Expert Systems with Applications 38 (2011) 14439–14450

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques

Shyi-Ming Chen [a,b,*], Chih-Yao Chien [a]

[a] Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, ROC
[b] Graduate Institute of Educational Measurement and Statistics, National Taichung University of Education, Taichung, Taiwan, ROC

ARTICLE INFO

Keywords:
Traveling salesman problem
Genetic algorithms
Ant colony systems
Simulated annealing
Particle swarm optimization
Genetic simulated annealing ant colony system with particle swarm optimization techniques

ABSTRACT

In this paper, we present a new method, called the genetic simulated annealing ant colony system with particle swarm optimization techniques, for solving the traveling salesman problem. We also make experiments using the 25 data sets obtained from the TSPLIB (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/) and compare the experimental results of the proposed method with the methods of Angeniol, Vaubois, and Texier (1988), Somhom, Modares, and Enkawa (1997), Masutti and Castro (2009) and Pasti and Castro (2006). The experimental results show that both the average solution and the percentage deviation of the average solution to the best known solution of the proposed method are better than the methods of Angeniol et al. (1988), Somhom et al. (1997), Masutti and Castro (2009) and Pasti and Castro (2006).

1. Introduction

The traveling salesman problem (Applegate, Bixby, Chvátal, & Cook, 2007; Lawler, Lenstra, & Shmoys, 1985) is an important research topic, where a traveling salesman wants to travel all cities but each city can only be visited once. The problem is to minimize the distance of the complete route, which the salesman takes. In recent years, some methods have been presented for solving the traveling salesman problem (Angeniol, Vaubois, & Texier, 1988; Chang, Huang, & Ting, 2010; Chen & Chien, 2010; Cheng & Wang, 2009; Chien & Chen, 2009; Ellabib, Calamai, & Basir, 2007; Li, Ju, & Zhang, 2008; Liu & Zeng, 2009; Marinakis & Marinaki, 2010; Naimi & Taherinejad, 2009; Nguyen, Yoshihara, Yamamori, & Yasunaga, 2007; Saadatmand-Tarzjan, Khademi, Akbarzadeh-T., & Moghaddan, 2007; Sauer & Coelho, 2008; Shi, Wang, Zhou, & Liang, 2008; Somhom, Modares, & Enkawa, 1997; Xie & Liu, 2008; Yi, Bi, Yang, & Tang, 2008). Angeniol et al. (1988) presented a method using self-organizing feature maps to solve the traveling salesman problem. Chang et al. (2010) presented a dynamic diversity control technique in genetic algorithms for searching the un-searched solution space in the traveling salesman problem. Chien and Chen (2009) presented a method for solving the traveling salesman problem based on the parallelized genetic ant colony system. Cheng and Wang (2009) presented a genetic algorithm with a decomposition technique to solve the vehicle routing problem with time windows. Ellabib et al. (2007) presented a multiple ant colony system with exchange strategies for solving the traveling salesman problem. Li et al. (2008) presented an improved ant colony optimization method for solving the traveling salesman problem. Liu and Zeng (2009) presented a genetic algorithm with reinforcement learning to solve the traveling salesman problem. Marinakis and Marinaki (2010) presented a hybrid algorithm by combining genetic algorithms and particle swarm optimization algorithms for solving the vehicle routing problem. Naimi and Taherinejad (2009) presented an ant colony algorithm with a new interpretation of a local updating process for solving the traveling salesman problem. Nguyen et al. (2007) presented a genetic algorithm to solve the traveling salesman problem. Saadatmand-Tarzjan et al. (2008) presented a novel constructive-optimizer neural network for solving the traveling salesman problem. Sauer and Coelho (2008) presented a discrete differential evolution with a local search method to solve the traveling salesman problem. Shi et al. (2008) presented an ant colony optimization method with time windows to solve the prize-collecting traveling salesman problem. Somhom et al. (1997) presented a self-organizing model to solve the traveling salesman problem. Xie and Liu (2008) presented a multi-agent optimization system for solving the traveling salesman problem. Yi et al. (2008) presented a fast elastic net method for solving the traveling salesman problem.

Genetic algorithms (GA) (Gen & Cheng, 1997; Goldberg, 1989; Gwiazda, 2006, 2007) have been widely used in the research area of computational intelligence. Genetic algorithms are inspired by evolutionary biology. Because genetic algorithms have the global

* Corresponding author. Address: Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, 43, Section 4, Keelung Road, Taipei 106, Taiwan, ROC. Tel.: +886 2 27376417, fax: +886 2 27301081.
E-mail address: smchen@mail.ntust.edu.tw (S.-M. Chen).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.04.163

searching ability and because they can be easily implemented, they are widely used in many areas. Kaur and Murugappan (2008) presented a novel hybrid genetic algorithm for solving the traveling salesman problem. Tasgetiren, Suganthan, Pan, and Liang (2007) presented a genetic algorithm with an iterated local search capability to further improve the solution quality for solving the traveling salesman problem. Zhao, Luo, Nie, and Li (2008) presented the balancing exploration and exploitation techniques to improve the explorative capabilities and exploitation effects of genetic algorithms for solving the traveling salesman problem.

Swarm intelligence is an important research topic in the research area of computational intelligence. It consists of a population of artificial agents mimicking the animals' behavior in the real world. Each agent follows some simple rules and interacts with the others to share information to lead the behavior to convergence. The most popular algorithms in the research field of swarm intelligence are the ant colony optimization (ACO) algorithms and the particle swarm optimization (PSO) algorithms. The first ACO algorithm is the ant system (AS) proposed by Dorigo et al. (1992) to solve the traveling salesman problem. Since then, some improved methods have been proposed, such as the ant colony system (ACS) (Dorigo & Gambardella, 1997a, 1997b; Gambardella & Dorigo, 1996), the MAX–MIN ant system (MMAS) (Stützle & Hoos, 1996, 2000), the rank-based ant system (AS_{rank}) (Bullnheimer, Hartl, & Strauss, 1997) and the KCC-Ants (Naimi & Taherinejad, 2009). Kennedy and Eberhart (1995) proposed the particle swarm optimization (PSO) algorithm which is a population-based algorithm simulating the movements of birds flocks. Krohling and Coelho (2006) presented a co-evolutionary particle swarm optimization with Gaussian distribution for solving the constrained optimization problem. Lin, Chen, and Lin (2009) presented a hybrid cooperative particle swarm optimization and cultural algorithm for neural networks for dealing with prediction problems. Ting, Rao, and Loo (2006) presented a hybrid particle swarm optimization algorithm to solve the unit commitment problem.

In this paper, we propose a new method, called the genetic simulated annealing ant colony system with particle swarm optimization techniques, to solve the traveling salesman problem. First, we use the ant colony system to generate the initial solutions of the genetic algorithms. Then, we use the genetic simulated annealing techniques to generate better solutions based on the initial solutions. If the solutions searched by the genetic simulated annealing techniques are better then the initial solutions, the system will use these better solutions to feedback the pheromone information to the ant colony system. After a predefined number of cycles, the system uses particle swarm optimization techniques to exchange the pheromone information between groups. We implement the proposed method using Microsoft Visual C++ 2008 on an Intel Core-i7 PC with the 25 TSP data sets obtained from the TSPLIB (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/). The experimental results show that both the average solution and the percentage deviation of the average solution to the best known solution of the proposed method are better than Angeniol et al.'s method (1988), Somhom's method (1997), Masutti and Castro's method (2009) and Pasti and Castro's method (2006).

The rest of this paper is organized as follows. In Section 2, we briefly review the concepts of ant systems and ant colony systems from Colorni et al. (1991), Dorigo (1992), and Dorigo et al. (1992, 1996). In Section 3, we briefly review the concepts of genetic algorithms (Goldberg, 1989; Gwiazda, 2006, 2007; Holland, 1975). In Section 4, we briefly review the simulated annealing techniques (Kirkpatrick, Gelatt, & Vecchi, 1983; Laarhoven & Aarts, 1987; Martin & Otto, 1996). In Section 5, we briefly review the particle swarm optimization techniques (Kennedy & Eberhart, 1995). In Section 6, we present a new method called the genetic simulated annealing ant colony system with particle swarm optimization techniques to solve the traveling salesman problem. In Section 7, we compare the experimental results of the proposed method with Angeniol et al.'s method (1988), Somhom's method (1997), Masutti and Castro's method (2009) and Pasti and Castro's method (2006). The conclusions are discussed in Section 8.

2. Ant colony optimization

In this section, we briefly review the ant system (AS) and the ant colony system (ACS). The ant system is proposed by Colorni, Dorigo, and Maniezzo (1991), Dorigo (1992), Dorigo, Maniezzo, and Colorni (1992, 1996). It is inspired by the food-seeking behavior of ants. While the ants search for food, they will deposit the pheromone on the path where they passed. Fig. 1 (Colorni et al., 1991) shows the relationship of the pheromone versus the length of the path, where the distance of the lower path in Fig. 1 is twice longer than the distance of the upper path. Therefore, if two ants leave the nest at the same time, then the ant which chooses the upper path will get the food while the ant which chooses the lower path will only be halfway. While the ant choosing the upper path returns to the nest from the food location, the ant choosing the lower path will finally get the food after some time. In this case, the pheromone level on the upper path is twice the amount of the pheromone level on the lower path. Thus, for the other ants, the upper path is more attractive than the lower path and they will choose the upper path more easily. After a period of searching time, all the ants will choose the upper path to get the food rather than the lower path. Likewise, this searching scheme can be applied to solve the traveling salesman problem.

Assume that there are n cities, m ants and the initial pheromone on each edge is set to a very small positive constant τ_0. Each ant randomly starts at a city and visits the other cities according to the transition rule (Colorni et al., 1991). After the ants complete their routes, the system evaluates the length of the routes. Then, the system uses the pheromone update rule to update the pheromone information. The learning procedure is to update the pheromone information repeatedly. The ideas of an ant system are reviewed as follows (Colorni et al., 1991):

(1) Transition rule: While the kth ant is at city r, the next city s is selected from the unvisited cities set J_r^k according to the following equation:

$$s = \arg\max_{u \in J_r^k} \lfloor \tau(r,u) \cdot \eta(r,u)^{\beta} \rfloor, \quad \text{if} \quad q \leqslant q_0 \ \text{(Exploitation)} \tag{1}$$

or select the next city s with the probability $P_k(r, s)$,

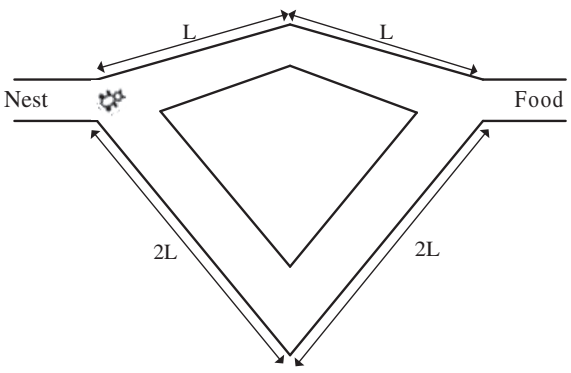

Fig. 1. The relationship between the pheromone versus the length of the path (Colorni et al., 1991).

$$P_k(r,s)=\begin{cases}\frac{\tau(r,s)\cdot\eta(r,s)^{\beta}}{\sum_{u\in J_r^k}\tau(r,u)\cdot\eta(r,u)^{\beta}}, & \text{if } s\in J_r^k,\\ 0, & \text{otherwise.}\end{cases}\quad \text{if } q>q_0 \text{ (Bias Exploitation)} \tag{2}$$

(2) Pheromone update rule:

$$\tau(r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\sum_{k=1}^{m}\Delta\tau_k(r,s) \tag{3}$$

$$\Delta\tau_k(r,s)=\begin{cases}\frac{1}{L_k}, & \text{if } (r,s)\in \text{route performed by the } k\text{th ant},\\ 0, & \text{otherwise},\end{cases} \tag{4}$$

where L_k is the length of the route completed by the kth ant.

Because the ant system requires a lot of computation, some methods have been proposed to improve the efficiency, such as the ant colony system (ACS) (Dorigo & Gambardella, 1997a, 1997b; Gambardella & Dorigo, 1996), the MAX–MIN ant system (MMAS) (Stützle & Hoos, 1996, 2000), the rank-based ant system (Bullnheimer et al., 1997) and the KCC-Ants (Naimi & Taherinejad, 2009). The ant colony system (ACS) (Dorigo & Gambardella, 1997a, 1997b; Gambardella & Dorigo, 1996) improves the ant system (AS) with two pheromone update operations. The local pheromone update operation is performed after the ants choose the next city to move; the global pheromone update operation is performed after all the ants complete their traveling routes, shown as follows:

(1) *Local update*:

$$\tau(r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\rho\cdot\tau_0, \tag{5}$$

where ρ is the pheromone evaporation parameter.

(2) Global update:

$$\tau(r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\rho\cdot\Delta\tau_k(r,s), \tag{6}$$

$$\Delta\tau_k(r,s)=\begin{cases}\frac{Q}{L_{gb}}, & \text{if } (r,s)\in \text{global best route},\\ 0, & \text{otherwise},\end{cases} \tag{7}$$

where ρ is the pheromone evaporation parameter, $0<\rho<1$, Q is a non-zero positive constant, the global best route is the route with the shortest route length which the ant system searched so far, and L_{gb} is the length of the global best route.

The MAX–MIN ant system (MMAS) (Stützle & Hoos, 1996, 2000) improves the ant system (AS) with a bounded pheromone level. Each pheromone level updated by the local pheromone update operation and the global pheromone update operation is bounded between τ_{min} and τ_{max}, shown as follows:

$$\tau(r,s)\leftarrow[(1-\rho)\cdot\tau(r,s)+\Delta\tau_k(r,s)]_{\tau_{\min}}^{\tau_{\max}}, \tag{8}$$

$$\Delta\tau^{best}(r,s)=\begin{cases}\frac{1}{L_{best}}, & \text{if } (r,s)\in \text{global best route},\\ 0 & \text{otherwise},\end{cases} \tag{9}$$

$$[x]_b^a=\begin{cases}a, & \text{if } x>a,\\ b & \text{if } x<b,\\ x, & \text{otherwise},\end{cases} \tag{10}$$

where ρ is the pheromone evaporation parameter and $\rho\in(0,1)$.

The rank-based ant system (AS_{rank}) (Bullnheimer et al., 1997) improves the ant system (AS) by ranking each ant according to the length of the route it completed. The amount of pheromone that should be deposited at an edge is weighted according to the rank order of the ants, shown as follows:

$$\tau(r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\sum_{\mu=1}^{\sigma-1}\Delta\tau^{\mu}(r,s)+\Delta\tau^{*}(r,s), \tag{11}$$

$$\Delta\tau^{\mu}(r,s)=\begin{cases}(\sigma-\mu)\cdot\frac{Q}{L_\mu}, & \text{if } (r,s)\in \text{the route performed by the } \mu\text{th ant},\\ 0, & \text{otherwise},\end{cases} \tag{12}$$

$$\Delta\tau^{*}(r,s)=\begin{cases}\sigma\cdot\frac{Q}{L^*}, & \text{if } (r,s)\in \text{best route},\\ 0, & \text{otherwise},\end{cases} \tag{13}$$

where ρ is the pheromone evaporation parameter, $\rho\in(0,1)$, σ is the number of elitist ants, μ is the ranking index, L_μ is the length of the route completed by the μth best ant, and L^* is the length of the best route.

3. Genetic algorithms

In this section, we briefly review the concepts of genetic algorithms (Holland, 1975). Genetic algorithms were inspired by the evolution process, observed in nature. The system encodes the parameters of a solution into a chromosome, where the basic element of a chromosome is the gene. The system performs four operations, i.e., selection operations, crossover operations, mutation operations and evaluation operations to search the near optimal solution. The selection operation selects those chromosomes having better fitness values. The fitness degree of a chromosome is evaluated by a predefined fitness function. The process of a genetic algorithm is reviewed as follows (Gen & Cheng, 1997; Goldberg, 1989; Gwiazda, 2006, 2007):

Step 1: /*Selection*/ The system uses the tournament selection (Gen & Cheng, 1997; Goldberg, 1989) or the roulette wheel selection (Gen & Cheng, 1997; Goldberg, 1989) to select better chromosomes into the gene pool until the gene pool is full. The roulette wheel selection selects chromosomes according to the probability P_i, shown as follows:

$$p_i=\frac{fitness(i)}{\sum_{k=0,\ldots,N-1}fitness(k)}, \tag{14}$$

where $fitness(i)$ is the fitness value of the ith chromosome and N is the size of the gene pool. The chromosomes with higher fitness values are more easily selected into the gene pool.

Step 2: /*Crossover*/ The system performs the crossover operation until the gene pool is full. If a random number generated by the system ranging between 0 and 1 is smaller than a predefined crossover rate CR, the system performs the crossover operation, where $CR\in(0,\ 1]$. The system chooses two chromosomes as parents and then reproduces their offspring after the crossover operation. After the crossover operation is done, the system puts their offspring into the gene pool. There are many kinds of crossover operation techniques, such as one-point crossover, two-points crossover, cut-and-splice, etc. (Gwiazda, 2006). Fig. 2 (Gwiazda, 2006) shows the techniques of one-point crossover, two-point crossover and cut-and-splice, respectively. The goal of the crossover operations is to spend the searching area through the recombination of chromosomes. The crossover operation continues until the gene pool is full.

Step 3: /*Mutation*/ The system performs the mutation operation after the crossover operation is performed. If a random number between 0 and 1 generated by the system is smaller than a predefined mutation rate MR, then the system performs the mutation operation, where $MR\in(0,1]$. The traditional mutation method is to randomly select a gene of the selected chromosome, and then randomly generate a new value to

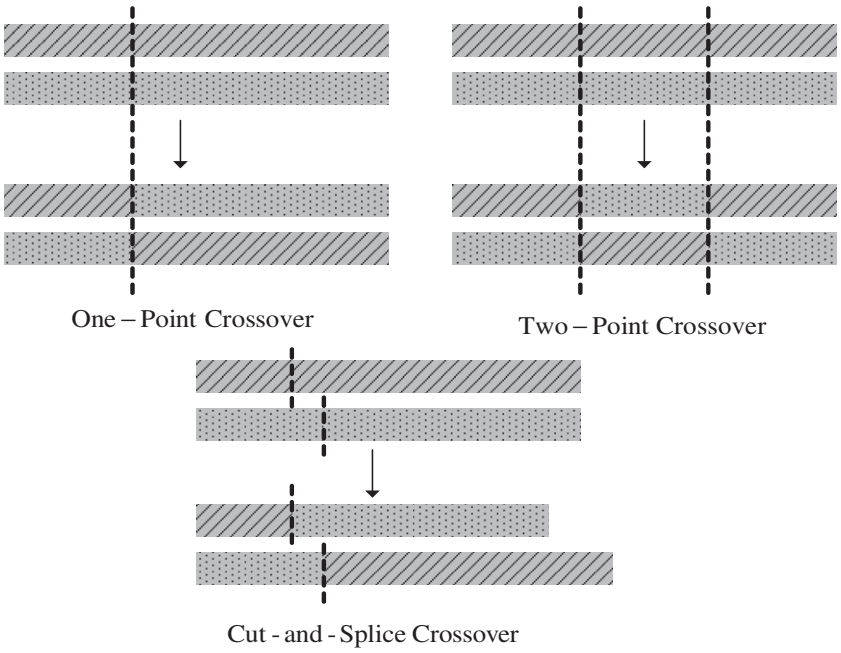

Fig. 2. The crossover operation techniques (Gwiazda, 2006).

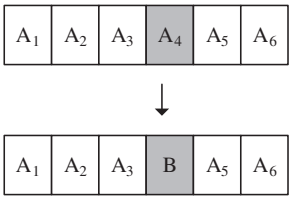

Fig. 3. The traditional mutation operation.

replace it, as shown in Fig. 3. The purpose of the mutation operation is to prevent the chromosomes in the new population to fall into a local minimum. It should be noted that not every chromosome performs the mutation operation in the evolution process. The mutation operation continues until every chromosome is certain to perform the mutation operation or not.

Step 4: /∗Evaluation∗/ After the crossover operation and the mutation operation, the system evaluates the fitness value of each chromosome based on a fitness function. If the maximum number of generations is reached or the termination condition is met, the chromosome with the highest fitness value is the best solution. Otherwise, go to **Step 1**.

4. Simulated annealing

In this section, we briefly review the simulated annealing (SA) techniques (Kirkpatrick, Gelatt, & Vecchi, 1983; Laarhoven & Aarts, 1987; Martin & Otto, 1996). The simulated annealing techniques were proposed by Kirkpatrick et al. (1983). The system simulates the annealing process of metal atoms. Metal atoms at a high temperature will become unstable from their initial states and search for other states. While cooling, the metal atoms will find an energy state that is lower than their initial state. The state changing procedure can be applied to solve real-world problems. The system generates a new state and then compares the energy of the new state with the energy of the current state. If the energy of the new state is lower than the energy of the current state, then the system accepts this state. Otherwise, the system changes to this state according to the transition probability P, shown as follows:

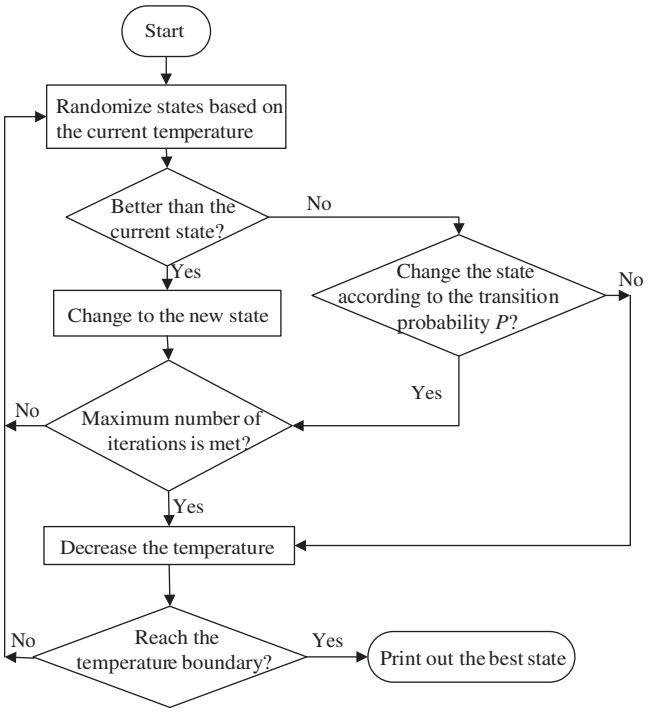

Fig. 4. The flowchart of the simulated annealing algorithm (Kirkpatrick et al., 1983).

$$P = e^{\left(\frac{-\Delta E}{kT}\right)}, \tag{15}$$

$$\Delta E = E(S') - E(S), \tag{16}$$

where k is the Boltzmann constant, T is the current temperature of the system, S is the current state, S' is the new state and E is the energy function. If the system temperature is cooling to a predefined temperature or the maximum number of iterations is met, the

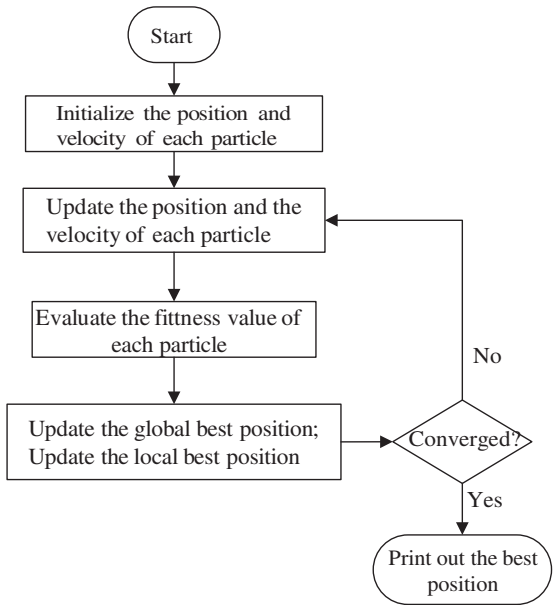

Fig. 5. The flowchart of the particle swarm optimization (Kennedy & Eberhart, 1995).

system prints out the best state, i.e., the near optimal solution. Fig. 4 shows the flowchart of the simulated annealing algorithm (Kirkpatrick et al., 1983).

5. Particle swarm optimization

In this section, we briefly review particle swarm optimization (PSO) techniques (Kennedy & Eberhart, 1995). The particle swarm optimization techniques were developed by Kennedy and Eberhart (1995). It is inspired by the social activity of flocks of animals, such as fish or birds. The system first randomly generates a group of particles whose positions represent potential solutions, and then updates the positions to the optimal one repeatedly. Each particle will follow the optimal solution and change its status, i.e., position and velocity, to get as close as possible to the optimal one. Assume that there are m particles in the D-dimensional searching space. Assume that the ith particle's position is represented as $X_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, \ldots, x_{iD})$ and assume that its moving velocity is represented as $V_i = (v_{i1}, v_{i2}, v_{i3}, v_{i4}, \ldots, v_{iD})$. The velocity of each particle is updated by the information of two best positions, i.e., the best position of the ith particle searched so far, which is called the "*pBest*", and the best position of the nearby particles searched so far, which is called the "*gBest*". The update rules of the position and the velocity of the ith particle are shown as follows (Kennedy & Eberhart, 1995):

$$V_i = V_i + C_1R_1(pBest - X_i) + C_2R_2(gBest - X_i), \quad (17)$$

$$X_i = X_i + V_i, \quad (18)$$

where C_1 and C_2 are positive constants, called acceleration coefficients, and R_1 and R_2 are random numbers, where $R_1 \in (0,1)$ and $R_2 \in (0,1)$. When the maximum number of iterations is met, the best particle so far is the near optimal solution. The flowchart of the particle swarm optimization is shown in Fig. 5 (Kennedy & Eberhart, 1995).

6. A new method for handling the traveling salesman problem

In this section, we present a new method, called the genetic simulated annealing ant colony system with particle swarm optimization techniques, for solving the traveling salesman problem. Assume that there are n cities in the traveling salesman problem. First, we use the ant colony system to generate the initial population of the genetic algorithms. Then, the system performs the genetic algorithms with simulated annealing mutation techniques to gain better solutions. For every C cycles, where C is a predefined number, the system will use the particle swarm optimization techniques to exchange the pheromone information between groups, where a cycle consists of "one execution of the ant colony system" and "some executions of the genetic algorithms with simulated annealing mutation techniques". Initially, the system generates g groups $G_0, G_1, \ldots,$ and G_g, where each group has N ants and each ant will randomly choose a city as its starting city. The initial pheromone level between any two cities is set to τ_0. The cycle counter, which will record how many times the proposed algorithm executes, is initially set to zero. The proposed method is now presented as follows:

Step 1: The kth ant of the ith group constructs its traveling sequence of cities using the following transition rule and the local pheromone update rule (Colorni et al., 1991):

(1) Transition rule: If $q \leqslant q_0$, then the kth ant of the ith group chooses to visit the next city s according to the following equations,

$$s = \arg\max_{u \in J_{k,i}(r)} [\tau_i(r,u) \cdot \eta(r,u)^{\beta}] \quad \text{(Exploitation)} \quad (19)$$

where $\tau_i(r,u)$ denotes the pheromone level between city r and city u of the ith group, $\eta(r,u)$ is the heuristic information given by the inverse of the distance from city r to city

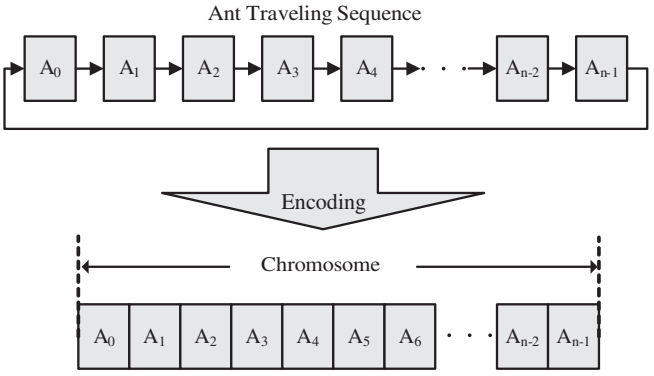

Fig. 6. Encode the traveling sequence of cities searched by the ant colony system into a chromosome.

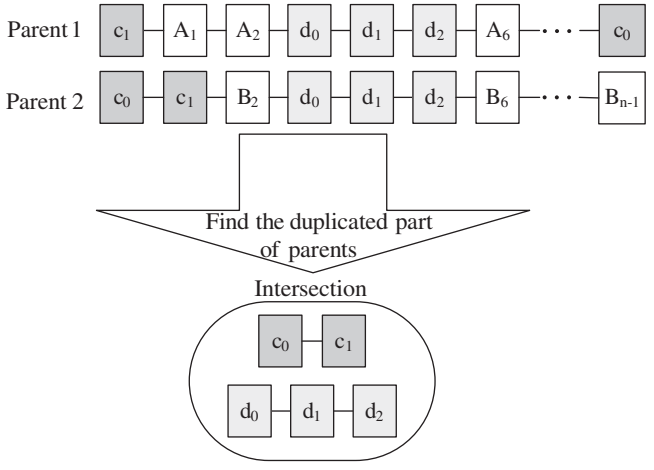

Fig. 7. The procedure of searching intersection of parents.

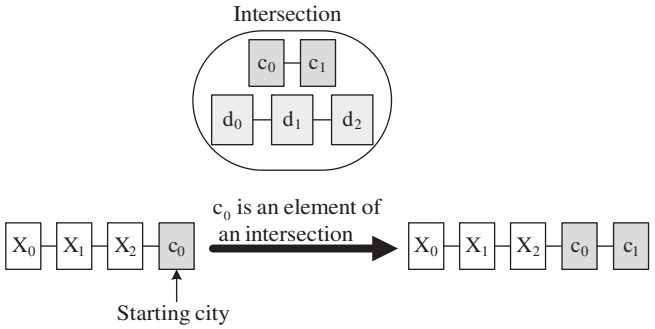

Fig. 8. Principle 1.

u, β is the parameter which controls the relative importance of the pheromone versus the heuristic information $\eta(r, u)$, q_0 is a predefined number to control the ant to choose the exploitation or the bias exploitation, $q_0 \in (0, 1)$, q is a random number and $q \in (0, 1)$. Otherwise, if $q > q_0$, then visit the next city s with the transition probability $P_{k,i}(r, s)$,

$$P_{k,i}(r,s) = \begin{cases} \frac{\tau(r,s)\cdot\eta(r,s)^{\beta}}{\sum_{u\in J_{k,i}(r)}\tau(r,u)\cdot\eta(r,u)^{\beta}}, & \text{if } s \in J_{k,i}(r), \\ 0, & \text{otherwise}, \end{cases} \quad \text{(Bias Exploitation)} \tag{20}$$

where $\tau_i(r, s)$ denotes the pheromone level between city r and city s of the ith group, $\eta(r, s)$ is the heuristic information given by the inverse of the distance from city r to city s, $J_{k,\,i}(r)$ denotes the set of cities not yet visited by the kth ant of the ith group, β is the parameter which controls the relative importance of the pheromone versus the heuristic information $\eta(r, s)$, q_0 is a predefined number to control the ant to either choose the exploitation or the bias exploitation, $q_0 \in (0, 1)$, q is a random number and $q \in (0, 1)$.

(2) Local pheromone update rule:

$$\tau_i(r,s) \leftarrow (1-\rho)\times\tau_i(r,s) + \rho\times\tau_0, \tag{21}$$

$$\text{if } \tau_i(r,s) < \tau^i_{\min}, \text{then let } \tau_i(r,s) = \tau^i_{\min}, \tag{22}$$

where ρ denotes the pheromone evaporation parameter, $\rho \in (0, 1)$, $\tau_i(r, s)$ denotes the pheromone level between city r and city s of the ith group, and $\tau^i_{\min}$ denotes the lower bound of the pheromone level in the ith group (Stützle & Hoos, 1996, 2000).

Step 2: Evaluate the distance of the routes completed by the ants in each group. Perform the global pheromone update operation (Stützle & Hoos, 1996, 2000), shown as follows, to update the pheromone level between cities for each group:

$$\tau_i(r,s) \leftarrow (1-\rho)\times\tau_i(r,s) + \rho\times\Delta\tau_i(r,s), \tag{23}$$

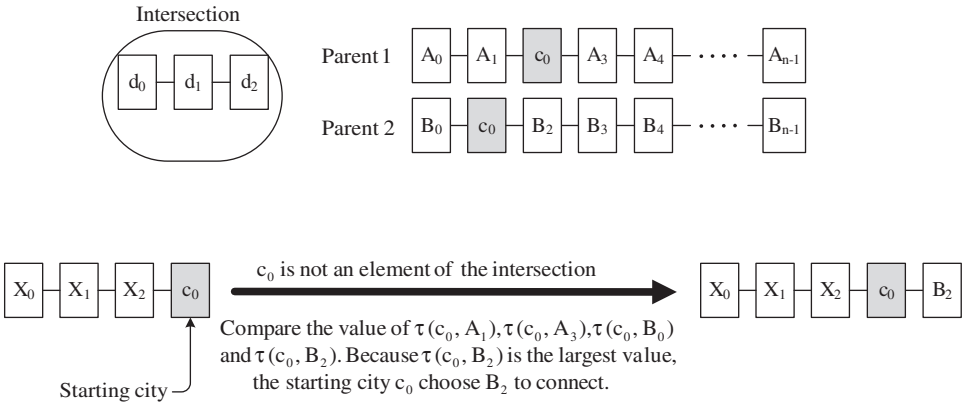

Fig. 9. Principle 2.

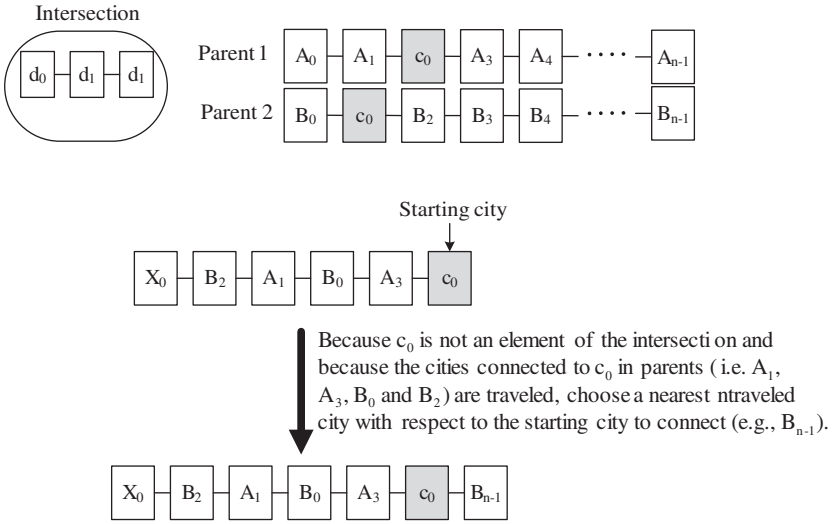

Fig. 10. Principle 3.

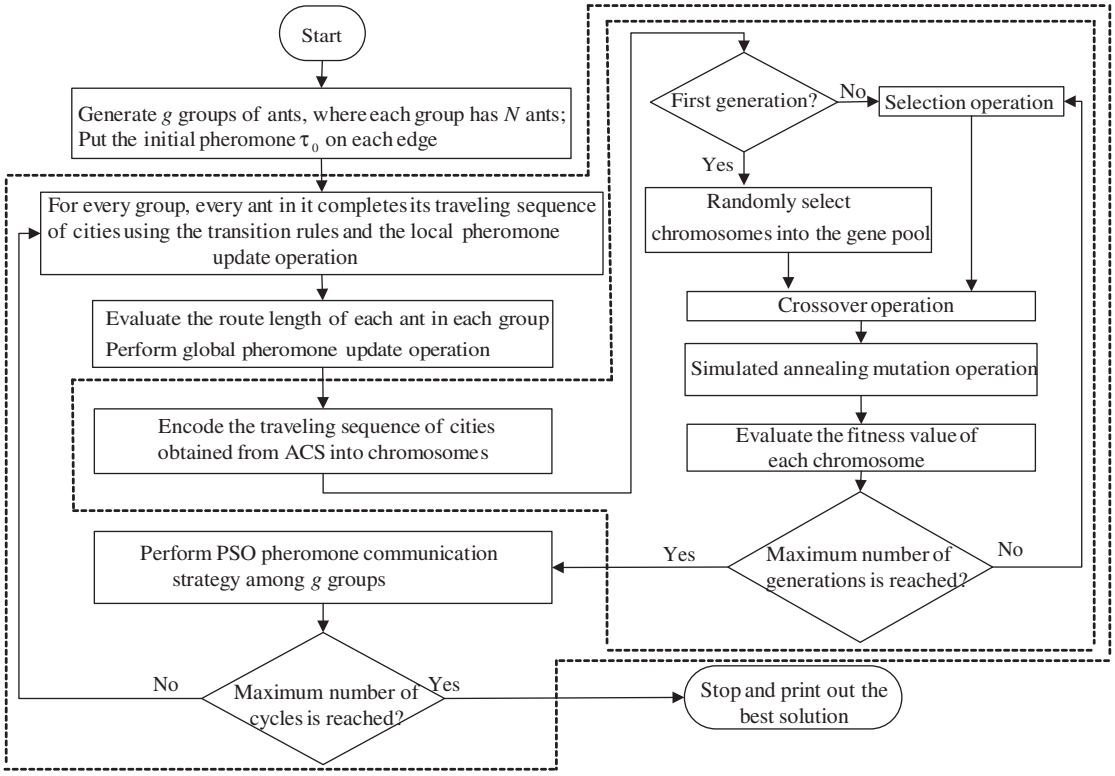

Fig. 11. The flowchart of the proposed method.

$$\Delta\tau_i(r,s)=\begin{cases}\frac{1}{L_{group\_i\_best}}, & \text{if } (r,s)\in \text{the best route of the } i\text{th group},\\ 0, & \text{otherwise},\end{cases} \quad (24)$$

$$\text{if } \tau_i(r,s)>\tau^i_{\max}, \text{ then let } \tau_i(r,s)=\tau^i_{\max},$$

where ρ denotes the pheromone evaporation parameter, $\rho \in (0,1)$, $\tau_i(r,s)$ denotes the pheromone level between city r and city s of the ith group, $\Delta\tau_i(r,s)$ denotes the quantity of pheromone per unit length laid on the edge formed by the route from city r to city s, $L_{group\_i\_best}$ denotes the shortest route distance among the g groups, and $\tau^i_{\max}$ denotes the upper bound of the pheromone level in the ith group. (Note: After performing Steps 1 and 2, each ant of the ith group has a traveling sequence of cities. That is, the system has g groups of the traveling sequences of cities and each group has N traveling sequences of cities.)

Step 3: Encode the traveling sequences of cities searched by the ant colony system into chromosomes. Let each city be a gene. The encoding process is shown in Fig. 6. Let the traveling sequences of cities searched by the ant colony system be the initial population of the genetic algorithms.

Step 4: /* Selection*/ For the ith group, the system uses the roulette wheel selection method (Gen & Cheng, 1997; Goldberg, 1989) to select x chromosomes from the ith group and y chromosomes from groups G_0, G_1, ..., and G_g into the gene pool (Chien & Chen, 2009). Therefore, the gene pool consists of $x+y$ chromosomes. The chromosomes with shorter route distances are more easily selected into the gene pool. Then, the chromosomes in the gene pool perform the crossover operation to reproduce the new population.

Step 5: /* Crossover */ For the ith group, randomly select two chromosomes from the gene pool as parents to perform the crossover operation if a random number generated by the system is smaller than or equal to a predefined crossover rate CR. The crossover operation consists of two strategies, where the system chooses the crossover strategy based on the parameter R_0, where R_0 is a parameter to decide which crossover strategy the system chooses and $R_0 \in (0,1)$, shown as follows (Chien & Chen, 2009):

Strategy 1: If the random number r generated by the system is larger than or equal to R_0, the system performs the *Bone-Crossover*. The system searches the intersection of parents, which is the duplicated gene sequence of the parents, as shown in Fig. 7. The system chooses the longest intersection to be the base and uses the following three principles to reproduce offspring:

Table 1
Parameter settings of the proposed method.

The number of groups (g)	4
Number of ants in each group	30
Parameter β for distance influence control	2
Pheromone evaporation parameter ρ	0.1
q_0	0.9
Parameter R_0 for crossover strategy	0.33
The number of GA generations	100
Crossover rate CR	1
Route mutation rate RMR	0.3
Pheromone mutation rate PMR	0.2
Starting temperature T_0 for simulated annealing	100
Cycles for pheromone communication (C)	30
Upper bound pheromone τ_{max}	$\frac{1}{(1-\rho)\times L_{iteration\_best}}$ (Stützle & Hoos, 1996, 2000) ($L_{iteration\_best}$ is the shortest tour length of the ith iteration)
Lower bound pheromone τ_{min}	$\frac{\tau_{max}}{20}$ (Stützle & Hoos, 1996, 2000)

Table 2
A comparison of the experimental results of the proposed method with Angeniol's method (1988), Somhom et al.'s method (1997), Pasti and Castro's method (2006) and Masutti and Castro's method (2009), where BKS represents the best known solution, Mean represents the average solution, SD represents the standard deviation, and Best represents the best solution found.

TSP instance	BKS	Angeniol's method (1988)			Somhom et al.'s method (1997)			Pasti and Castro's method (2006)			Masutti and Castro's method (2009)			The proposed method		
		Mean	SD	Best	Mean	SD	Best	Mean	SD	Best	Mean	SD	Best	Mean	SD	Best
ncit64	6400	6594.07	72.97	6482	6547.60	66.02	6482	6400.00	0.00	6400	6400.00	0.00	6400	**6400.00**	0.00	**6400**
eil51	426	442.90	4.59	432	440.57	3.44	433	438.70	3.52	429	437.47	4.20	427	**427.27**	0.45	**427**
eil76	538	563.20	6.45	554	562.27	5.23	552	556.10	8.03	542	556.33	5.30	541	**540.20**	2.94	**538**
eil101	629	665.93	7.27	655	655.57	5.95	640	654.83	6.57	641	648.63	3.85	638	**635.23**	3.59	**630**
berlin52	7542	8363.70	241.82	7778	8025.07	248.79	7715	8073.97	270.14	7716	7932.50	277.25	7542	**7542.00**	0.00	**7542**
bier127	118,282	128920.33	6563.13	120,110	121733.33	1240.04	119,840	121780.33	1564.02	118,760	120886.33	1158.79	118,970	**119421.83**	580.83	**118,282**
ch130	6110	6416.80	92.97	6265	6307.23	62.95	6203	6291.77	64.98	6142	6282.40	60.15	6145	**6205.63**	43.70	**6141**
ch150	6528	6842.80	105.97	6634	6751	62.18	6631	6753.20	83.01	6629	6738.37	76.14	6602	**6563.70**	22.45	**6528**
rd100	7910	8444.50	144.26	8088	8239.40	103.91	8028	8253.93	148.61	7947	8199.77	80.77	7982	**7987.57**	62.06	**7910**
lin105	14,379	16111.37	609.10	14,999	14475.60	118.24	14,379	14702.23	328.37	14,379	14400.17	44.03	14,379	14406.37	37.28	**14,379**
lin318	42,029	45832.83	541.67	44,869	43922.90	382.29	43,154	43704.97	391.48	42,975	43696.87	410.06	42,834	**43002.90**	307.51	**42,487**
kroA100	21,282	24678.80	862.09	23,009	21616.77	164.21	21,410	21868.47	245.76	21,369	21522.73	93.34	21,333	**21370.47**	123.36	**21,282**
kroA150	26,524	29960.90	564.39	28,948	27401.33	252.03	26,930	27346.43	223.31	26,932	27355.97	327.91	26,678	**26899.20**	133.02	**26,524**
kroA200	29,368	33228.33	638.63	31,669	30415.67	132.86	30,144	30257.53	342.98	29,594	30190.27	273.38	29,600	**29738.73**	356.07	**29,383**
kroB100	22,141	25966.40	713.12	24,026	22622.50	75.33	22,548	22853.60	208.56	22,596	22661.47	193.47	22,343	**22282.87**	183.99	**22,141**
kroB150	26,130	29404.53	730.02	27,886	26806.33	250.14	26,342	26752.13	210.48	26,395	26631.87	232.86	26,264	**26448.33**	266.76	**26,130**
kroB200	29,437	33838.13	662.85	32,351	30286.47	301.23	29,703	30415.60	349.50	29,831	30135.00	276.78	29,637	**30035.23**	357.48	**29,541**
kroC100	20,749	23496.13	844.30	22,344	21149.87	187.99	20,921	21231.60	215.92	20,915	20971.23	108.16	20,915	**20878.97**	158.64	**20,749**
kroD100	21,294	23909.03	544.06	23,076	21845.73	154.30	21,500	22027.87	303.22	21,457	21697.37	157.03	21,374	**21620.47**	226.60	**21,309**
kroE100	22,068	24828.03	699.55	23,642	22682.47	214.05	22,379	22815.50	268.37	22,427	22715.63	260.16	22,395	**22183.47**	103.32	**22,068**
rat575	6773	8301.83	106.88	8107	7173.63	39.47	7090	7125.07	40.91	7039	7115.67	37.47	7047	**6933.87**	27.62	**6891**
rat783	8806	10721.60	108.81	10,532	9387.57	39.44	9316	9326.30	48.45	9185	9343.77	46.97	9246	**9079.23**	52.69	**8988**
rl1323	270,199	301424.33	2891.50	293,350	300899.00	2717.10	295,780	300286.00	2678.56	295,060	305314.33	2315.81	300,770	**280181.47**	1761.66	**277,642**
fl1400	20,127	21174.67	247.33	20,649	20742.60	115.80	20,558	21070.57	205.16	20,745	21110.00	163.29	20,851	21349.63	527.88	20,593
d1655	62,128	71168.07	587.19	68,875	68046.37	379.33	67,459	71431.70	589.16	70,323	72113.17	698.57	70,918	**65621.13**	1031.94	**64,151**

Principle 1: If the starting city is an element of an intersection, then connect it with the base and remove it from the intersection set, shown in Fig. 8.
Principle 2: If the starting city is not an element of an intersection, then find the connected cities among the parents and choose the city with a higher pheromone level to connect, shown in Fig. 9.
Principle 3: If the cities connected to the starting city are already visited, then choose the nearest city to connect, shown in Fig. 10.

Strategy 2: If the random number r generated by the system is smaller than R_0, then the system performs the two-point crossover operation. The system randomly generates two crossover points. Then, exchange the genes between the crossover points.
Step 6: /∗ Simulated Annealing Mutation ∗/ Set the system's initial temperature to T_0. For the ith group, randomly select a chromosome S. If a random number between 0 and 1 generated by the system is smaller than a predefined route mutation rate RMR, then the system uses the simulated annealing technique to perform the mutation operation with the selected chromosome.

Table 3
A comparison of the experimental results of the proposed method with Somhom et al.'s method (1997), Aras et al.'s method (1999), Cochrane and Beasley's method (2003), and Masutti and Castro's method (2009), where BKS represents the best known solution, PDav represents the percentage deviation of the average solution in relation to the best known solution, PDbest represents the percentage deviation of the best solution found in relation to the best known solution.

TSP instance	BKS	Somhom et al.'s method (1997)		Aras et al.'s method (1999)		Cochrane and Beasley's method (2003)		Masutti and Castro's method (2009)		The proposed method	
		PDav	PDbest	PDav	PDbest	PDav	PDbest	PDav	PDbest	PDav	PDbest
eil51	426	3.43	1.64	N/A	2.86	2.89	0.94	2.69	0.23	**0.30**	**0.23**
eil76	538	5.46	1.49	N/A	4.98	4.35	2.04	3.41	0.56	**0.41**	**0.00**
eil101	629	4.17	1.27	N/A	4.65	3.78	1.11	3.12	1.43	**0.99**	**0.16**
berlin52	7542	5.81	0.00	N/A	N/A	7.01	0.00	5.18	0.00	**0.00**	**0.00**
bier127	118,282	3.41	0.84	N/A	N/A	3.00	0.69	2.20	0.58	**0.96**	**0.00**
ch130	6110	2.82	1.02	N/A	N/A	2.82	1.13	2.82	0.57	**1.57**	**0.51**
ch150	6528	3.91	1.76	N/A	N/A	3.23	1.78	3.22	1.13	**0.55**	**0.00**
rd100	7910	3.99	1.28	N/A	2.09	3.64	1.19	3.66	0.91	**0.98**	**0.00**
lin105	14,379	2.75	0.62	N/A	1.98	1.08	0.00	0.15	0.00	**0.19**	**0.00**
lin318	42,029	4.16	2.17	N/A	N/A	4.31	2.65	3.97	1.92	**2.32**	**1.09**
kroA100	21,282	2.62	0.31	N/A	N/A	1.31	0.57	1.13	0.24	**0.42**	**0.00**
kroA150	26,524	4.61	1.47	N/A	N/A	3.06	1.55	3.14	0.58	**1.41**	**0.00**
kroA200	29,368	3.70	0.86	N/A	5.72	3.27	0.92	2.80	0.79	**1.26**	**0.05**
kroB100	22,141	2.91	1.43	N/A	N/A	2.20	1.53	2.35	0.91	**0.64**	**0.00**
kroB150	26,130	2.82	1.67	N/A	N/A	2.60	1.06	1.92	0.51	**1.22**	**0.00**
kroB200	29,437	4.83	1.55	N/A	N/A	2.31	0.88	2.37	0.68	**2.03**	**0.35**
kroC100	20,749	3.18	0.85	N/A	N/A	1.70	0.80	1.07	0.80	**0.63**	**0.00**
kroD100	21,294	2.28	0.89	N/A	N/A	1.87	0.80	1.89	0.38	**1.53**	**0.07**
kroE100	22,068	3.35	1.71	N/A	N/A	2.56	1.52	2.93	1.48	**0.52**	**0.00**
rat575	6773	N/A	N/A	N/A	N/A	6.20	4.89	5.06	4.05	**2.38**	**1.74**
rat783	8806	N/A	N/A	N/A	N/A	6.57	5.66	6.11	5.00	**3.10**	**2.07**
rl1323	270,199	N/A	N/A	N/A	N/A	11.85	11.29	13.00	11.31	**3.69**	**2.75**
fl1400	20,127	2.78	1.68	N/A	N/A	4.26	2.12	4.88	3.60	6.07	2.32
d1655	62,128	N/A	N/A	N/A	N/A	10.09	9.10	16.07	14.15	**5.62**	**3.26**

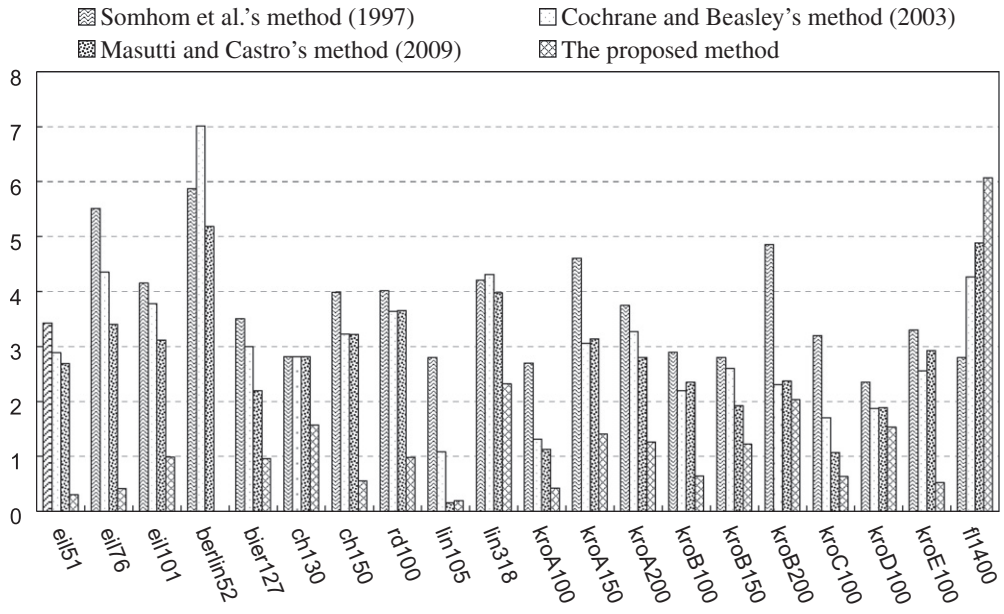

Fig. 12. Percentage deviations of the average solution to the best known solution of each TSP dataset for different methods.

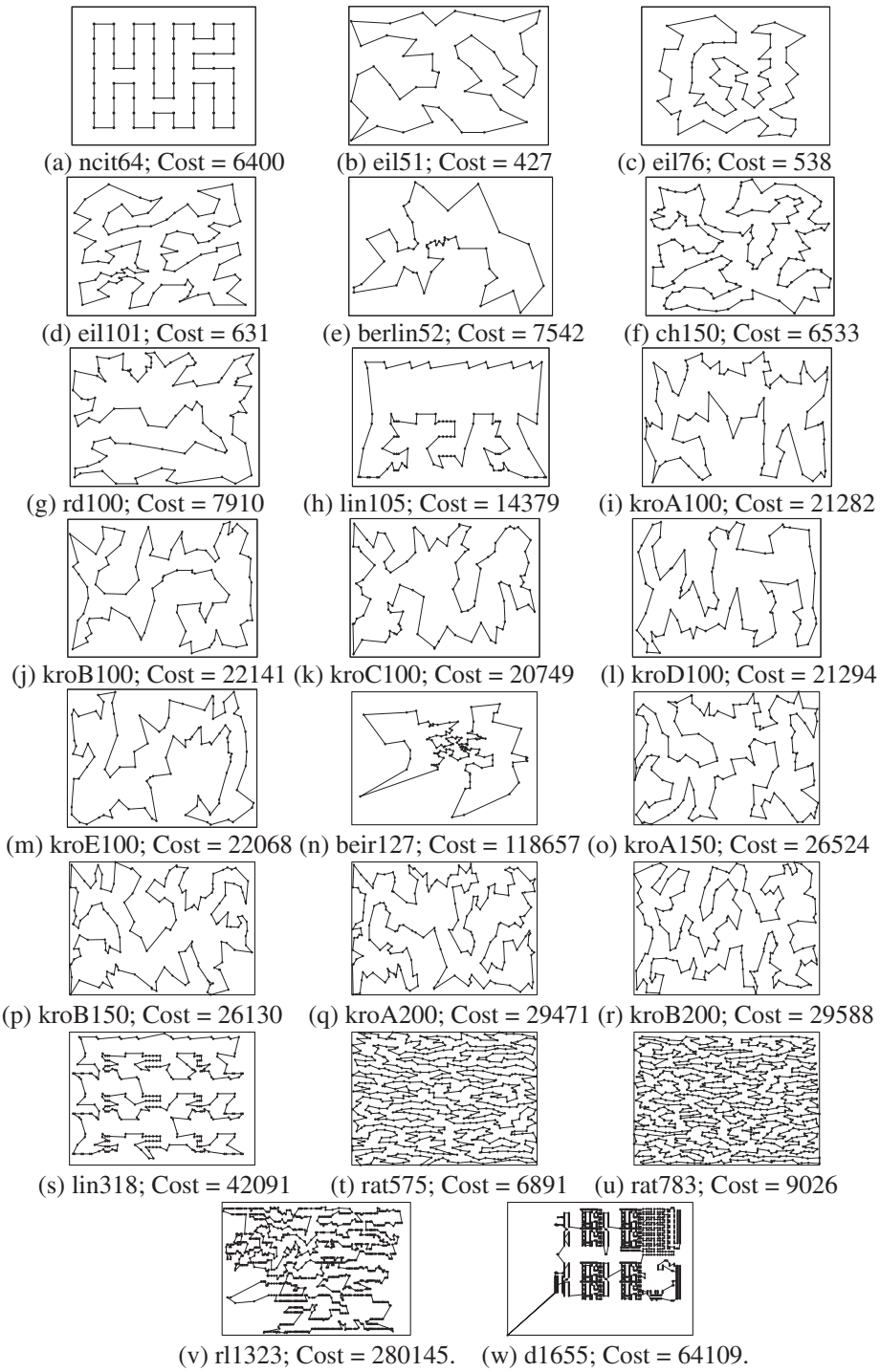

Fig. 13. Best routes found by the proposed method: (a) ncit64, (b) eil51, (c) eil76, (d) eil101, (e) berlin52, (f) ch150, (g) rd100, (h) lon105, (i) kroA100, (j) kroB100, (k) kroC100, (l) kroD100, (m) kroE100, (n) ber127, (o) kroA150, (p) kroB150, (q) kroA200, (r) kroB200, (s) lin318, (t) rat575, (u) rat 783, (v) rl1323, (w) d1655.

The system randomly selects a gene in the chromosome S to mutate to get a new chromosome S' and evaluates the energy of the chromosome S'. If the energy of the chromosome S' is lower than the energy of the chromosome S or a random number between 0 and 1 generated by the system is smaller than the probability $P = \exp^{\frac{-\Delta E}{kT}}$, where $\Delta E = Energy(S') - Energy(S)$, k is the Boltzmann constant and T is the system's current temperature, then the system puts the chromosome S' into the gene pool. Otherwise, the system leaves the chromosome S unchanged and puts it back into the gene pool. If a random number between 0 and 1 generated by the system is smaller than a predefined pheromone mutation rate *PMR*, then the system randomly selects one edge and changes the pheromone level on it into a random number between $\tau^i_{\min}$ and $\tau^i_{\max}$, where an edge is formed by two adjacent cities in the traveling sequence of cities, and $\tau^i_{\min}$ and $\tau^i_{\max}$ denote the lower bound and the upper bound of the pheromone level in the ith group, respectively. This simulated annealing mutation procedure will continuously process until T_0 is annealing to a predefined temperature, where T_0 is the initial temperature of the simulated annealing mutation process.

Step 7: Evaluate the fitness values of the offspring for each group. The fitness value is defined as the total distance of the

route. If the maximum number of generations is reached, then go to **Step 8**. Otherwise, go to **Step 4.**

Step 8: /* PSO Pheromone Communication Strategy*/ For every C cycles, the system uses particle swarm optimization techniques to exchange the pheromone experience among g groups, shown as follows:

$$\tau_i(r,s) \leftarrow \tau_i(r,s) + v, \tag{25}$$

$$v = 2R_1(\tau_{in}(r,s) - \tau_i(r,s)) + 2R_2(\tau_{gb}(r,s) - \tau_i(r,s)), \tag{26}$$

where R_1 and R_2 are random numbers between 0 and 1 generated by the system, respectively, $\tau_i(r, s)$ denotes the pheromone level between city r and city s in the ith group and gb denotes the index of the best group, which is the group that owns the best solution, i and in are the indices of the groups, and $i \neq in \neq gb$.

Step 9: If the termination condition or the predefined maximum number of cycles is reached, then print out the best route and its length and stop. Otherwise, go to **Step 1**.

Fig. 11 shows the flowchart of the proposed method.

7. Experimental results

In order to test its performance, we have implemented the proposed method using Microsoft Visual C++ 2008 on an Intel Core-i7 PC with 25 datasets, obtained from TSPLIB (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/). In this paper, according to TSPLIB, the distance between any two cities is calculated by the Euclidian distance and then rounded off after the decimal point. We also compare the experimental results of the proposed method with the ones presented in (Angeniol, Vaubois, & Texier, 1988; Masutti & Castro, 2009; Pasti & Castro, 2006; Somhom et al., 1997). In this paper, the implemented program runs 1000 cycles for each dataset, where each cycle consists of one run of the ant the colony system and 100 generations of the genetic algorithm. Table 1 shows the parameter settings of the proposed method.

Table 2 makes a comparison of the experimental results of the proposed method with Angeniol's method (1988), Somhom et al.'s method (1997), Pasti and Castro's method (2006) and Masutti and Castro's method (2009) with respect to the best solutions and the average solutions for 30 independent runs. In Table 2, the best results are emphasized in bold. We can see that the experimental results of the proposed method, using these 25 data sets, are better than the results of Angeniol's method (1988), Somhom et al.'s method (1997), Pasti and Castro's method (2006) and Masutti and Castro's method (2009). For the data sets eil76, rd100, kroA100, kroA150, kroB100, kroB150, kroC100, kroD100, and kroE100 shown in Table 2, we also can see that Angeniol's method (1988), Somhom et al.'s method (1997), Pasti and Castro's method (2006) and Martin and Otto's method (1996) cannot find the best solutions shown in the TSPLIB Web Site (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/), whereas the proposed method can find the best known solutions.

Table 3 makes a comparison of the experimental results of the percentage deviations of the average solution and the best solution to the best known solution shown in the TSPLIB Web Site (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/) for different methods, where the percentage deviation of the average solution to the best known solution *PDav* and the percentage deviation of the best found solution to the best known solution *PDbest* are defined as follows:

$$PDav = \frac{\text{average solution} - \text{best known solution}}{\text{best known solution}} \times 100,$$

$$PDbest = \frac{\text{best solution} - \text{best known solution}}{\text{best known solution}} \times 100.$$

From Table 3, we can see that the percentage deviations of the average solution and the percentage deviations of the best solution of the proposed method are better than Somhom et al.'s method (1997), Aras, Oommen, Chvatal, and Cook's method (1999), Cochrane and Beasley's method (2003), and Masutti and Castro's method (2009) for the data sets eil51, eil76, eil101, berlin52, bier127, ch150, rd100, lin105, lin318, kroA100, kroA150, kroA200, kroB100, kroB150, kroB200, kroC100, kroD100, kroE100, rat575, rat783, rl1323 and d1655.

Fig. 12 makes a comparison of the experimental results of the percentage deviations of the average solution to the best known solution shown in the TSPLIB Web Site (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/) for different methods. From Fig. 12, we can see that the proposed method gets smaller percentage deviations than the methods presented in Somhom, Modares, and Enkawa (1997), Cochrane and Beasley (2003) and Masutti and Castro (2009). Fig. 13 shows some of the best solutions searched by the proposed method and their route lengths.

8. Conclusions

In this paper, we have presented a new method, called the genetic simulated annealing ant colony system with particle swarm optimization techniques, to solve the traveling salesman problem. It combines the ant colony system, simulated annealing, genetic algorithms and the particle swarm techniques to solve the traveling salesman problem. We also have made experiments using the 25 data sets obtained from the TSPLIB (http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/) and have compared the experimental results of the proposed method with the methods of Angeniol (1988), Somhom, Modares, and Enkawa (1997), Pasti and Castro (2006) and Masutti and Castro (2009). The experimental results show that both the average solution and the percentage deviation of the average solution to the best known solution of the proposed method are better than Angeniol et al.'s method (1988), Somhom et al.'s method (1997), Pasti and Castro's method (2006) and Masutti and Castro's method (2009). From Fig. 12, we can see that the proposed method gets smaller percentage deviations than Cochrane and Beaskley's method (2003), Somhom et al.'s method (1997) and Masutti and Castro's method (2009).

Acknowledgement

This work was supported in part by the National Science Council, Republic of China, under Grant NSC 97-2221-E-011-107-MY3.

References

Angeniol, B., Vaubois, G. D. L. C., & Texier, J. Y. L. (1988). Self-organizing feature maps and the traveling salesman problem. *Neural Networks, 1*(4), 289–293.

Applegate, D. L., Bixby, R. E., Chvátal, V., & Cook, W. J. (2007). *Traveling salesman problem: A computational study*. New Jersey: Princeton University Press.

Aras, N., Oommen, B. J., Chvatal, V., & Cook, W. (1999). The Kohonen network incorporating explicit statistics and its application to the traveling salesman problem. *Neural Networks, 12*(9), 1273–1284.

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1997). A new rank based version of the ant system – A computational study. *Central European Journal for Operations Research and Economics, 7*(1), 25–38.

Chang, P. C., Huang, W. S., & Ting, C. J. (2010). Dynamic diversity control in genetic algorithm for mining unsearched solution space in TSP problems. *Expert Systems with Applications, 37*(3), 1863–1878.

Chen, S. M., & Chien, C. Y. (2010). A new method for solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. In *Proceedings of the 2010 international conference on machine learning and cybernetics, Qingdao, Shandong, China* (pp. 2477–2482).

Chien, C. Y., & Chen, S. M. (2009). A new method for handling the traveling salesman problem based on parallelized genetic colony systems. In *Proceedings of the 2009*

international conference on machine learning and cybernetics, Boading, Hebei, China (pp. 2828–2833).
Cheng, C. B., & Wang, K. P. (2009). Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm. *Expert Systems with Applications, 36*(4), 7758–7763.
Cochrane, E. M., & Beasley, J. E. (2003). The co-adaptive neural network approach to the Euclidean traveling salesman problem. *Neural Networks, 16*(10), 1499–1525.
Colorni, A., Dorigo, M., & Maniezzo, V. (1991). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life, Paris, France* (pp. 134–142).
Dorigo, M. (1992). *Learning and nature algorithm*. Ph.D. dissertation, Dipartmento di Elecctonica, Polotecnico di Milano, Italy (in Italian).
Dorigo, M., Maniezzo, V., & Colorni, A. (1992). *Positive feedback as a search strategy*. Tech. rep. 99-016, Dipartimento di Elecctonica, Polotecnico di Milano, Italy.
Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on System, Man, and Cybernetics Part-B: Cybernetics, 26*(1), 29–41.
Dorigo, M., & Gambardella, L. M. (1997a). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation, 1*(1), 53–66.
Dorigo, M., & Gambardella, L. M. (1997b). Ant colonies for the traveling salesman problem. *Biosystems, 34*(2), 73–81.
Ellabib, I., Calamai, P., & Basir, O. (2007). Exchange strategies for multiple ant colony system. *Information Sciences, 177*(5), 1248–1264.
Gambardella, L. M., & Dorigo, M. (1996). Solving symmetric and asymmetric TSPs by ant colonies. In *Proceedings of 1996 IEEE international conference on evolutionary computation, Nagoya, Japan* (pp. 622–627).
Gen, M., & Cheng, R. (1997). *Genetic algorithms and engineering design*. NY: John Wiley & Sons.
Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Boston, MA: Addison-Wesley.
Gwiazda, T. D. (2006). *Genetic algorithms reference*. *Crossover for single-objective numerical optimization problems* (Vol. I). Berlin: Springer.
Gwiazda, T. D. (2007). *Genetic algorithms reference*. *Mutation operator for numerical optimization problems* (Vol. II). Lomianki, Poland: Tomasz Gwiazda.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press.
Kaur, D., & Murugappan, M. M. (2008). Performance enhancement in solving traveling salesman problem using hybrid genetic algorithm. In *Proceedings of the 2008 annual meeting of the North American fuzzy information processing society, New York* (pp. 1–6).
Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the 1995 IEEE international conference on neural networks, Piscataway, New Jersey* (Vol. 4, pp. 1942–1948).
Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671–680.
Krohling, R. A., & Coelho, L. D. S. (2006). Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, 36*(6), 1407–1416.
Lawler, E. L., Lenstra, J. K., & Shmoys, D. B. (1985). The traveling salesman problem: A guided tour of combinatorial optimization. *Series in discrete mathematics & optimization*. Chichester: Wiley.
Laarhoven, P. J. M. V., & Aarts, E. H. L. (1987). *Simulated annealing: Theory and applications*. Netherland: Kluwer Academic.
Li, L., Ju, S., & Zhang, Y. (2008). Improved ant colony optimization for the traveling salesman problem. In *Proceedings of the 2008 international conference on intelligent computation technology and automation* (Vol. 1, pp. 76–80).
Liu, F., & Zeng, G. (2009). Study of genetic algorithm with reinforcement learning to solve the TSP. *Expert Systems with Applications, 36*(3), 6995–7001.
Lin, C. J., Chen, C. H., & Lin, C. T. (2009). A hybrid of cooperative particle swarm optimization and cultural algorithm for neural fuzzy networks and its prediction applications. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, 39*(1), 55–68.
Marinakis, Y., & Marinaki, M. (2010). A hybrid genetic – Particle swarm optimization algorithm for solving the vehicle routing problem. *Expert Systems with Applications, 37*(2), 1446–1455.
Martin, O. C., & Otto, S. W. (1996). Combining simulated annealing with local search heuristics. *Annals of Operations Research, 63*(1), 57–75.
Masutti, T. A. S., & Castro, L. N. D. (2009). A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem. *Information Sciences, 179*(10), 1454–1468.
Naimi, H. M., & Taherinejad, N. (2009). New robust and efficient ant colony algorithms: Using new interpretation of local updating process. *Expert Systems with Applications, 36*(1), 481–488.
Nguyen, H. D., Yoshihara, I., Yamamori, K., & Yasunaga, M. (2007). Implementation of an effective hybrid GA for large-scale traveling salesman problem. *IEEE Transactions on System, Man, and Cybernetics – Part B, 37*(1), 92–99.
Pasti, R., & Castro, L. N. D. (2006). A neuro-immune network for solving the traveling salesman problem. In *Proceedings of 2006 international joint conference on neural networks, Vancouver, BC, Canada* (pp. 3760–3766).
Saadatmand-Tarzjan, M., Khademi, M., Akbarzadeh-T., M.-R., & Moghaddan, H. A. (2007). A novel constructive-optimizer neural network for the traveling salesman problem. *IEEE Transactions on Systems, Man, Cybernetics – Part B: Cybernetics, 37*(4), 754–770.
Sauer, J. G., & Coelho, L. (2008). Discrete differential evolution with local search to solve the traveling salesman problem: Fundamentals and case studies. In *Proceedings of the 7th IEEE international conference on cybernetic intelligence systems, London, England* (pp. 1–6).
Shi, X., Wang, L., Zhou, Y., & Liang, Y. (2008). An ant colony optimization method for prize-collecting traveling salesman problem with time windows. In *Proceedings of the fourth international conference on natural computation, Jinan, China* (Vol. 7, pp. 480–484).
Somhom, S., Modares, A., & Enkawa, T. (1997). A self-organizing model for the traveling salesman problem. *Journal of the Operational Research Society, 48*(4–6), 919–928.
Stützle, T. & Hoos, H. H. (1996). *Improving the ant system: A detail report on the MAX–MIN ant system*. Tech. rep. AIDA-96-12, Darmstadt University of Technology, Computer Science Department, Intellectics Group.
Stützle, T., & Hoos, H. H. (2000). MAX–MIN ant system. *Future Generation Computer Systems, 16*(8), 889–914.
Tasgetiren, M. F., Suganthan, P. N., Pan, Q. K., & Liang, Y. C. (2007). A genetic algorithm for the generalized traveling salesman problem. In *Proceedings of the 2007 IEEE congress on evolutionary computation, Singapore* (pp. 2382–2389).
Ting, T. O., Rao, M. V. C., & Loo, C. K. (2006). A novel approach for unit commitment problem via an effective hybrid particle swarm optimization. *IEEE Transactions on Power Systems, 21*(1), 411–418.
Xie, X. F., & Liu, J. (2008). Multiagent optimization system for solving the traveling salesman problem (TSP). *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, 39*(2), 489–502.
Yi, J., Bi, W., Yang, G., & Tang, Z. (2008). A fast elastic net method for traveling salesman problem. In *Proceedings of the 8th international conference on intelligent systems design and applications, Kaohsiung, Taiwan* (pp. 462–467).
Zhao, G., Luo, W., Nie, H., & Li, C. (2008). A genetic algorithm balancing exploration and exploitation for the traveling salesman problem. In *Proceedings of the fourth international conference on natural computation, Jinan, China* (Vol. 1, pp. 505–509).