Study on Vehicle Routing Problem Based on Improved Particle Swarm Optimization Algorithm

Lin Wang
Chang'an University, School of Automobile
Middle-section of Nan'er Huan Road, Xi'an, China
wangling_fighting@163.com

Guoliang Shen
Chang'an University, School of Electronics and Control Engineering
Middle-section of Nan'er Huan Road, Xi'an, China
15771772135@163.com

Shifeng Niu
Chang'an University, School of Automobile
Middle-section of Nan'er Huan Road, Xi'an, China
nsf530@163.com

ABSTRACT

To improve the optimization ability of particle swarm optimization, we propose an improved particle swarm optimization algorithm and apply it to solve vehicle routing problem. The adaptive particle swarm optimization algorithm based on the combination of linear and nonlinear inertia weight is helpful for the local search in the pre-stage and the global search in the later stage. Combined with simulated annealing algorithm, the improved particle swarm optimization algorithm can avoid the local optimum and the deviation from the optimal solution. In this paper, the vehicle path model is constructed, and the improved particle swarm optimization algorithm is used to optimize the vehicle routing problem. Compared with the genetic algorithm and the improved particle swarm optimization algorithm, the results show that the improved particle swarm optimization algorithm can quickly and effectively get optimal solution to the vehicle routing problem, demonstrating the improved particle swarm optimization algorithm is a more effective method to solve the vehicle routing problem.

KEYWORDS: Particle Swarm Optimization, Vehicle Routing Problem, Inertia Weight, Simulated Annealing Algorithm

1 INTRODUCTION

The vehicle routing problem (VRP), firstly proposed by Dantzig and Ramser in 1959[1], is a combinatorial optimization and integer programming problem. It refers to a series of delivery points (or receipt points) that form the appropriate driving path, so vehicles pass through them orderly to achieve certain goals (such as the shortest distance, the least cost, the less time spent, etc.) in the presence of certain constraints. VRP, proved to be NP-Complete, exists in many aspects of life and is an important problem in the fields of transportation, distribution, and logistics. The main solutions to VRP are heuristic algorithm and exact algorithm. In the heuristic algorithm, genetic algorithm, tabu search algorithm, simulated annealing algorithm and ant colony algorithm are mostly used to solve VRP, and good results have been obtained.

Particle swarm optimization (PSO)[2], a bionic algorithm that simulates the flight of birds, has the advantages of small number of individuals, simple calculation and good robustness. PSO has achieved fairly good performance in all kinds of multidimensional continuous space optimization problems[3]. In this paper, an improved an improved particle swarm optimization algorithm (IPSO) is proposed. Firstly, the adaptive particle swarm optimization algorithm based on the combination of

linear and nonlinear inertia weight is given, which is beneficial to the local search in the pre-stage and the global search in the later stage. Then, to avoid the particle swarm optimization being trapped into the local optimum, the idea of inter-cross from the genetic algorithm is proposed. The IPSO has faster convergence speed, can quickly search the global optimal solution and improve the efficiency of the algorithm.

2 DESCRIPTION OF VRP

VRP is generally described as: A logistics center has a total of vehicles, the capacity of each car is $q_k(k=1,2,...,K)$. Now there are L customer delivery tasks that needs to be done, use $1,2,...,L$ to represent the tasks. The capacity of the i-th customer delivery vehicle is $g_i(i=1,2,...,L)$, and $\max g_i \le \max q_k$. Finally, the shortest route to solve the problem is obtained.

The mathematical model adopted in this paper is the model proposed in the literature [4]. The logistics center number is represented as 0, and the customer delivery task number is represented as $1,2,...,L$. The delivery task and the logistics center are represented as $i(i=0,1,2,...,L)$. The definitions are as follows:

$$y_{ki}=\begin{cases}1 & \textit{the customer delivery task } i \textit{ is completed by the vehicle } k\ ;\\ 0 & \textit{or.}\end{cases} \tag{1}$$

$$x_{ijk}=\begin{cases}1 & \textit{vehicle travels from site } i \textit{ to destination } j;\\ 0 & \textit{or.}\end{cases} \tag{2}$$

The cost of transporting goods here is c_{ij}, indicating the cost of the vehicle from the origin i to the destination j. The c_{ij} here can also represent other meanings, such as distance, time, etc. The conditions required for vehicle optimization are represented by the following expressions:

$$\begin{cases}\min Z=\sum_i\sum_j\sum_k c_{ij}x_{ijk};\\ \sum_i g_i y_{ki}\le q_k \quad k \textit{ is } \forall k;\\ \sum_k y_{ki}=1 \quad i=1,2,....,L;\\ \sum_i x_{ijk}=y_{kj} \quad j=0,1,....,L; k \textit{ is } \forall k;\\ \sum_j x_{ijk}=y_{ki} \quad i=0,1,....,L; k \textit{ is } \forall k;\\ X=(x_{ijk})\in S;\\ x_{ijk}=0 \textit{ or } 1 \quad i,j=0,1,....,L; k \textit{ is } \forall k;\\ y_{ki}=0 \textit{ or } 1 \quad i=0,1,....,L; k \textit{ is } \forall k.\end{cases} \tag{3}$$

As the formula shows, the conditions required for transportation is that (1) each customer delivery task has a transport vehicle; (2) each customer delivery task has a vehicle to be completed; (3) the total quantity demanded of each custom delivery task will not exceed each vehicle's total capacity; (4) the the total length of the vehicle routes reaches the minimum value.

3 IMPROVED PARTICLE SWARM OPTIMIZATION ALGORITHM

IPSO is an evolutionary computation method proposed by Kennedy and Eberhart in 1995[5], the idea of the algorithm originated from imitation of bird predation[2]. Suppose there is a flock of birds are looking for food, they know their current position and the distance from the food, but do not know the specific location of food, so they need to search the area around the bird which is the nearest to the food. In the PSO, the search space is the flight space of birds and the entire flock is a particle swarm in search space. Each bird is considered as a particle of no mass or volume, and the food is the ultimate optimal solution. Each particle's state corresponds to the solution of each of the optimization problems, and each particle has it's own adaptive value which is determined by the optimized function. The particle with the highest adaptive value is the optimal particle. Each particle itself has a certain speed to determine the direction and distance of the flight. Each particle is looking for its optimal adaptive value in each generation, and finally finds the optimal adaptive value for itself and the whole population.

3.1 Simple Particle Swarm Optimization Algorithm

Assuming that the dimension of the search space is n, the total particle number is m, and the number of iterations is t. The position of the $i-th$ particle is represented by vector $X_i^t = (x_{i1}, x_{i2}, ..., x_{in})$. The velocity is represented by vector $V_i^t = (v_{i1}, v_{i2}, ..., v_{in})$. The optimal location of the $i-th$ particle search in history so far is expressed in $P_i^t = (p_{i1}, p_{i2}, ..., p_{in})$. The solution corresponding to the optimal location here is the optimal solution. The position of the $g-th$ particle search is the best position in the whole population history so far, and is represented by vector $P_g^t = (p_{g1}, p_{g2}, ..., p_{gn})$. Then, the following expression particles are updated for their position and velocity:

$$V_i^{t+1} = wV_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (P_g^t - X_i^t) \tag{4}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{5}$$

Among them, $1 \le i \le m$ and the search is carried out in n-dimensional space; c_1 and c_2 are nonnegative constants and are usually called acceleration factors or learning factors; w is inertia weight; r_1 and r_2 are random numbers between [0,1]; t is the number of iterations in the current generation; $t+1$ represents the number of iterations for the next generation.

It can be seen by the formula (4) and the formula (5), the inertia weight w reflects the effect of the last generation velocity on the current generation velocity. That is, the velocity at which the current generation velocity inherits the velocity of the last generation. The larger inertia weights are adapted to the large range search of particles in the n-dimensional space. Conversely, smaller inertia weights are adapted to small range searches of particles in the n-dimensional space. Learning factors c_1 reflects the influence of the optimal location of self-search on the particle search in space; learning factors c_2 reflects the influence of the optimal location of the particle swarm search on the particle search in space. Values of c_1 and c_2 should be kept in a relatively balanced range. If the values of c_1 and c_2 is too large, that is, the acceleration is too large to skip the optimal solution during the search process; if the values of c_1 and c_2 is too small, that is, the acceleration is too small that the particle will be searched in the position of the optimal solution in the search process, and the optimal solution can not be obtained at last. Therefore, the values of c_1 and c_2 are generally ordered $c_1 = c_2 = 2$. In the search process, each dimension of particle velocity and position should have a limit, the velocity limit defined here is $[-V\max_d, V\max_d]$ and the position limit defined here is $[-X\max_d, X\max_d]$. When the particle's velocity and position exceed the limit in the search

process, the particle is then taken as its boundary value. After analyzing the parameters of PSO, the convergence parameter condition of PSO was given Maurice[6].

3.2 Improved Particle Swarm Optimization Algorithm

The main improvements of PSO are as follows:

(1) Improvement of inertia weight by combining linear and nonlinear method. The inertia weight can balance the velocity of the previous generation to the contemporary velocity in the iteration, that is to say, the search ability of the PSO algorithm is demonstrated. With the increasing of the number of iterations, a linear decreasing function is used to control it in the pre-stage and is conducive to the global search; the nonlinear decrement function is adopted to control it in the later stage which is beneficial to local search.

(2) Combining simulated annealing algorithm. PSO is simple, fast and easy to implement, but it is prone to premature convergence and easy to fall into local optima. The simulated annealing algorithm has better local search ability, to further improve the effectiveness of the algorithm, PSO is improved by combining simulated annealing algorithm to avoid the local optimum and deviate from the optimal solution.

3.2.1 The Combination of Linear and Nonlinear Inertia Weight

In the search process, the inertia weight of each particle is improved according to the combination of linear and nonlinear method. In the early stage, each particle search according to the linear decreasing inertia weight, to make wide area search in the early stage, ensure that the search velocity is not too fast to lead to the best solution of missed search, and have a strong global search capability. With the increase in the number of iterations, when the number of iterations reaches a certain value, the inertia weight of each particle is searched according to the nonlinear decrement. Because the particles have reached the optimum value, the particles are searched in a small range, and the search velocity needs to be accelerated. The purpose is to ensure that particles can search the optimal solution quickly and effectively.

The improved expressions for linear and nonlinear inertia weights are given below:

$$W(t)=\begin{cases} W_{\max}-(W_{\max}-W_{\min})\times h_1(t) & ,t\le h; \\ W'-(W'-W_{\min})\times h_2(t) & ,t\ge h. \end{cases} \tag{6}$$

Among them, $W_{\max}$ is the largest inertia weight, that is the initial inertia weight; $W_{\min}$ is the minimum inertia weight, that is the termination of inertia weight; h is iteration number, its range is $0\le h\le T$; T is the maximum number of iterations; W' is the value of the inertia weight when the number of iterations $t=h$ in the iteration process; $h_1(t)$ is the value of the inertia weight when the number of iterations $t\le h$ in the iteration process; $h_2(t)$ is the value of the inertia weight when the number of iterations $t\ge h$ in the iteration process.

In the process of using linear and nonlinear combination of inertia weights, particles are searched at the initial stage ($t\le h$) according to linearly decreasing inertia weights. When $t=h$, record the value of the inertia weight at this time, and the linear decreasing inertia weight is converted to nonlinear decreasing inertia weight. The particles are then searched according to the nonlinear decreasing inertia weight.

3.2.2 The Combination of Linear and Nonlinear Inertia Weight

The simulated annealing algorithm originated from the principle of solid annealing. It was firstly proposed by N.Metropolis et al. in 1953[7] and was introduced to combinatorial optimization by S.Kirkpatrick in 1983. It heats the solids up to a higher temperature and then cools them down slowly. In the process of heating, the particles in the solid move randomly with the increase of temperature, and the energy can be increased; when it cools slowly, the particles can achieve equilibrium at each temperature. When the temperature of the solid reaches the normal temperature,

the particles in the solid will reach the ground state and the internal energy will be reduced to a minimum.

In the field of combinatorial optimization, the annealing process of solids can be formulated as: The internal energy of a solid is equivalent to the evaluation function in combinatorial optimization problems; the state of a solid is equivalent to the solution of a problem; the minimum internal energy of a solid is equivalent to the optimal solution of the problem.

In the optimization process, the PSO selects a current optimal solution every time, easily to fall into the local optimum and deviate from the global optimum. However, the hybrid simulated annealing algorithm can well overcome this shortcoming. The simulated annealing algorithm introduces random factors into the search process, and can accept the solution of the current difference with a given probability, so it can jump out of the local optimal solution to a great extent and achieve the global optimal solution.

Assumed that $C(s^t)$ is the evaluation function, in which s^t is the current state. The current temperature is T. The probability of temperature reduction when the energy difference is ΔE is $P(\Delta E)$. Then, the mathematical description process of simulated annealing algorithm is shown as follows:

(1) If $C(s^{t+1}) \geq C(s^t)$, that is, when the evaluation function of the next state is greater than or equal to the value of the evaluation function of the current state, the particle always accepts the move;

(2) If $C(s^{t+1}) < C(s^t)$, that is, when the evaluation function of the next state is less than the value of the evaluation function of the current state, the particle will accept the motion at a given probability, and this probability will decrease as the number of iterations increases.

The probability in the mathematical description process is expressed as:

$$P(\Delta E) = e^{\frac{\Delta E}{KT}} \tag{7}$$

Among them, K is constant, and $\Delta E < 0$.

It can be seen from the formula (7) that with the increase of temperature T, the probability of decreasing the temperature of the energy difference of ΔE is greater. Conversely, the lower the temperature, the smaller the probability of cooling.

Thus, in the process of finding the optimal solution by the particle swarm, if the solution of the next state is superior to the current solution (the adaptive value of the next state is greater than the current adaptive value), the next state solution is used to replace the current solution, that is, to update the current solution. If the solution of the next state is inferior to the current one, the solution of the next state is accepted with a certain probability according to the principle of the simulated annealing algorithm, so that it can be taken as a current solution with a certain probability. In theory, the simulated annealing algorithm has been proved to be a global optimization algorithm, and the probability of convergence to the global optimum is 1.

After the simulated annealing algorithm is applied, the particles will accept the particles with lower probability when they search for the best adaptive value. Therefore, the initial temperature should be sufficiently large to ensure that the particle has a strong initial jump in the initial stage. The initial temperature here is defined as:

$$T_0 = \frac{C(p_b^1) - C(p_w^1)}{\ln p_s} \tag{8}$$

Among them, $C(p_b^1)$ is the evaluation function value of the optimal particle when the initial random particle is defined; $C(p_w^1)$ is the evaluation function value of the worst particle when the initial random particle is defined; p_s is the probability that the optimal particle will be replaced by the worst particle at the initial temperature.

Roulette game is also called proportional selection method. The basic idea is that the probability of each individual is proportional to its adaptive value. The concrete calculation process is as follows:

(1) Calculate the adaptive value of each particle in the particle swarm $C(i=1,2,...,N)$ and N is the particle swarm size;

(2) Calculate the probability that each particle could be passed on to the next generation:

$$P(x_i)=\frac{C(x_i)}{\sum_{j=1}^{N}C(x_j)} \tag{9}$$

(3) Calculate the cumulative probability of each particle:

$$q_i=\sum_{j=1}^{i}P(x_j) \tag{10}$$

(4) Generated a uniformly distributed random number r between $[0,1]$;

(5) If $r<q_1$, select x_1; otherwise, select x_k to make $q_{k-1}<r<q_k$.

In this algorithm, the jump probability is $e^{\frac{-(C(p_i)-C(p_g))}{T}}$, and the probability that each particle is inherited to the next generation is:

$$P(x_i)=\frac{e^{\frac{-(C(p_i)-C(p_g))}{T}}}{\sum_{j=1}^{N}e^{\frac{-(C(p_j)-C(p_g))}{T}}} \tag{11}$$

Therefore, according to probability $P(x_i)$, roulette strategy is used to determine whether more contemporary particles are replacing the contemporary.

After adding the simulated annealing algorithm, the specific search steps are designed as follows:

(1) Initialize the particle swarm. Define the particle swarm number is m, the number of search space dimensions is d (the number of variables in each particle's velocity and position is d). Define the initial iteration for algebra $t=1$, the maximum iteration number is T. Define random parameters for r_1 and r_2; define learning factors (acceleration factor) for c_1 and c_2. The initial inertia weight is defined as $W_{\max}$, and the inertia weight $W_{\min}$ is terminated. The initial position X_i^t and initial velocity V_i^t of each particle are given, and the initial temperature value T_0 is defined.

(2) Calculate the initial adaptive value of each particle in the particle population. The initial adaptive value for each particle are calculated based on the initial position and the initial velocity of the given particles.

(3) Calculate the inertia weight according to the formula (6).

(4) Update the next generation of each particle position and velocity according to the formula (4), (5).

(5) Calculate the adaptive value according to the velocity and position of the updated particles and determine the next generation of particle fitness and contemporary adaptive value of the size. If the adaptive value of the next generation particle is superior to that of the present, the next generation particle adaptive value is used as the optimum position of the particle p_i^t. Otherwise, roulette strategies are used to allow some particles to be selected and to accept particles that p_i^t are more contemporary to be updated, and the rest to remain the same.

(6) Reduce the temperature. The temperature reversal function is chosen as $T_k = \lambda T_{k-1}$, where λ is constant.

(7) Turn to Step (3) until the maximum iterations are reached. If satisfied, then move on to the next step.

(8) Output the optimal position and velocity of particles after iteration.

4 APPLICATION

4.1 VRP Model

Referring to the ideas in document [8], the VRP model is established as follows:

The vehicle route dimension are constructed, which are set as 2L dimensions. The central warehouse carries on the assignment to each customer, so the delivery process involves the vehicle selection of distribution target task and the selection of delivery order for each task vehicle. Therefore, for vehicle number k and distribution order r, L delivery tasks are divided into two L dimensional position vectors: X_k (delivery target, task vehicle selection) and X_r (distribution order of each task vehicle). The corresponding velocity vectors are V_k and V_r, respectively.

For example, there are 10 customer delivery tasks that need to be completed. The warehouse is shipped by the central warehouse, and there are 4 vehicles in the central warehouse, and the corresponding position vectors can be represented as:

Number of delivery tasks: 1 2 3 4 5 6 7 8 9 10

X_k: 1 2 2 2 3 3 3 3 4 4

X_r: 1 3 1 2 1 4 2 3 1 2

It can be seen from the delivery task:

vehicle $1: 0 \to 1 \to 0$

vehicle $2: 0 \to 3 \to 4 \to 2 \to 0$

vehicle $3: 0 \to 5 \to 7 \to 8 \to 6 \to 0$

vehicle $4: 0 \to 9 \to 10 \to 0$

4.2 Algorithm Implementation

(1) Initialize the particle swarm. Define the value of X_k as an integer between $1 \sim \mathrm{k}$ (vehicle number). The value of X_r is the real number between $1 \sim \mathrm{L}$ (the number of delivery tasks). The value of V_k is an integer between $-(\mathrm{k}-1) \sim (\mathrm{k}-1)$. Define parameters r_1, r_2, c_1 and c_2. Define the initial iteration is algebra $t = 1$, the maximum iteration number is T.

(2) Calculate the initial adaptive value of each particle in the particle swarm. The initial adaptive value of each particle is calculated according to the initial position and initial velocity of the given particle.

(3) Calculate the inertia weight according to the formula (6).

(4) Update each particle's velocity and position of the next generation according to fomula (4) and fomula (5).

(5) Calculate the adaptive value according to the velocity and position of the updated particles and determine the next generation of particle fitness and contemporary adaptive value of the size. If the adaptive value of the next generation particle is superior to that of the present, the next generation particle adaptive value is used as the optimum position of the particle p_i^t. Otherwise, roulette strategies are used to allow some particles to be selected and to accept particles that p_i^t are more contemporary to be updated, and the rest to remain the same.

(6) Reduce the temperature. The temperature reversal function is chosen as $T_k = \lambda T_{k-1}$, where λ is constant.

(7) Turn to Step (3) until the maximum iterations are reached. If satisfied, then move on to the next step.

(8) Output the optimal position and velocity of particles after iteration.

4.3 Simulation and Analysis

In this paper, the case in document [9,10] is simulated and compared with the results of the algorithm proposed in the literature [9,10]. Cases are as follows:

A logistics center has 2 vehicles, of each the capacity is 8 tons. There are 8 customer delivery tasks need to be completed, the distance and quantity demanded between each customer is shown in Table 1. The shortest route to solve the problem is finally obtained. Among them, c_{ij} represents the distance between the logistics center and each customer, and the unit of quantity demanded is ton.

Table 1 Quantity demanded and the distance between the logistics center and each customer

c_{ij}	0	1	2	3	4	5	6	7	8
0	0	4	6	7.5	9	20	10	16	8
1	4	0	6.5	4	10	5	7.5	11	10
2	6	6.5	0	7.5	10	10	7.5	7.5	7.5
3	7.5	4	7.5	0	10	5	9	9	15
4	9	10	10	10	0	10	7.5	7.5	10
5	20	5	10	5	10	0	7	9	7.5
6	10	7.5	7.5	9	7.5	7	0	7	10
7	16	11	7.5	9	7.5	9	7	0	10
8	8	10	7.5	15	10	7.5	10	10	0
Quantity Demanded	0	1	2	1	2	1	4	2	2

The algorithm parameters are defined as follows:

The learning factor is selected as $c_1 = c_2 = 1$. r_1 and r_2 are random numbers between [0,1]. The maximum iteration number in the pre-stage is $T_1 = 500$ and the maximum iteration number in the later stage is $T_2 = 500$. In fomula (6), the parameters in the pre-stage are defined as $h = T_1 / 2$, $h_1(t) = t / T_1$ and $h_1(t) = e^{(-16(T_1 - t)^3)/T_1^3}$, the parameters in the later stage are defined as $h = T_2 / 2$, $h_1(t) = t / T_2$ and $h_2(t) = e^{(-16(T_1 - t)^3)/T_1^3}$. Among them, λ is defined $\lambda = 0.95$.

The IPSO is used in this paper, and the running results are shown in table 2.

Table 2 The running results of IPSO

Run times	1	2	3	4	5	6	7	8	9	10
Final distance	67.5	68	67.5	68	69	67.5	67.5	68	67.5	67.5
Run times	11	12	13	14	15	16	17	18	19	20
Final distance	69	67.5	67.5	68	67.5	68	68	67.5	67.5	67.5
Run times	21	22	23	24	25	26	27	28	29	30
Final distance	67.5	67.5	67.5	69	68	67.5	67.5	69	68	67.5

Tables and figures should be placed close after their first reference in the text. All figures and tables should be numbered with Arabic numerals. Table headings should be centred above the tables. Figure captions should be centred below the figures as shown in Figure 1.

The route order of each vehicle in the delivery task is $0 \to 4 \to 7 \to 6 \to 0$ and $0 \to 1 \to 3 \to 5 \to 8 \to 2 \to 0$ respectively.

Compare the results of IPSO with the results of PSO and the improved genetic algorithm.

Table 3 Table parameters

Algorithm	Minimum distance	Average distance	Running time
IPSO	67.5	67.83	0.032s
PSO	67.5	68.375	0.024s
Improved Genetic Algorithm	67.5	68.3	0.032s

As can be seen from Table 3, the average distance is better than the other two algorithms under the premise that the IPSO runs 30 times, and the running time is shorter.

It can be seen that the IPSO has achieved good results in VRP. Compared with the PSO and the improved genetic algorithm, IPSO has the advantages of high efficiency, good optimization effect, high search result and high quality. Therefore, the IPSO is helpful and effective for VRP.

5 CONCLUSIONS

To overcome disadvantage of slow convergence rat and local optimum, the IPSO is proposed in this paper. Introduced the combination of linear and nonlinear inertia weight to PSO, the local search velocity in the pre-stage and the global search velocity in the later stage are improved. And the PSO is further improved by applying the genetic algorithm in raising the accuracy and speed effectively. Through the simulation and analysis of the example, the IPSO can search the optimal path in a relatively short period of time and get the shortest distance. Compared with the PSO and the improved genetic algorithm, the results indicated that IPSO can improve the success rate of searching the best route and is better and more effectively method for VRP.

REFERENCES

Dantzig, George Bernard; Ramser, John Hubert (1959). The Truck Dispatching Problem [C]. Management Science. 6 (1): 80–91.

Kennedy J, Eberhart R C. Particle swarm optimization[A]. Proc. IEEE International Conference on Neural Networks, IV[C]. Piscaraway, NJ: IEEE Service Center, 1995. 1942-1948.

Eberhart R C, Shi Y. Particle Swarm Optimization: Developments, Applications and Resources[C]. Proc. Congress on Evolutionary Computation 2001. Piscataway, NJ: IEEE Press，2001.81-86.

Li Jun, Guo gang. Theory and method of vehicle optimal scheduling for logistics distribution [M]. Beijing: China Materials Press,.2001.76-77.

Eberhart R C, Kennedy J. A new using particle swarm theory[C]. Proceedings of the 6th international Symposiumon Micro Machineand Human Science.Piscataway,N.J. ,USA:IEEE Service Center, 1995: 39-43.

Steinbrunn M, Moerkotte G, Kemper A. Heuristic and Ran2 domized Optimization for the Join Ordering Problem[J]. The VLDB Journal , 1997, 6 (3): 8-17.

Shi Y, Eberhart R C. Empirical study of particle swarm optimization[A]. Proceedings of the 1999 Congress on Evolutionary Computation[C]. Piscataway, NJ: IEEE Service Center, 1999. 1945-1950

Shi Y, Eberhart R C. Empirical study of particle swarm optimization[A]. Proceedings of the 1999 Congress on Evolutionary Computation[C]. Piscataway, NJ: IEEE Service Center, 1999. 1945-1950.

Xiao Jianmei, Li Junjun, Wang Xihuai. Improved particle swarm optimization algorithm for vehicle routing problem. Computer integrated manufacturing system, 2005, 11 (4):577-581.

Zhang Yuxing, Fan Jianhua, Xu Jiangang, Chen Dongsheng. Improved genetic algorithm for vehicle routing problem. Journal of Tianjin University of Technology, 2006, 22 (5),:79 - 82.