N_{id}	Near neighbor best position (nbest) of the ith particle at the dth dimension
c_p	Personal best position acceleration constant
c_g	Global best position acceleration constant
c_l	Local best position acceleration constant
c_n	Near neighbor best position acceleration constant
X_i	Vector position of ith particle, $[x_{i1}, x_{i2}, \ldots, x_{iD}]$
V_i	Vector velocity of ith particle, $[v_{i1}, v_{i2}, \ldots, v_{iD}]$
P_i	Vector personal best position of ith particle, $[p_{i1}, p_{i2}, \ldots, p_{iD}]$
P_g	Vector global best position, $[p_{g1}, p_{g2}, \ldots, p_{gD}]$
R_i	Set of vehicle routes corresponding to ith particle
$\varphi(X_i)$	Fitness value of X_i
$X^{\min}$	Minimum position value
$X^{\max}$	Maximum position value
FDR	Fitness-distance-ratio

Algorithm 1. PSO framework for CVRP

1. Initialize I particles as a population, generate the ith particle with random position X_i in the range $[X^{\min}, X^{\max}]$, velocity $V_i = 0$ and personal best $P_i = X_i$ for $i = 1 \ldots I$. Set iteration $t = 1$.
2. For $i = 1 \ldots I$, decode X_i to a set of vehicle route R_i (see Decoding method in Section 3).
3. For $i = 1 \ldots I$, compute the performance measurement of R_i, i.e., the total cost of the routes, and set this as the fitness value of X_i, represented by $\varphi(X_i)$.
4. Update pbest: For $i = 1 \ldots I$, update $P_i = X_i$, if $\varphi(X_i) < \varphi(P_i)$.
5. Update gbest: For $i = 1 \ldots I$, update $P_g = P_i$, if $\varphi(P_i) < \varphi(P_g)$.
6. Update lbest: For $i = 1 \ldots I$, among all pbest from K neighbors of the ith particle, set the personal best which obtains the least fitness value to be P^L_i.
7. Generate nbest: For $i = 1 \ldots I$, and $d = 1 \ldots D$, set $p^N_{id} = p_{jd}$ that maximizing fitness-distance-ratio (FDR) for $j = 1 \ldots I$. Where FDR is defined as:

$$FDR = \frac{\varphi(X_i) - \varphi(P_j)}{|x_{id} - p_{jd}|} \text{ which } i \neq j \tag{1}$$

8. Update the velocity and the position of each ith particle:

$$w(t) = w(T) + \frac{t - T}{1 - T}[w(1) - w(T)] \tag{2}$$

$$v_{id}(t+1) = w(t)v_{id}(t) + c_p u(p_{id} - x_{id}(t)) + c_g u(p_{gd} - x_{id}(t)) + c_l u(p_{lid} - x_{id}(t)) + c_n u(p_{nid} - x_{id}(t)) \tag{3}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{4}$$

If $x_{id}(t+1) > X^{\max}$,

$$x_{id}(t+1) = X^{\max} \quad (5)$$
$$v_{id}(t+1) = 0 \quad (6)$$

If $x_{id}(t+1) < X^{\min}$,

$$x_{id}(t+1) = X^{\min} \quad (7)$$
$$v_{id}(t+1) = 0 \quad (8)$$

9. If the stopping criterion is met, i.e. $t = T$, go to step 10. Otherwise, $t = t + 1$ and return to step 2.
10. Decode P_g as the best set of vehicle route found R^* with its corresponding performance measurement $\varphi\ (P_g)$.

This framework is starting with I particles with a particular representation that corresponds with I different set of vehicle routes. Then by following the PSO movement mechanism, the particles are moving to different positions, which mean other sets of routes are assessed. Whenever a better set of routes is found, its corresponding best particle information is updated. This movement process is iterated with an expectation to find better and better routes. Since the best particle position for the whole swarm is always being kept, the best vehicle route can be decoded from this information at the end of iteration.

This algorithm is flexible for handling different kind of solution representation and vehicle route problems. It can be applied for any real-valued solution representation and it is also possible to use it for solving some VRP variants other than CVRP, as long as the decoding method is clearly defined. As mentioned before, two specific solution representations are proposed in order to apply this algorithm for solving CVRP. The details of these representations will be explained in the following section.

3. Solution representations and decoding methods

3.1. Solution representation SR-1

The solution representation SR-1 of CVRP with n customers and m vehicles consists of $(n + 2m)$ dimensional particle. Each particle dimension is encoded as a real number. The first n dimensions are related to customers, each customer is represented by one dimension. The last $2m$ dimensions are related to vehicles, each vehicle is represented by two dimensions as the reference point in Cartesian map. This solution representation is first proposed by Ai and Kachitvichyanukul (2007).

The decoding method for this representation into the CVRP solution starts with extracting the position value of the first n dimension of particle to make a priority list of customer to enter route. It can be done by sorting the first n dimensional values in ascending order and taking the dimension index as the customer priority list. The next step is to extract the reference point for vehicles from the last $2m$ dimension of particle. The priority matrix of vehicles is constructed based on the relative distance between these points and customers location. A customer is prioritized to be served by vehicle which has closer distance. Finally, vehicle routes are constructed based on the customer priority list and vehicle priority matrix. The basic procedure of this decoding method is also the same with the decoding method of Ai and Kachitvichyanukul (2007). However, there is a slightly modification in the route construction step, which incorporates 2-opt local improvement procedure to a route right after a customer is inserted to the route. Schematic example of the whole decoding procedure of the representation SR-1 for problem with 6 customers and 2 vehicles is shown in Fig. 1 and the route construction procedure is graphically illustrated in Fig. 2. The notation and the decoding algorithm for this representation are presented in Algorithm 2.

Notation
x_{id} Position of the ith particle at the dth dimension
R_{ij} Route of the jth vehicle corresponding to the ith particle

Algorithm 2. Decoding method of solution representation SR-1

1. Construct the priority list of customers (U).
 a. Build set $S = \{1, 2, \ldots, n\}$ and $U = \emptyset$.
 b. Select c from set S where $x_{ic} = \min_{d \in S} x_{id}$.

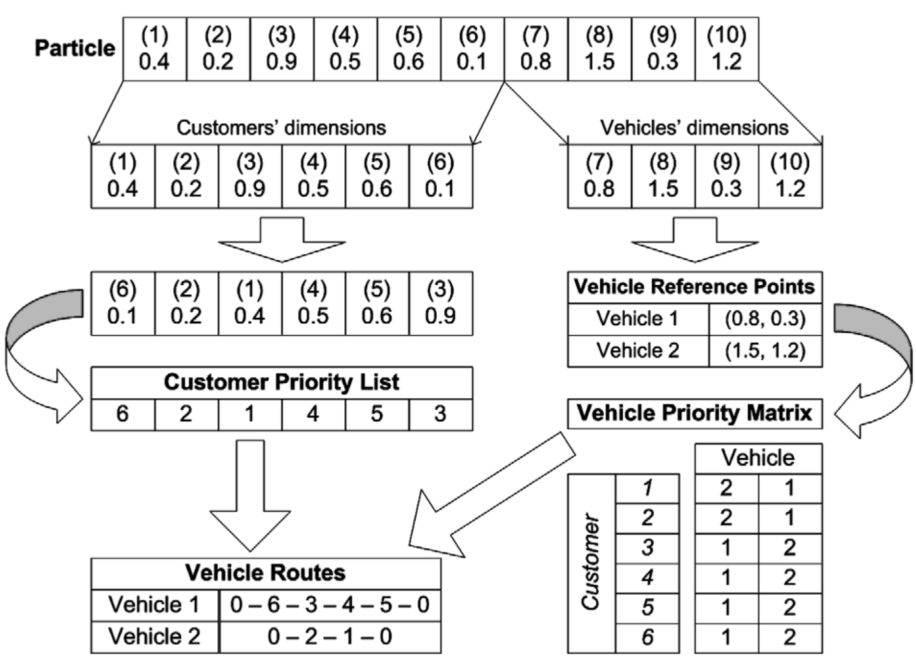

Fig. 1. Solution representation SR-1 and decoding steps to vehicle routes.

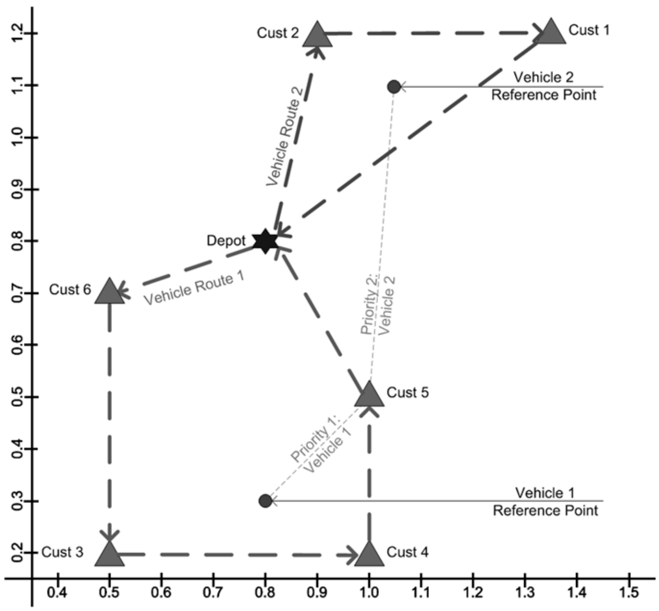

Fig. 2. Illustration of vehicle routes construction of SR-1.

c. Add c to the last position in set U.
d. Remove c from set S.
e. Repeat steps 1b–d until $S = \emptyset$.

2. Construct the vehicle priority matrix (V).
 a. Set the vehicle reference position. For $j = 1 \ldots m$, set $xref_j = x_{i,n+j}$ and $yref_j = x_{i,n+m+j}$.
 b. For each customer k, $k = 1 \ldots n$.
  i. For each vehicle $j = 1 \ldots m$, set δ_j as the Euclidean distance between customer k and the reference point of vehicle j.
  ii. Build set $S = \{1, 2, \ldots, m\}$ and $V_k = \emptyset$.
  iii. Select c from set S where $\delta_c = \min_{d \in S} \delta_d$.
  iv. Add c to the last position in set V_k.
  v. Remove c from set S.
  vi. Repeat step 2b.iii–v until $S = \emptyset$.

3. Construct vehicle route.
 a. Set $k = 1$.
 b. Add customer one by one to the route.
  i. Set $l = U_k$ and $p = 1$.
  ii. Set $j = V_{l,p}$.
  iii. Make a candidate of new route by inserting customer l to the best sequence in the route R_{ij}, which has the smallest additional cost.
  iv. Check the capacity and route time constraint of the candidate route.
  v. If a feasible solution is reached, update the route R_{ij} with the candidate route, then apply 2-opt procedure to the route R_{ij}; go to step 3c.
  vi. If $p = m$, go to step 3c. Otherwise, set $p = p + 1$ and go to step 3b.ii.
 c. If $k = n$, stop. Otherwise, set $k = k + 1$ and repeat 3b.

3.2. Solution representation SR-2

The solution representation SR-2 consists of $3m$ dimensional particle and each particle dimension is encoded as a real number. All dimensions are related to vehicles, each vehicle is represented by three dimensions: two dimensions for the reference point and one dimension for the vehicle coverage radius.

The decoding method for this representation starts with the transformation of particle to the vehicle orientation points and the vehicle coverage radius. The vehicle routes are then constructed based on these points and radius. For each vehicle, starting from the first to the last vehicle, a feasible route consists of customers that located inside its coverage area and have not been assigned to other vehicle is constructed. Vehicle coverage area is defined as an area inside a circle centered at its reference point within its coverage radius. Afterward, the 2-opt, 1-1 exchange, and 1-0 exchange procedures are applied to the constructed routes. If there are remained customers that have not been assigned to any vehicle, the customers are inserted one by one to the existing routes as long as the route feasibility is maintained. Finally, the local improvement procedures are re-applied to all of the routes. Schematic example of the decoding procedure of the representation SR-2 is illustrated in Fig. 3 and the route construction procedure is illustrated in Fig. 4. The formal decoding algorithm for this representation is described in Algorithm 3.

Algorithm 3. Decoding method of solution representation SR-2

1. Extracting vehicle properties, for each vehicle $j = 1 \ldots m$.
 a. Set reference point, $xref_j = x_{i,3j-2}$ and $yref_j = x_{i,3j-1}$.
 b. Set coverage radius, $r_j = x_{i,3j}$.

2. Route construction.
 a. For each vehicle j, construct route of customers that located inside circle with center point $(xref_j, yref_j)$ and radius r_j.
  - Customer is inserted to the route one by one according to its distance from the center point, priority given to closer customer.
  - Consider all constraints (vehicle capacity and routing time constraints) to maintaining route feasibility.
  - Inserting position: best position in the existing route.

 b. Optimize the partial constructed routes with following local improvement procedures: 2-opt, 1-1 exchange and 1-0 exchange.
 c. For remaining customers, insert to the partial constructed routes:
  - Customer is inserted to the route one by one according to its distance from the depot, priority is given to customer located further away.
  - Consider all constraints (vehicle capacity and routing time constraints) to maintaining route feasibility.
  - Inserting position:

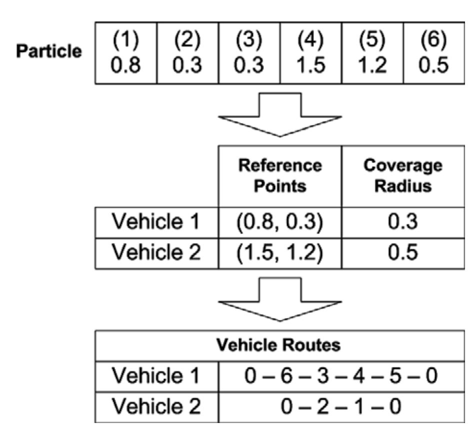

Fig. 3. Solution representation SR-2 and decoding steps to vehicle routes.

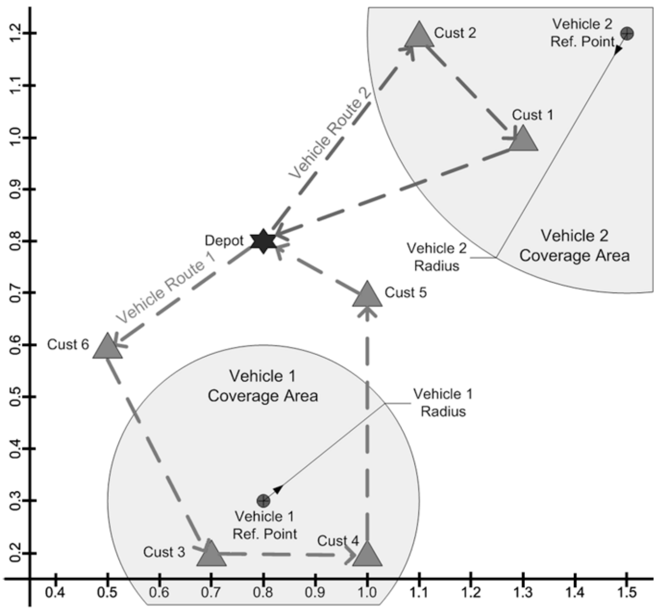

Fig. 4. Illustration of vehicle routes construction of SR-2.

- Vehicle: evaluate all, priority given to the closest vehicle. Distance of a customer to a route is measured by the distance between the customer to the closest customer exists in the route.
- Customer is inserted before the closest existing customer in the route.

d. Optimize the routes with the following local improvement procedures: 2-opt, 1-1 exchange and 1-0 exchange.

3.3. Local improvement procedures

Three common local improvement procedures, 2-opt, 1-1 exchange, and 1-0 exchange, are incorporated in the decoding methods described in the previous sub section. These procedures will be briefly reviewed in this section.

The 2-opt procedure works on improving single route by systematically exchange the route direction between two pairs of consecutive customers in the route and evaluate whether the routing cost of the route is improved or not. The exchange mechanism is illustrated in Fig. 5, in which the route direction between customers $i-(i+1)$ and $j-(j+1)$ are interchanged. If the routing cost of the modified route is better than the routing cost of original one, the route is updated with the modified one.

The 1-1 exchange and 1-0 exchange procedures work on improving two adjacent routes by exchange customer(s) between routes and evaluate whether the total routing cost of the two routes is improved or not. The 1-1 exchange procedure systematically interchanges one customer from the first route with another customer from the second route. The 1-0 exchange procedure systematically moves one customer from the first route to the second route. If the routing cost of the modified routes is better than the routing cost of original ones, the routes are updated with the modified ones. These procedures are demonstrated in Fig. 6 and 7, respectively, in which customer i in the first route is interchanged with customer j in the second route (Fig. 6) and customer i in the first route is moved before customer j in the second route (Fig. 7).

The formal algorithm of these local improvement procedures are described in Algorithms 4–6. In order to reduce the computational effort, an additional step to limit the number of exchange is added to original procedures. In this modified procedures, the exchange is performed whenever the location of customer i and j are within range δ.

Algorithm 4. Procedure 2-opt

1. Set n = numbers of customer in the route.
2. For $i = 1 \ldots (n-2)$ and $j = (i+2) \ldots n$.
   a. Modify route by changing the route direction of customer in the sequence number i, $(i+1)$, j, and $(j+1)$ as shown in Fig. 5.
   b. Evaluate the feasibility of modified route and the routing cost improvement.
   c. If feasibility is maintained and the routing cost is improved, keep the modified route. Otherwise, return the route to the condition before the last step 2a.

Algorithm 5. Procedure 1-1 exchange

1. Set n = number of customers in the first route.
2. Set m = number of customers in the second route.

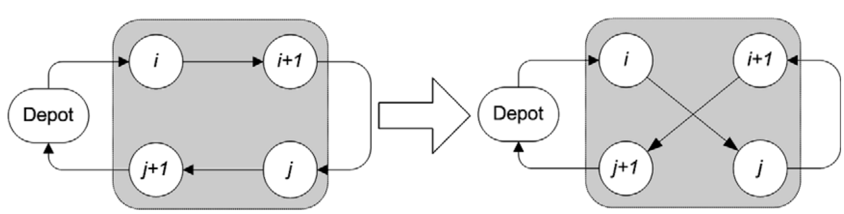

Fig. 5. Illustration of 2-opt procedure.

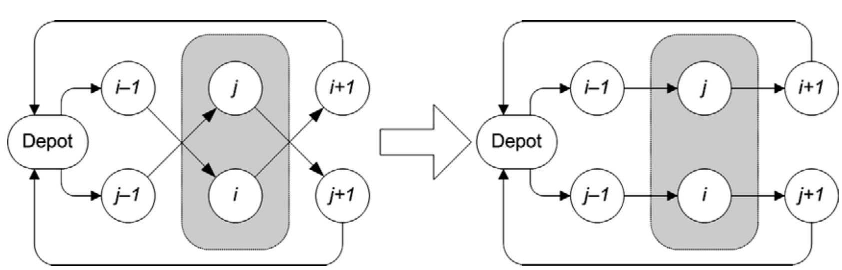

Fig. 6. Illustration of 1-1 exchange procedure.

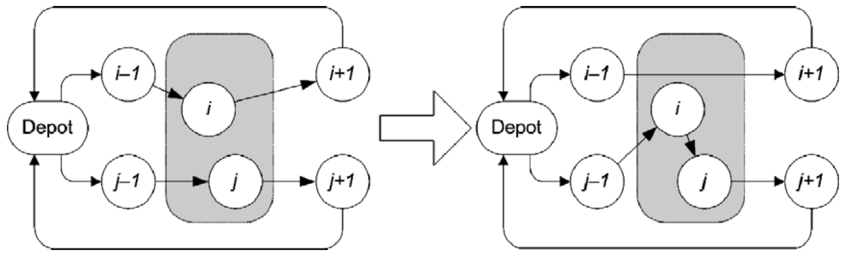

Fig. 7. Illustration of 1-0 exchange procedure.

3. For $i = 1 \ldots n$ and $j = 1 \ldots m$.
   a. If the distance between customer with sequence number i in the first route and customer with sequence number j in the second route is within range δ, go to step 3b. Otherwise, repeat step 3a with the next value of i or j.
   b. Modify routes by interchanging the customer with sequence number i in the first route with the customer with sequence number j in the second route.
   c. Evaluate the feasibility of modified routes and the routing cost improvement.
   d. If feasibility is maintained and the routing cost is improved, keep the modified route. Otherwise, return the route to the condition before the last step 3a.

Algorithm 6. Procedure 1-0 Exchange

1. Set n = number of customers in the first route.
2. Set m = number of customers in the second route.
3. For $i = 1 \ldots n$ and $j = 1 \ldots m$.
   a. If the distance between customer with sequence number i in the first route and customer with sequence number j in the second route is within range δ, go to step 3b. Otherwise, repeat step 3a with the next value of i or j.
   b. Modify routes by moving the customer with sequence number i in the first route before the customer with sequence number j in the second route.
   c. Evaluate the feasibility of modified routes and the routing cost improvement.
   d. If feasibility is maintained and the routing cost is improved, keep the modified route. Otherwise, return the route to the condition before the last step 3a.

4. Computational result

Two set of computational experiments are conducted to test the performance of the PSO with the two solution representations for solving the CVRP. The first set of experiment is performed in order to compare the result of these proposed methods with PSO of Chen et al. (2006). In this experiment, the proposed methods are applied to the same sixteen benchmark problems that had been used by Chen. The second set of experiment is conducted in order to evaluate performance of these methods for the larger size problem. The benchmark data set of Christofides, Mingozzi, and Toth (1979) are selected as testing case, since these data are widely used as CVRP benchmark and they cover larger problems than the Chen's problem set.

The algorithm is implemented in C# language using Microsoft Visual Studio.NET 1.1 on a PC with Intel P4 3.4 GHz – 1 GB RAM. For each data set, 5 replications of the algorithm are tried. The PSO parameters are: Number of Particle, $I = 50$; Number of Iteration, $T = 1000$; Number of Neighbor, $K = 5$; First inertia weight, $w(1) = 0.9$; Last inertia weight, $w(T) = 0.4$; Personal best position acceleration constant, $c_p = 0.5$; Global best position acceleration constant, $c_g = 0.5$; Local best position acceleration constant, $c_l = 1.5$; Near neighbor best position acceleration constant, $c_n = 1.5$.

4.1. Comparison with Chen's results

The first computational experiment is conducted on the same sixteen benchmark problems that had been used by Chen et al. (2006). In these benchmark problems, the total number of customers is varying from 29 to 134 customers, and the total number of vehicles is varying from 3 to 10 vehicles. The computational results of both solution representation SR-1 and SR-2 for these benchmark problems are presented in Table 1, in term of the best objective function found and the average computational time over 5 replications. For comparison purpose, the best known solution (BKS) and the Chen's solution also displayed here. In this table, an objective function value with bold italic typeface indicates that the corresponding solution is exactly equal to the best known solution.

From the solution quality point of view, the result from representation SR-2 is better than Chen's and SR-1 result. It is shown in Table 1 that the SR-2 solutions are very close to the best-known solution, in which the solution of ten out of sixteen instances are exactly same as the best-known solution and the remainders are only slightly larger than the best-known solutions. Furthermore, it provided the best objective function among three methods for almost all instances (fourteen out of sixteen instances).

In term of computational effort, both representations SR-1 and SR-2 are faster than Chen. Even though the Chen experiment is performed on different machines than the SR-1 and SR-2 experiments, the time gap between Chen and SR-1 or SR-2 computational time is significantly large, especially for the big problems, i.e. instance with at least 100 customers (Fn135k7, Mn101k10, etc.). In general, representation SR-1 requires shorter time than SR-2 with

Table 1
Computational result of the Chen's benchmark problems

Instance	No. cust.	No. vhcl.	Objective function (Cost)				Comp. time (s)		
			BKS	Chen	SR-1	SR-2	Chen[b]	SR-1	SR-2
An33k5	32	5	661	***661***[a]	***661***[a]	***661***[a]	32	11	13
An46k7	45	7	914	***914***[a]	***914***[a]	***914***[a]	129	19	23
An60k9	59	9	1354	***1354***[a]	1366	1355	309	28	40
Bn35k5	34	5	955	***955***[a]	***955***[a]	***955***[a]	38	12	14
Bn45k5	44	5	751	***751***[a]	***751***[a]	***751***[a]	134	17	20
Bn68k9	67	9	1272	***1272***[a]	1278	1274	344	33	50
Bn78k10	77	10	1221	1239	1239	1223[a]	429	41	64
En30k3	29	3	534	***534***[a]	541	***534***[a]	28	11	16
En51k5	50	5	521	528	***521***[a]	***521***[a]	301	21	22
En76k7	75	7	682	688	691	***682***[a]	527	38	60
Fn72k4	71	4	237	244	***237***[a]	***237***[a]	398	58	53
Fn135k7	134	7	1162	1215	1184	***1162***[a]	1526	178	258
Mn101k10	100	10	820	824	821	***820***[a]	874	60	114
Mn121k7	120	7	1034	1038	1041	1036[a]	1734	88	89
Pn76k4	75	4	593	602	599	594[a]	496	51	48
Pn101k4	100	4	681	694	686	683[a]	978	99	86

[a] The best objective function among Chen's, SR-1, and SR-2 results.
[b] Computational time on Pentium IV 1.8 GHz with 256 MB RAM.

exceptions on the instances Fn72k4, Pn76k4 and Pn101k4. These exceptions are not uncommon since the computational effort is also depend on the problem instance characteristics.

As summary, the proposed PSO framework with representation SR-2 can be considered as the best method, since it outperformed SR-1 in terms of solution quality and dominated Chen's in term of both solution quality and computational time.

4.2. Christofides benchmark problem

The second computational experiment is conducted on the benchmark problems of Christofides et al. (1979). This benchmark set comprise of problems with randomly distributed and clustered customers, problems with and without route time constraint, and varies number of customers. The computational results for these benchmark problems are presented in Table 2, in term of the best objective function found and the average computational time over five replications. For comparison purpose, the best known solution (BKS) and the Ai and Kachitvichyanukul's (2007) results (SR-1°) are also displayed here. In this table, an objective function value with bold italic typeface indicates that the corresponding solution is exactly equal to the best-known solution (BKS) and the value displayed below the solution is the percentage deviation of the solution from the corresponding best-known solution.

The results on Table 2 shows that the proposed PSO framework with both representation SR-1 and SR-2 could provide a reasonably good solutions, with an exception for instance vrpnc9 with representation SR-1. The results for SR-1 are at most within 5.00% deviation from the best-known solution and the results for SR-2 are at most within 2.51% deviation from the best-known solution. In addition, one SR-1 solution and four SR-2 solutions are approaching exactly the best-known solution. From the computational point of view, it is shown that the representation SR-2 required more computational effort than the SR-1. Also, the computational time for both representations is reasonable, in which less than four and nine minutes, respectively.

Table 2
Computational result of Christofides benchmark problems

Instance	No. cust.	No. vhcl.	Objective function (Cost)				Comp. time (s)		
			BKS	SR-1°	SR-1	SR-2	SR-1°	SR-1	SR-2
vrpnc1	50	5	524.61	***524.61*** 0.00%	***524.61*** 0.00%	***524.61***[a] 0.00%	47	21	24
vrpnc2	75	10	835.26	865.86 3.66%	849.58 1.71%	844.42[a] 1.10%	99	39	57
vrpnc3	100	8	826.14	840.91 1.79%	835.80 1.17%	829.40[a] 0.39%	123	61	101
vrpnc4	150	12	1028.42	1068.22 3.87%	1067.57 3.81%	1048.89[a] 1.99%	235	113	223
vrpnc5	199	17	1291.45	1365.15 5.72%	1345.84 4.21%	1323.89[a] 2.51%	375	188	413
vrpnc6	50	6	555.43	560.89 0.98%	556.68 0.23%	***555.43***[a] 0.00%	51	21	30
vrpnc7	75	11	909.68	–	952.77 4.74%	917.68[a] 0.88%	–	42	69
vrpnc8	100	9	865.94	878.59 1.46%	877.84 1.37%	867.01[a] 0.12%	133	61	115
vrpnc9	150	14	1162.55	–	N/A[b]	1181.14[a] 1.60%	–	125	295
vrpnc10	199	18	1395.85	–	1465.66 5.00%	1428.46[a] 2.34%	–	208	517
vrpnc11	120	7	1042.11	1045.38 0.31%	1051.87[a] 0.94%	1052.34 0.98%	133	89	93
vrpnc12	100	10	819.56	820.62 0.13%	820.62 0.13%	***819.56***[a] 0.00%	128	60	88
vrpnc13	120	11	1541.14	1569.14 1.84%	1566.32 1.63%	1546.20[a] 0.33%	178	86	160
vrpnc14	100	11	866.37	***866.37*** 0.00%	867.13 0.09%	***866.37***[a] 0.00%	137	64	99

[a] The best objective function between SR-1 and SR-2 results.
[b] All replications yielded infeasible solution.

The comparison between the results of representation SR-1° and SR-1 proves that the addition of the local improvement procedure in the decoding method has significant effect to the solution quality, since the SR-1 results are generally better than the SR-1° results. Even though this addition causes an additional effort, however, the total computational effort can be reduced by using smaller number of particles without affecting the solution quality. The computational results demonstrate that the representation SR-1 using 50 particles is approximately twice faster than the representation SR-1° which was using 100 particles, even if the quality of solution from SR-1 is better than SR-1°.

The high-quality result yielded by the proposed method may come from two factors. First, the decoding scheme gives higher possibility to get feasible solution, since a rigorous constraint checking has already been done while constructing the route. Second, the solution quality is improved from the route construction heuristics, including the local improvement procedures. The combinations of these efforts are potential for yielding good solutions.

5. Conclusion

This paper presents two solution representations, SR-1 and SR-2, and the corresponding decoding methods for solving the capacitated vehicle routing problem (CVRP) using particle swarm optimization (PSO). The representation SR-1 is a $(n+2m)$-dimensional particle for CVRP with n customers and m vehicles. The representation SR-2 is a $3m$-dimensional particle. The proposed representations are applied using a framework based on GLNPSO, a PSO algorithm with multiple social learning structures, and tested using some benchmark problems.

The computational result shows that proposed PSO framework with both representation SR-1 and SR-2 is effective for solving CVRP. Both representations are proven more effective than the other PSO method for solving CVRP, in term of solution quality and computational time. In term of solution quality, it is shown that the proposed PSO framework with representation SR-2 is better than the framework with representation SR-1. However, the representation SR-2 required more computational effort than the representation SR-1.

Some further research for applying the proposed method to other VRP variants should be carried out. Since the variants of VRP differ from one another only on the specific problem constraints, the adjustment is only required in the constraint feasibility checking of the decoding method. However, the effectiveness of this idea needs further exploration.

Acknowledgements

The authors wish to thank the High Performance Computing Group at Asian Institute of Technology (AITHPC) and the Thai GRID Center for the access of computing facility and the technical support.

The first author expresses gratitude to Universitas Atma Jaya Yogyakarta, Indonesia for providing the financial support for his study in Asian Institute of Technology, Thailand.

References

Ai, T. J., & Kachitvichyanukul, V. (2007). A particle swarm optimization for the capacitated vehicle routing problem. *International Journal of Logistics and SCM Systems, 2*, 50–55.

Baker, B. M., & Ayechew, M. A. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research, 30*, 787–800.

Berger, J., & Barkaoui, M. (2003). A new hybrid genetic algorithm for the capacitated vehicle routing problem. *Journal of the Operational Research Society, 54*, 1254–1262.

Bullnheimer, R., Hartl, F., & Strauss, C. (1999). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research, 89*, 319–328.

Chen, A. L., Yang, G. K., & Wu, Z. M. (2006). Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. *Journal of Zhejiang University Science A, 7*, 607–614.

Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. In N. Christofides et al. (Eds.), *Combinatorial Optimization* (pp. 315–338). Chicester: John Wiley & Sons.

Clerc, M. (2006). *Particle Swarm Optimization*. London: ISTE.

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J. Y., & Semet, F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research Society, 53*, 512–522.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science, 6*, 80–91.

Doerner, K., Gronalt, M., Hartl, R. F., Reimann, M., Strauss, C., & Stummer, M. (2002). Saving ants for the vehicle routing problem. In Cagnoni et al. (Eds.), *EvoWorkshops 2002*. *LNCS* (vol. 2279, pp. 11–20). Berlin: Springer-Verlag.

Haimovich, M., Rinnooy Kan, A. H. G., & Stougie, L. (1988). Analysis of heuristics for vehicle routing problems. In Golden B. L., & Assad, A. A., *Vehicle routing: methods and studies*. Amsterdam: North-Holland.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 1942–1948.

Kennedy, J., & Eberhart, R. C. (2001). *Swarm Intelligence*. San Francisco: Morgan Kaufman Publishers.

Lysgaard, J., Letchford, A. N., & Eglese, R. W. (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming, 100*, 423–445.

Pongchairerks, P., & Kachitvichyanukul, V. (2005). A non-homogenous particle swarm optimization with multiple social structures. In *Proceedings of international conference on simulation and modeling*. A5-02.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research, 31*, 1985–2002.