Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem*

CHEN Ai-ling†, YANG Gen-ke, WU Zhi-ming
(*Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China*)
†E-mail: chengchengcal@sjtu.edu.cn
Received Aug. 3, 2005; revision accepted Oct. 19, 2005

Abstract: Capacitated vehicle routing problem (CVRP) is an NP-hard problem. For large-scale problems, it is quite difficult to achieve an optimal solution with traditional optimization methods due to the high computational complexity. A new hybrid approximation algorithm is developed in this work to solve the problem. In the hybrid algorithm, discrete particle swarm optimization (DPSO) combines global search and local search to search for the optimal results and simulated annealing (SA) uses certain probability to avoid being trapped in a local optimum. The computational study showed that the proposed algorithm is a feasible and effective approach for capacitated vehicle routing problem, especially for large scale problems.

Key words: Capacitated routing problem, Discrete particle swarm optimization (DPSO), Simulated annealing (SA)
doi:10.1631/jzus.2006.A0607 **Document code:** A **CLC number:** TP14

INTRODUCTION

The vehicle routing problem (VRP), which was first introduced by Dantzig and Ramser (1959), is a well-known combinatorial optimization problem in the field of service operations management and logistics. The capacitated vehicle routing problem (CVRP) is an NP-hard problem for simultaneously determining the routes for several vehicles from a central depot to a set of customers, and then return to the depot without exceeding the capacity constraints of each vehicle. In practice, the problem is aimed at minimizing the total cost of the combined routes for a fleet of vehicles. Since cost is closely associated with distance, in general, the goal is to minimize the distance travelled by a fleet of vehicles with various constraints.

Many different approaches have been developed to solve the CVRP. In general, the approaches are divided into two classes: exact algorithms (Christofides *et al*., 1981; Toth and Vigo, 1998) and heuristic algorithms. Since the capacitated vehicle routing problem is an NP-hard problem (Laporte, 1992), as Toth and Vigo (2002) reported, no exact algorithm can consistently solve CVRP-instances with more than 50 customers; thus, the heuristic approaches are considered as reasonable choice in finding solutions for large-scale instances. Available heuristics include simulated annealing algorithms (Osman, 1993), tabu search algorithms (Gendreau *et al*., 1994; Cordeau *et al*., 2001), genetic algorithms (Baker and Ayechew, 2003), and ant colony algorithm (Bell and McMullen, 2004). In this paper, we introduce a very fast and easily implemented hybrid algorithm based on discrete particle swarm optimization (DPSO) and simulated annealing (SA) algorithm. The proposed method uses DPSO to assign the customers on routes and SA algorithm to avoid becoming trapped in local optimum.

PROBLEM DESCRIPTION

The capacitated vehicle routing problem is a

* Project (No. 60174009) supported by the National Natural Science Foundation of China

difficult combinatorial optimization problem, and generally can be described as follows: Goods are to be delivered to a set of customers by a fleet of vehicles from a central depot. The locations of the depot and the customers are given. The objective is to determine a viable route schedule which minimizes the distance or the total cost with the following constraints:

(1) Each customer is served exactly once by exactly one vehicle;

(2) Each vehicle starts and ends its route at the depot;

(3) The total length of each route must not exceed the constraint;

(4) The total demand of any route must not exceed the capacity of the vehicle.

Assume that the depot is node 0, and N customers are to be served by K vehicles. The demand of customer i is q_i, the capacity of vehicle k is Q_k, and the maximum allowed travel distance by vehicle k is D_k. Then the mathematical model of the CVRP based on the formulation given by Bodin *et al*.(1983) is described as follows:

$$\text{Minimize} \quad \sum_{k=1}^{K}\sum_{i=0}^{N}\sum_{j=0}^{N} C_{ij}^{k} X_{ij}^{k} \tag{1}$$

subject to:

$$X_{ij}^{k} = \begin{cases} 1 & \text{if vehicle } k \text{ travels from customer } i \text{ to } j, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

$$\sum_{k=1}^{K}\sum_{i=0}^{N} X_{ij}^{k} = 1, \quad j = 1, 2, \ldots, N, \tag{3}$$

$$\sum_{k=1}^{K}\sum_{j=0}^{N} X_{ij}^{k} = 1, \quad i = 1, 2, \ldots, N, \tag{4}$$

$$\sum_{i=0}^{N} X_{it}^{k} - \sum_{j=0}^{N} X_{tj}^{k} = 0, \; k=1, 2, \ldots, K; \; t=1, 2, \ldots, N, \tag{5}$$

$$\sum_{i=0}^{N}\sum_{j=0}^{N} d_{ij}^{k} X_{ij}^{k} \le D_k, \quad k = 1, 2, \ldots, K, \tag{6}$$

$$\sum_{j=0}^{N} q_j \left(\sum_{i=0}^{N} X_{ij}^{k} \right) \le Q_k, \quad k = 1, 2, \ldots, K, \tag{7}$$

$$\sum_{j=1}^{N} X_{0j}^{k} \le 1, \quad k = 1, 2, \ldots, K, \tag{8}$$

$$\sum_{i=1}^{N} X_{i0}^{k} \le 1, \quad k = 1, 2, \ldots, K, \tag{9}$$

$$X_{ij}^{k} \in \{0,1\}, \; i, j=0, 1, 2, \ldots, N; \; k=1, 2, \ldots, K, \tag{10}$$

where N represents the number of customers, and K is the number of vehicles, and C_{ij}^{k} is the cost of travelling from customer i to customer j by vehicle k and d_{ij}^{k} is the travel distance from customer i to customer j by vehicle k.

The objective function Eq.(1) is to minimize the total cost by all vehicles. Constraints Eqs.(3) and (4) ensure that each customer is served exactly once. Constraint Eq.(5) ensures the route continuity. Constraint Eq.(6) shows that the total length of each route has a limit. Constraint Eq.(7) shows that the total demand of any route must not exceed the capacity of the vehicle. Constraints Eqs.(8) and (9) ensure that each vehicle is used no more than once. Constraint Eq.(10) ensures that the variable only takes the integer 0 or 1.

DISCRETE PARTICLE SWARM OPTIMIZATION

Standard particle swarm optimization

Particle swarm optimization (PSO) is a parallel population-based computation technique proposed by Kennedy and Eberhart (Eberhart and Kennedy, 1995; Kennedy and Eberhart, 1995), which was motivated by the organisms behavior such as schooling of fish and flocking of birds. PSO can solve a variety of difficult optimization problems (Salman *et al*., 2002; Shigenori *et al*., 2003). PSO's major difference from genetic algorithm (GA) is that PSO uses the physical movements of the individuals in the swarm and has a flexible and well-balanced mechanism to enhance and adapt to the global and local exploration abilities, whereas GA uses genetic operators. Another advantage of PSO is its simplicity in coding and consistency in performance. The global optimizing model proposed by Shi and Eberhart (1999) is as follows:

$$\begin{aligned} V_{id} = W \times V_{id} + C_1 \times Rand \times (P_{\text{best}} - X_{id}) \\ + C_2 \times rand \times (G_{\text{best}} - X_{id}), \end{aligned} \tag{11a}$$

$$X_{id} = X_{id} + V_{id}, \tag{11b}$$

where V_{id} is the velocity of particle i, X_{id} is the particle position, W is the inertial weight. C_1 and C_2 are the positive constant parameters, *Rand* and *rand* are the random functions in the range [0, 1], P_{best} is the best position of the ith particle and G_{best} is the best position among all particles in the swarm.

Discrete particle swarm optimization (DPSO)

Due to its global and local exploration abilities, simplicity in coding and consistency in performance, PSO algorithm has been widely applied in many fields although PSO algorithm was originally proposed for continuous optimization problems. In our research, we mainly use discrete data to process problems. Therefore, developing a mechanism to realize discrete optimization problem is attractive. We adopt the quantum discrete PSO algorithm proposed by Yang *et al.*(2004) to solve the CVRP.

In quantum theory, a bit that is the minimum unit carrying information is always in a state of the range [0, 1]. A quantum particle vector is defined as follows:

$$\boldsymbol{V}=[\boldsymbol{V}_1,\boldsymbol{V}_2,...,\boldsymbol{V}_M],\ (\boldsymbol{V}_i=[v_i^1,v_i^2,...,v_i^N]), \tag{12}$$

where $0\le v_i^j\le 1$ (i=1, 2, …, M; j=1, 2, …, N); N is the particle's length and M is the swarm size. v_i^j denotes the probability of the jth bit of the ith particle being 0. The following description is the rule from a quantum particle vector to a discrete particle vector.

Assume that $\boldsymbol{X}=[\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_M]$ ($\boldsymbol{X}_i=[x_i^1,x_i^2,...,x_i^N]$) is the particle denotation for the practical problems. Where $x_i^j\in\{0,1\}$ (i=1, 2, …, M; j=1, 2, …, N) represents the corresponding discrete particle position of the quantum particle v_i^j, N is the particle's length and M is the swarm size. For each v_i^j (i=1, 2, …, M; j=1, 2, …, N), generate a random number in the range [0, 1]. If the random number is greater than v_i^j, then x_i^j=1, otherwise x_i^j=0. DPSO algorithm can be described as follows:

$$V_{\text{localbest}}=\alpha\times x_{\text{localbest}}+\beta\times(1-x_{\text{localbest}}), \tag{13a}$$
$$V_{\text{globalbest}}=\alpha\times x_{\text{globalbest}}+\beta\times(1-x_{\text{globalbest}}), \tag{13b}$$
$$\boldsymbol{V}=w\times\boldsymbol{V}+c1\times V_{\text{localbest}}+c2\times V_{\text{globalbest}}, \tag{13c}$$

where $\alpha+\beta=1$, $0<\alpha, \beta<1$ are control parameters which indicate the control degree of $\boldsymbol{V}$. $w+c1+c2=1$, $0< w, c1, c2<1$. In Eq.(13c), the first part represents the inertia of previous probability; the second part is the "cognition" part, which represents the local exploration probability; the third part is the "social" part, which represents the cooperation among all quantum particles. So w, $c1$ and $c2$ represent the degree of the belief in oneself, local exploration and global exploration, respectively. The process of implementing the discrete PSO (DPSO) is described as follows:

Step 1: Initialize the quantum particles $\boldsymbol{V}$ and the discrete particles $\boldsymbol{X}$.

Step 2: For discrete particles $\boldsymbol{X}$, calculate the fitness.

Step 3: Calculate $V_{\text{localbest}}$ according to Eq.(13a).

Step 4: Calculate $V_{\text{globalbest}}$ according to Eq.(13b).

Step 5: Compute quantum probability $\boldsymbol{V}$ according to Eq.(13c).

Step 6: Calculate discrete particles $\boldsymbol{X}$, If $rand>v_i^j$, then x_i^j=1; else x_i^j=0.

Step 7: Loop to Step 2 until one of the stopping criteria (generally, a sufficiently good fitness or the specified number of generations) is satisfied.

Fitness function

Fitness is used to evaluate the performance of particles in the swarm. Generally, choosing a proper objective function as fitness function to represent the corresponding superiority of each particle is one of the key factors for successful resolution of the relevant problem using DPSO algorithm. In the CVRP, the objective is to minimize the total cost or distance. Therefore, according to the description in Section 2, we choose the following equation as fitness function:

$$Fit=\sum_{k=1}^{K}\sum_{i=0}^{N}\sum_{j=0}^{N}C_{ij}X_{ij}^k. \tag{14}$$

The fitness function constraints are described in Section 2. The objective of the scheduling is to minimize the total cost, i.e. *Fit*, so the particle with the minimal fitness will outperform others and should be reserved during the optimization process.

DPSO for capacitated vehicle routing problem

In this section, we describe the formulation of DPSO algorithm for the capacitated vehicle routing problem. How to encode a schedule is one of the key issues in successfully applying DPSO to the CVRP, namely, finding a suitable mapping from problem solution to DPSO particle. Consider the problem in which N customers are to be served by K vehicles, and then we can set up a search space of $N\times K$ dimensions. Every particle is composed of K sections and every section has N discrete points. The value of each discrete point is 0 or 1. If the value is 1, it represents that the corresponding customer is served by the relevant vehicle. The position of each particle indicates the relevant sequence of the customers served by each vehicle. We can give an example of the problem in which 8 customers are served by 2 vehicles. Fig.1 shows a stochastic particle position of this example.

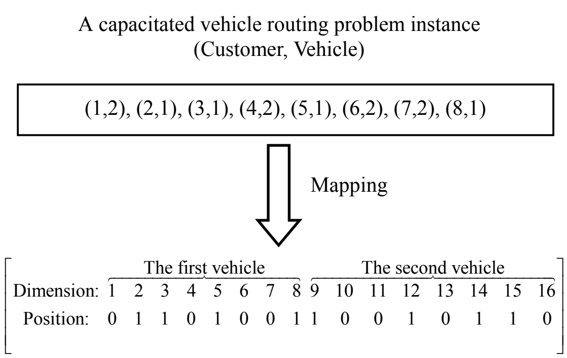

Fig.1 A DPSO particle mapping for the example

In the encoding process, a particle is represented as a 2D array as shown in Fig.1. For the CVRP in which N customers are to be served by K vehicles, the first dimension in the 2D array of a particle is an $N\times K$ dimension vector ($s_1, s_2, \ldots, s_{N\times K}$), where s_i (i=1, 2, …, $N\times K$) is a natural number in the range $[1, N\times K]$, which is not equal to each other. $l=s_i-\lfloor (s_i-1)/N\rfloor\times N$ represents the lth customer and $k=\lfloor (s_i-1)/N\rfloor+1$ denotes the kth vehicle. The second dimension is also an $N\times K$ dimension vector, where every position takes 0 or 1. If the (s_i)th bit takes 1, it represents that the lth customer ($l=s_i-\lfloor (s_i-1)/N\rfloor\times N$) will be served by the kth vehicle ($k=\lfloor (s_i-1)/N\rfloor+1$). Otherwise, the lth customer will not be served by the kth vehicle.

Based on the CVRP constraints, it must be guaranteed that each customer is served exactly once by exactly one vehicle, the total length of each route must not exceed the constraint and the total demand of any route must not exceed the capacity of the vehicle. However, DPSO operation cannot ensure the constraints, so we must check the solutions after DPSO operation as follows:

Check each particle according to each route. If the value of more than one position in the corresponding positions of all sections in this particle is 1, we randomly select one position in these positions and make its value be 1 and others be 0. If the values of the corresponding positions of all sections in this particle are all 0, we randomly select one position and make its value be 1 and others unchanged. If the total length in the route exceeds the limited value or the total demand of each route exceeds the capacity of the vehicle, the solution is unfeasible. For unfeasible solutions, carry out the DPSO operation until the solutions become feasible.

SIMULATED ANNEALING ALGORITHM

Simulated annealing (SA) algorithm is a meta-strategy local search method that attempts to avoid producing the poor local maximum inherent in the steepest ascent method. It employs additional random acceptance and selection strategies. The random acceptance strategy allows occasional downhill moves to be accepted with certain probabilities (Hasan and Osman, 1995). SA algorithm has produced good results for many scheduling problems (Osman and Potts, 1989; Osman, 1993; van Laarhoven *et al*., 1992; Xia and Wu, 2005).

In SA algorithm, the improvements are obtained by choosing another solution (s') that belongs to the neighborhood [$N(s_0)$] of the current solution (s_0). When the current solution changes from s_0 to s', the objective function will also change, namely, $\Delta=Fit(s')-Fit(s_0)$. For the minimization problem, if $\Delta<0$, the new solution s' will be accepted. If $\Delta\geq 0$, the new solution will be accepted with the probability $\exp(-\Delta/t)$, where t is the temperature. Generally, the algorithm starts from a high temperature, and then the temperature is gradually decreased. At each temperature, the search will be performed for a certain number of iterations, which is called the temperature length. When the termination condition is satisfied, the algorithm will stop.

For the CVRP, DPSO algorithm is used to realize the assignment process. SA algorithm can be used to sequence the customers served by each vehicle.

Neighborhood selection

In the process of sequencing the customers in each route using SA algorithm, the neighborhood selection rule greatly influences the performance of the solution for CVRP. In our research, we adopt pair-exchange to obtain neighbors, namely, swap the positions of adjacent elements. Fig.2 is the result of stochastic pair exchange for the first route shown in Fig.1.

$$\begin{bmatrix} & \overbrace{\begin{array}{cccccccc} & & & & & & & \end{array}}^{\text{The first route}} \\ \text{Dimension:} & \begin{array}{cccccccc} 2 & 1 & 4 & 3 & 6 & 5 & 8 & 7 \end{array} \\ \text{Position:} & \begin{array}{cccccccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{array} \\ & \end{bmatrix}$$

Fig.2 Result of pair exchange for a route

After exchanging the customers in the same route of a particle using pair exchange rule every time, the fitness of the new solution is calculated. If the fitness is improved, the new solution is accepted. Otherwise, the new solution is accepted with the probability $\exp(-\Delta/t)$. Through the above process, we can obtain satisfactory results rapidly.

Control parameters selection

An SA algorithm generally must be carefully designed as the choice of its parameters might affect the quality of the solution and computation (Hasan and Osman, 1995). In general, a slow search will lead to better solutions. However, the slow search tends to consume more computation time. Therefore, it is necessary to take a tradeoff between them.

Control parameters were set according to problem characteristics. Through many experiments, we found that the solutions and running time are both better when initial temperature is set according to the maximal difference in fitness value between any two neighboring solutions. The length of temperature denotes the number of moves made at the same temperature, and generally, it is set according to the size of neighborhood solutions for a given solution. In SA optimization process, the temperature is gradually lowered. It is well known that the method that specifies temperature with the equation $t_n=\lambda t_{n-1}$ is often a good choice and can provide a tradeoff between computational time and good solutions. The smaller the cooling rate λ, the quicker temperature descends. To terminate the algorithm, we select the termination temperature (t_f), when current temperature $t<t_f$ the algorithm will stop. Generally, t_f is a small value.

DPSO-SA OPTIMIZATION ALGORITHM

By combining DPSO with SA algorithm, we can get a new hybrid optimization approach DPSO-SA. DPSO has strong global search ability but, as a stochastic search algorithm, cannot guarantee to converge to the global optimal solution at the end. SA algorithm uses a certain probability to avoid becoming trapped in a local optimum. This hybrid approach makes full use of the strong global search ability of DPSO and the strong local search ability of SA and offsets the weaknesses of each other. The algorithm is shown in Fig.3.

COMPUTATIONAL RESULTS

To illustrate the effectiveness and good performance of the proposed algorithm, various kinds of benchmark instances with different sizes have been selected for the computation.

We programmed the algorithms in Matlab 6.5 and ran them on Mobile Intel Pentium IV CPU 1.80 GHz with 256 M RAM.

For small-scale problems, computational experiment was carried out on the following instance: 8 customers are served by 2 vehicles, each with the capacity is 8 and the constraint of the total length of each route is 40. The depot is node 0. The distance between different customers and the demand of each customer are given in Table 1.

To evaluate the approach proposed in the paper, the results from this approach were compared with those from double populations genetic algorithm (DGA) proposed by Zhao *et al.*(2004). The parameters for the two approaches are selected by many experiments as follows:

The parameters for DGA: size of each population: 30; maximum number of generations: 100; crossover rates of two populations are 0.7 and 0.8, and

Begin
Initialize parameters: swarm size, maximum of generation, α, β, w, $c1$, $c2$; Set t_0, t_f, λ, R by experiment; Generation:=1;
Initialize quantum particles V and discrete particles $\boldsymbol{X}$;
Evaluate each particle's fitness according to Eq.(14);
Obtain $X_{\text{globalbest}}$ and $X_{\text{localbest}}$;
Repeat
Compute $V_{\text{localbest}}$ according to Eq.(13a) and compute $V_{\text{globalbest}}$ according to Eq.(13b);
Compute quantum probability $\boldsymbol{V}$ according to Eq.(13c);
Obtain $\boldsymbol{X}$ according to $\boldsymbol{V}$;
If $(rand>V_i^j)$ x_i^j=1;
Else x_i^j=0;
Compute each particle's fitness according to Eq.(14);
Find new $X_{\text{globalbest}}$ and $X_{\text{localbest}}$ and update $X_{\text{globalbest}}$ and $X_{\text{localbest}}$;
Carry out SA subprogram on each route of each particle;
Compute the particle's fitness according to Eq.(14);
Update $X_{\text{globalbest}}$ and $X_{\text{localbest}}$;
Generation=Generation+1;
Until (one of termination conditions is satisfied)
Output the optimization results;
End

SA algorithm subprogram (for each route of each particle):
{t_n=t_0;
Repeat
r=1;
Repeat
Generate a neighboring solution s' from s by the pair exchange rule;
Compute fitness of s', then Δ=$Fit(s')$−$Fit(s)$;
If (Δ<0) s' is accepted;
Else
If ($rand$<exp(−Δ/t_n)) s' is accepted;
Update the best solution found so far;
r=r+1;
Until (r>R)
t_n=$\lambda\times t_n$;
Until (t_n<t_f)}

Fig.3 DPSO-SA hybrid optimization algorithm

mutation rates are 0.05 and 0.1, respectively.

The parameters for DPSO-SA: maximum number of generations: 100; swarm size: 30; α=0.35, β=0.65, w=0.25, $c1$=0.25, $c2$=0.5, t_0=30, t_f=0.1, λ=0.9.

The instance is randomly run 10 times. Table 2 shows the results in 10 runs.

From Table 2, it is very clear that DPSO-SA yields the optimal solutions in all 10 experiments. The corresponding optimal route is: route 1: 0→2→8→5→3→1→0, the quantity of goods delivered is 7 and the travel distance is 34; route 2: 0→6→7→4→0, the quantity of goods delivered is 8 and the travel distance is 33.5. While DGA yielded the optimal route only two times over 10 runs. The results were much worse than those of DPSO-SA.

For problems of a little larger scale, computational experiment was conducted on the instances from Vehicle Routing Data Sets (http://www.Branchandcut.org/VRP/data/) and the results were compared with those from GA with 2-opt proposed by Wang *et al*.(2004) and those from SA.

To obtain proper parameters, each of the algorithms was run from different random initial solutions under many different parameter settings. After many experiments, the parameters of each algorithm were set as follows:

The parameters for GA with 2-opt: Crossover rate: 0.75; Mutation rates: (Swap: 0.05, Insertion: 0.15, Inversion: 0.01); Population size and termination condition are set according to the problem scale respectively.

The parameters for DPSO-SA: α=0.30, β=0.70, w=0.2, $c1$=0.3, $c2$=0.5. Swarm size and termination condition are also given according to the problem

Table 1 Distance between customers and the requirement qualities of each customer

Customer	Demand	Customer								
		0	1	2	3	4	5	6	7	8
0	–	0	4	6	7.5	9	20	10	16	8
1	1	4	0	6.5	4	10	5	7.5	11	10
2	2	6	6.5	0	7.5	10	10	7.5	7.5	7.5
3	1	7.5	4	7.5	0	10	5	9.0	9.0	15
4	2	9.0	10	10	10	0	10	7.5	7.5	10
5	1	20	5	10	5	10	0	7.0	9.0	7.5
6	4	10	7.5	7.5	9.0	7.5	7	0	7.0	10
7	2	16	11	7.5	9.0	7.5	9	7.0	0	10
8	2	8	10	7.5	15	10	7.5	0	10	0

Table 2 Comparison of results of DGA and DPSO-SA

Order	DGA	PSO-SA
1	69.0	67.5
2	70.5	67.5
3	67.5	67.5
4	70.5	67.5
5	69.0	67.5
6	71.0	67.5
7	69.5	67.5
8	70.0	67.5
9	67.5	67.5
10	70.0	67.5

scale; t_0, t_f, λ, R are set based on the experiments and the specific problems.

The parameters for SA: t_0, t_f, λ, R are set based on the experiments and the specific problems.

The algorithms were terminated whenever there was no improvement in 30 successive generations, which enables a reduction in running time.

Each instance was randomly run 5 times for each algorithm. Computation time and best results are shown in Table 3.

Table 3 presents the computational performance of each algorithm. It can be observed that the results of the DPSO-SA algorithm are very close to the best known solution so far. Even if for the large-scale problem, the solutions of DPSO-SA are also very good. The computational speed of an application is also an important means of measuring the ability of an algorithm. Therefore, running time of each algorithm is reported. Through comparison, we can find that the speed of DPSO-SA is greater than that of the other two methods for most of the test instances. Therefore, DPSO-SA algorithm is a feasible and effective approach for solving CVRP.

CONCLUSION

We studied a new hybrid optimization approach combining discrete particle swarm optimization and simulated annealing to solve the capacitated vehicle routing problem. The performance of the approach is evaluated for comparison with the results obtained by other methods for a number of benchmark instances. Although the global optimality cannot be guaranteed, the performance of the results is greatly improved. Moreover, the DPSO-SA algorithm is efficient in running time.

It should be noted that research on solving the problem of larger size can be a challenging task. Future

Table 3 Comparison results of GA with 2-opt and DPSO-SA for the instances*

Problem	N	K	GA with 2-opt		DPSO-SA		SA		BKS
			Distance	Time (s)	Distance	Time (s)	Distance	Time (s)	
A-n33-k5	32	5	661	39.6	661	32.3	661	38.2	661
A-n46-k7	45	7	928	136.4	914	128.9	931	143.8	914
A-n60-k9	59	9	1360	295.5	1354	308.8	1363	286.3	1354
B-n35-k5	34	5	955	46.9	955	37.6	960	58.4	955
B-n45-k5	44	5	762	129.3	751	134.2	760	123.5	751
B-n68-k9	67	9	1296	396.2	1272	344.3	1298	409.2	1272
B-n78-k10	77	10	1248	568.4	1239	429.4	1256	483.3	1221
E-n30-k3	29	3	534	30.5	534	28.4	534	69.3	534
E-n51-k5	50	5	531	289.6	528	300.5	541	362.4	521
E-n76-k7	75	7	697	498.7	688	526.5	704	619.3	682
F-n72-k4	71	4	246	468.5	244	398.3	253	604.6	237
F-n135-k7	134	7	1246	1894.2	1215	1526.3	1243	2533.9	1162
M-n101-k10	100	10	836	992.1	824	874.2	848	986.6	820
M-n121-k7	120	7	1068	1643.1	1038	1733.5	1081	2729.5	1034
P-n76-k4	75	4	605	528.4	602	496.3	612	489.6	593
P-n101-k4	100	4	706	1213.2	694	977.5	715	1964.9	681

* N is the number of customers, K is the number of vehicles, BKS is the best known solution so far; GA with 2-opt represents the best objective value of GA with 2-opt found over 5 runs; DPSO-SA denotes the best objective value of DPSO-SA algorithm found over 5 runs; SA represents the best objective value of SA found over 5 runs. Distance is the best distance of each algorithm. Time is the time needed to obtain the best value

research should focus on improving the DPSO-SA algorithm for solving larger vehicle routing problems. The proposed approach is also suitable for other problems. It is especially suitable for discrete optimization problems. Research on applying the proposed approach to other combinatorial optimization problems should be an interesting subject.

References

Baker, B.M., Ayechew, M.A., 2003. A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, **30**(5):787-800. [doi:10.1016/S0305-0548(02)00051-5]

Bell, J.E., McMullen, P.R., 2004. Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, **18**(1):41-48. [doi:10.1016/j.aei.2004.07.001]

Bodin, L., Golden, B.L., Assad, A., Ball, M.O., 1983. The state of the art in the routing and scheduling of vehicles and crews. *Computers & Operations Research*, **10**:69-221.

Christofides, N., Mignozzi, A., Toth, P., 1981. Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxations. *Mathematical Programming*, **20**(1):255-282. [doi:10.1007/BF01589353]

Cordeau, J.F., Laporte, G., Mercier, A., 2001. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, **52**(8):928-936. [doi:10.1057/palgrave/jors/2601163]

Dantzig, G.B., Ramser, R.H., 1959. The truck dispatching problem. *Management Science*, **6**:80-91.

Eberhart, R., Kennedy, J., 1995. A New Optimizer Using Particle Swarm Theory. Proceeding of the Sixth International Symposium on Micro Machine and Human Science, p.39-43.

Gendreau, M., Hertz, A., Laporte G., 1994. A tabu search heuristic for the vehicle routing problem. *Management Science*, **40**:1276-1290.

Hasan, M., Osman, I.H., 1995. Local search algorithm for the maximal planar layout problem. *International Transactions in Operational Research*, **2**(1):89-106. [doi:10.1016/0969-6016(95)00029-7]

Kennedy, J., Eberhart, R., 1995. Particle Swarm Optimization. Proceeding of the 1995 IEEE International Conference on Neural Network, p.1942-1948.

Laporte, G., 1992. The vehicle routing problem: an overview of exact and approximate algorithms. *European Journal of Operational Research*, **59**(3):345-358. [doi:10.1016/0377-2217(92)90192-C]

Osman, I.H., 1993. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, **41**(4):421-451. [doi:10.1007/BF02023004]

Osman, I.H., Potts, C.N., 1989. Simulated annealing for permutation flow shop scheduling. *Omega*, **17**(6):551-557. [doi:10.1016/0305-0483(89)90059-5]

Salman, A., Ahmad, I., Madani, S.A., 2002. Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, **26**(8):363-371. [doi:10.1016/S0141-9331(02)00053-4]

Shi, Y., Eberhart, R., 1999. Empirical Study of Particle Swarm Optimization. Proceedings of Congress on Evolutionary Computation, p.1945-1950.

Shigenori, N., Takamu, G.J., Toshiki, Y., Yoshikazu, F., 2003. A hybrid particle swarm optimization for distribution state estimation. *IEEE Trans. on Power Systems*, **18**(1):60-68. [doi:10.1109/TPWRS.2002.807051]

Toth, P., Vigo, D., 1998. Exact Solution of the Vehicle Routing Problem. *In*: Crainic, T.G., Laporte, G.(Eds.), Fleet Management and Logistics. Kluwer Academic Publishers, Norwell, MA, p.1-31.

Toth, P., Vigo, D., 2002. The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Vol. 9. SIAM, Philadelphia, PA.

van Laarhoven, P.J.M., Aarts, E.H.L., Lenstra, J.K., 1992. Job shop scheduling by simulated annealing. *Operations Research*, **40**:113-125.

Wang, Z.Z., Cheng, J.X., Fang, H.G., Qian, F.L., 2004. A hybrid optimization algorithm solving vehicle routing problems. *Operations Research and Management Science*, **13**:48-52 (in Chinese).

Xia, W.J., Wu, Z.M., 2005. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, **48**(2):409-425. [doi:10.1016/j.cie.2005.01.018]

Yang, S.Y., Wang, M., Jiao, L.C., 2004. A Quantum Particle Swarm Optimization. Proceeding of the 2004 IEEE Congress on Evolutionary Computation, **1**:320-324.

Zhao, Y.W., Wu, B., Jiang, L., Dong, H.Z., Wang, W.L., 2004. Study of the optimizing of physical distribution routing problem based on genetic algorithm. *Computer Integrated Manufacturing Systems*, **10**(3):303-306 (in Chinese).