Int j simul model 13 (2014) 2, 230-242
ISSN 1726-4529 Original scientific paper

OPTIMIZATION OF DISTRIBUTION ROUTE SELECTION BASED ON PARTICLE SWARM ALGORITHM

Wu, Z.
Information College, Capital University of Economics and Business, 121 Zhangjialukou, Huaxiang Fengtai District, Beijing 100070, P. R. China
E-Mail: wuz9080@163.com

Abstract
This paper mainly discusses the application of the particle swarm optimization in logistics distribution routing problems. Combining with the characteristics of logistics and distribution, it established a mathematical model of the distribution routing problem. Introducing three kinds of optimization strategies in the particle swarm optimization to optimize the particle swarm algorithm, constructing three different particle swarm algorithms of LinWPSO, SAPSO and RandWPSO, used respectively the standard of PSO, LinWPSO, SAPSO and RandWPSO to solve calculation cases of problems in logistics and route, the results showed that the performance of the LinWPSO, SAPSO and RandWPSO to solve vehicle routing problem is better than standard PSO. Performance of SAPSO is optimal, which can effectively solve vehicle routing problems of the logistics distribution, when the problem size increases, optimization advantages of the SAPSO will display fully, we can greatly shorten the delivery mileage by using the SAPSO to solve the logistics distribution routing problem.
(Received, processed and accepted by the Chinese Representative Office.)

Key Words: Supply chain, Logistics and Distribution, PSO

1. INTRODUCTION

Distribution is a very important aspect of the logistics system. Of all the costs in the logistics, distribution costs accounted for a very high proportion. Path planning problem is the core issue of the distribution system, and the research focus too. Reasonable path arrangement can effectively improve transport efficiency and reduce service costs.

This paper backed on the logistics and distribution, conducted in-depth research of logistics distribution routing problem using several improved PSO. Particle swarm algorithm is an evolutionary computation technology based on swarm intelligence method, which has a profound intelligence background, a quick convergence speed, it is easy to implement and only a few parameters need to be adjusted, and thus when put forth was became a new research focus of intelligent optimization and evolutionary computing field. The basic idea of PSO is to find the optimal solution through collaboration between individuals and information sharing of the group, which was widely used in scientific and engineering problems.

For defects of premature convergence and lack of local search capabilities of the PSO, this paper introduces three PSO strategies of linear decreasing weights optimization (LinWPSO), self adaptive weight optimization (SAPSO), and random weight optimization (RandWPSO), these three algorithms have great optimization on convergence precision than standard PSO. Which, in the convergence precision, the SAPSO is bigger than the LinWPSO,and the LinWPSO is bigger than the RandWPSO.

2. STRATEGY OF WEIGHT IMPROVEMENT OF THE PSO

The basic particle swarm optimization algorithm can be described as following:

$$v_i(n+1) = v_i(n) + c_1 r_1 (p_i - x_i(n)) + c_2 r_2 (p_g - x_i(n)) \quad (1)$$

$$x_i(n+1) = x_i(n) + v_i(n) \tag{2}$$

Based on the basic particle swarm optimization algorithm, Shi and other scholars has amended on the previous eq. (1), and introduced inertia weight factor ω [1, 2]:

$$v_i(n+1) = \omega v_i(n) + c_1 r_1 (p_i - x_i(n)) + c_2 r_2 (p_g - x_i(n)) \tag{3}$$

Inertia weight ω was used to control the impact of the particle's former velocity over current velocity; it will affect global and local search capabilities of the particle. You can balance global and local search capability by select a suitable ω, which allows the algorithm to find the optimal solution quickly with a minimum iteration time. Initially, Shi taken ω as a constant and later experiment was found that dynamic ω will be able to get better optimization results than a fixed value. Because the smaller ω can strengthen local search capabilities, while the larger ω can speed up the convergence speed, so through adjusting ω we can achieve the balance between convergence speed and local search capability. Dynamic ω can change linearly in search process of PSO algorithm; it can also change dynamically based on a measure function of the PSO algorithm performance [3, 4].

Through simulation experiments we found, that the impact of parameters on the performance of the algorithm has certain rules which can be found. Weight ω setting is an important content of parameter setting, which has a great impact over algorithm optimization results. For different optimization problem, the ω setting when obtain optimal result is often not identical completely.

2.1 Weight policy of linearly decreasing

Since the larger inertia weight is conducive to escape from local minima, to facilitate global search, while a smaller inertia weight is conducive for accurate local search of the current search area, which is conducive to convergence of the algorithm, so against the easy prematurity of the PSO algorithm and the oscillation phenomenon generated in the late algorithm near the global optimal solution , the linear changing weight can be used, so that the inertia weight can decreases linearly from the maximum value ω_{max} to the minimum value ω_{min}, change formula of ω with the algorithm iteration time is [5, 6]:

$$\omega = \omega_{\max} - t \frac{\omega_{\max} - \omega_{\min}}{t_{\max}} \tag{4}$$

In the formula, ω_{max} is the maximum inertia weight and ω_{min} is the minimum inertia weight respectively, t is the former iteration step, t_{max} is the total iteration time of the algorithm. ω often varies between 0.4 and 0.9, decrease with the increase of iteration steps. Linear transformation of the ω makes the algorithm has a faster convergence speed in early time, while has strong local search ability in later time. Introduction of ω greatly improved the performance of the PSO algorithm, for different search problems, you can adjust the global and local search capability; the PSO algorithm can be successfully applied to many practical problems either. Change curve of ω is as Fig. 1.

2.2 Self adaptive weight strategy

In order to balance the global search ability and the ability of local improvements of the PSO algorithm, non-linear dynamic inertia weight factor formula can be used either, the expression is as following:

$$\omega = \begin{cases} \omega_{\min} - \dfrac{(\omega_{\max} - \omega_{\min}) * (f - f_{\min})}{f_{avg} - f_{\min}}, f \le f_{avg} \\ \omega_{\max}, f > f_{avg} \end{cases} \tag{5}$$

wherein ω_{max} and ω_{min} express the maximum and minimum values of ω respectively, f represents the current target function value of the particle, f_{avg} and f_{min} stand respectively for the current average target value and the minimum target value of all particles. In the above formula, the inertia weight is automatically changed with the objective function value of the particle, so called self adaptive weight [7].

When the target values of each particle are identical or regional local optimal, the inertia weight will increase, when the target value of particles are scattered, the inertia weight will decrease, while for particle whose objective function value is better than the average target value, the corresponding inertia weighting factor is small, thereby protecting the particle, in contrary to particle whose objective function value is worse than the average target value , the corresponding inertia weight factor is large, so that the particle moves closer to a better search area. Change curve of ω is as Fig. 2.

2.3 Random weight strategy

Set the standard PSO algorithm as the random number obeying some random distribution, so to some extent, we can overcome the shortage brought by the linear decreasing of the ω from two aspects.

First, if it close to the best point in early evolution, random ω may generate a relatively smaller ω value, to accelerate the convergence speed, in addition, if we can't find the best point in initial algorithm, the linear decreasing of ω, making the algorithm can't converge to this best point eventually, while the random generation of the ω can overcome this limitation [8].

The calculation formula of the ω is as following:

$$\begin{cases} \omega = \mu + \sigma * N(0,1) \\ \mu = \mu_{\min} + (\mu_{\max} - \mu_{\min}) * rand(0,1) \end{cases} \tag{6}$$

in which $N(0,1)$ represents a random number that is standard normally distributed, $rand(0,1)$ represents a random number between 0 and 1. Change curve of ω is as Fig. 3.

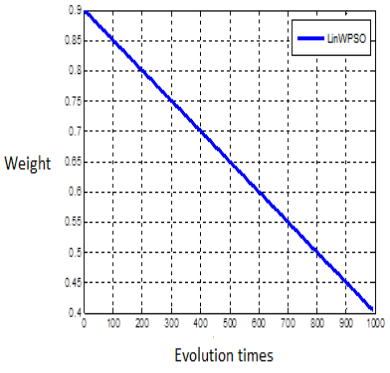

Figure 1: Linear decreasing ω.

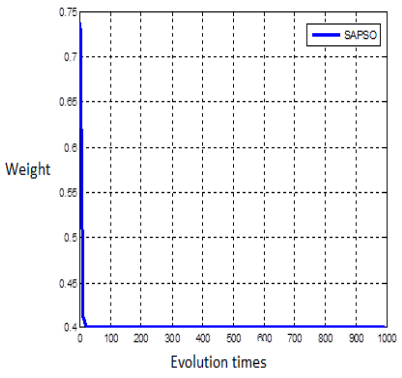

Figure 2: Self adaptive ω.

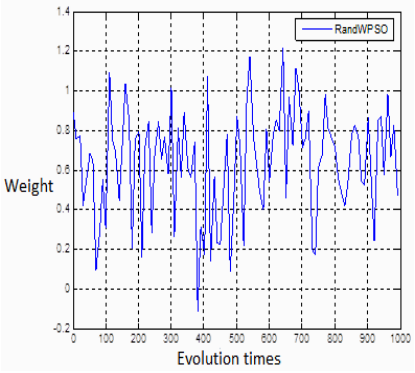

Figure 3: Random ω.

3. THREE KINDS OF WEIGHT IMPROVEMENT STRATEGY TEST

3.1 Introduction of several test functions

(1) Griewank function formula:

$$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1 \tag{7}$$

It can be obtained from the image of the function, that the function obtained the minimum value of 0 in a certain range.

(2) Rastrigin function formula:

$$f(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10) \tag{8}$$

Obtained from the function image, the function obtained the minimum value of 0 in a certain range.

(3) Schaffer function formula:

$$f(x,y)=0.5-\frac{\sin^2\sqrt{x^2+y^2}-0.5}{[1+0.001(x^2+y^2)]^2} \tag{9}$$

Obtained from the function image, the function obtained the minimum value of -1 within a certain range.

(4) Ackley function formula:

$$f(x)=-20\exp\left[-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right]-\exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right]+20+e \tag{10}$$

Obtained from the function image, the function obtained the minimum value of 0 in a certain range.

(5) Rosenbrock function formula:

$$f(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2] \tag{11}$$

Obtained from the function image, the function obtained the minimum value of 0 in a certain range.

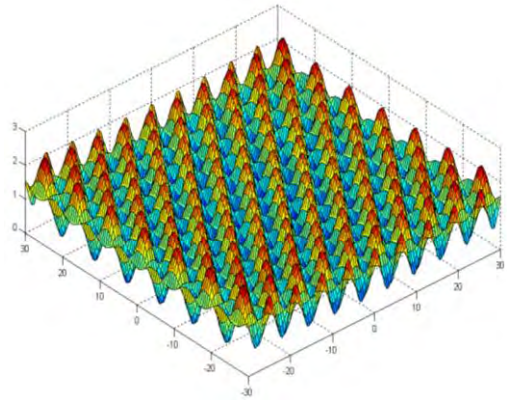

Figure 4: Griewank function.

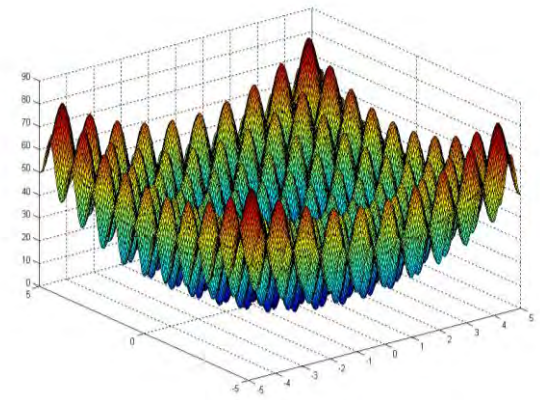

Figure 5: Rastrigin function.

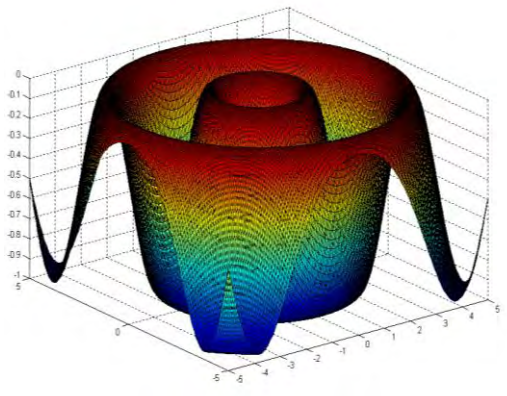

Figure 6: Schaffer function.

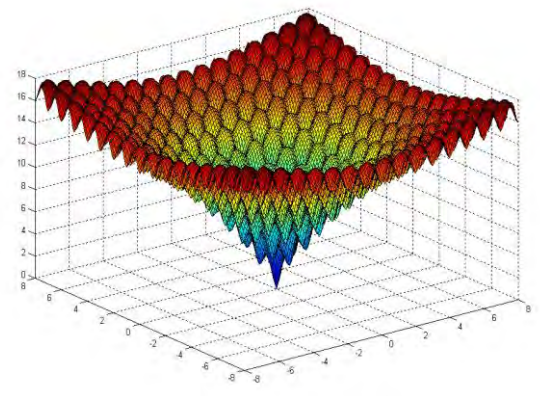

Figure 7: Ackley function.

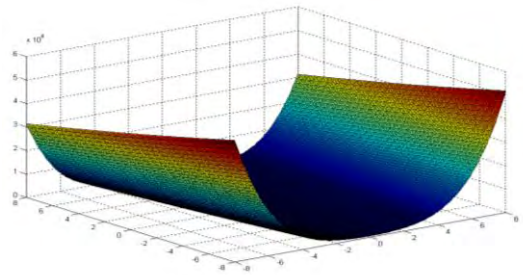

Figure 8: Rosenbrock function.

3.2 Test results and conclusion analysis

Use the five listed test functions above, test repeatedly for three strategies of LinWPSO, SAPSO, RandWPSO, take the average value of 10 times running results from the test results, the following is the resulting evolution curve and the data results. Parameter selection of the following test: particle dimension $D = 10$; r_1, r_2 are randomly generated numbers between [0, 1]; learning factors c_1, c_2 were selected as two. Select the number of particles N as 40. Select the number of iterations as 500 generations. Each function's test result as following:

Table I: Test result data table.

Test function	Improvement strategy	Average optimal fitness value	Running time(s)	Standard difference
Griewank	LinWPSO	0	1.464	4.756398e-008
	SAPSO	0	1.898	6.175240e-005
	RandWPSO	0	1.468	3.958067e-008
Rastrigin	LinWPSO	10.172	1.259	2.930492
	SAPSO	7.945	1.653	3.880726
	RandWPSO	8.217	1.257	4.666770
Schaffer	LinWPSO	-0.998	0.857	0
	SAPSO	-0.996	1.134	0.004693
	RandWPSO	-0.995	0.857	0.004096
Ackley	LinWPSO	0	1.423	0.557927
	SAPSO	0.411	1.942	0.626560
	RandWPSO	0.293	1.415	0.674569
Rosenbrock	LinWPSO	0	0.745	0
	SAPSO	0	0.964	0
	RandWPSO	0	0.740	0

(1) From the optimization effect

Three improved algorithms are good enough to optimize the function, which can converge stably to the optimal solution region. Where fluctuation of the initial adaptive value of the linear decreasing improvement strategy is big, which is linearly related to the ω, which inherited more of the speed of the previous generation, which is easy to leap out a local and search, so the fluctuation is big.

(2) From the convergence rate

Obtained from the evolution curve, of these three improvement strategies, SAPSO has obvious advantages on accuracy and speed of convergence.

(3) From the running time

Know from the operation results data table, that during many tests, running time of SAPSO is greater than the other two algorithms, so when the iteration time is big and repeat test time is frequent, the running time will have a greater difference.

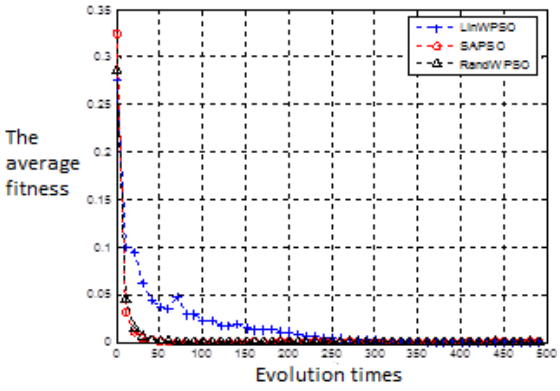

Figure 9: Griewank function's test evolution.

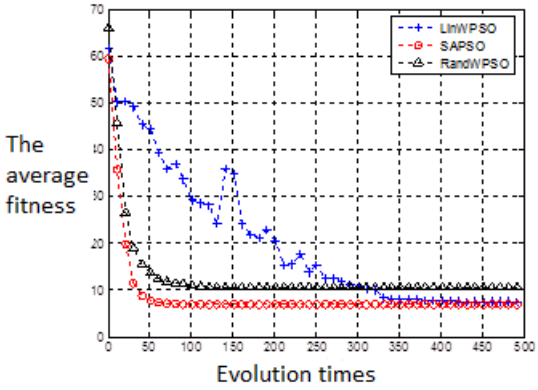

Figure 10: Rastrigin function's test evolution.

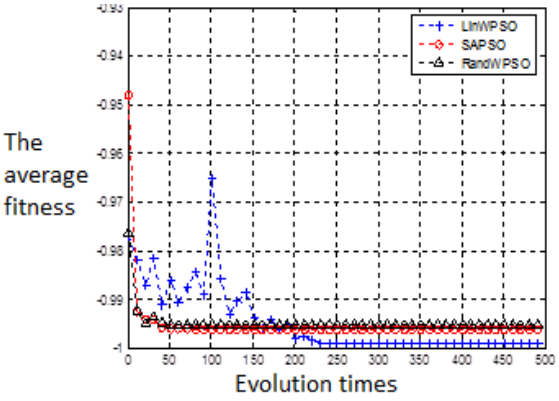

Figure 11: Schaffer function's test evolution.

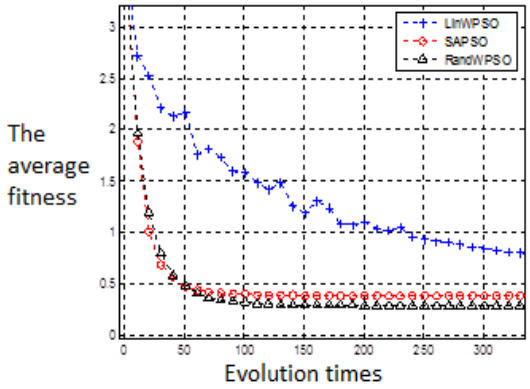

Figure 12: Ackley function's test evolution.

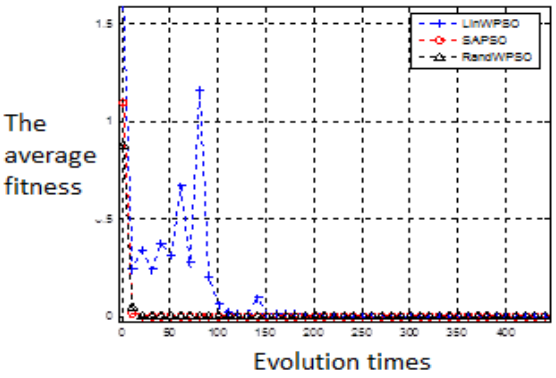

Figure 13: Rosenbrock function's test evolution.

4. SOLVE LOGISTICS AND DISTRIBUTION ROUTING PROBLEM WITH PSO

4.1 Mathematical model of the logistics distribution vehicle routing

Logistics distribution routing problem can be described as following: from a distribution centre, with multiple distribution vehicles, deliver to multiple customers, each customer's location and goods demand are certain, each distribution vehicle's load is certain, we require reasonable arrangements for vehicle delivery routes, make the objective function been optimized, and meet the following conditions [9, 10]:

1, each customer's demand total of each distribution path does not exceed the carrying capacity of the delivery vehicle;

2, a car can only choose one path, but can serve multiple customers;

3, a customer has one and only one car service for him;

4, the vehicle starts from the distribution centre, along a route, after delivers the loading goods to corresponding customers, returns to their distribution centre.

Before arranging route, you need to estimate the number of vehicles using. In reality, the more complex of the cargo is loaded (unloaded) from the vehicle, we have the more constraints, and the less of the vehicle's actual cargo capacity. We use the following eq. (12) to determine the number of vehicles K that we need:

$$K = \left[\frac{\sum_{i=1}^{L} g_i}{aq} \right] + 1 \tag{12}$$

where [ ] indicates to take integer that is not bigger than the figure in the brackets, $a \in (0,1)$ can be adjusted according to how many of constraints, the more general constraints, a is smaller, and vice versa, a is bigger. The general value of a is 0.85. c_{ij} means transport costs between clients i and j, such as time, distance, cost and so on. Distribution centre is numbered 0, the number of each customer is i ($i = 1, \ldots, L$), the number of each vehicle is k ($k = 1, \ldots, K$), the goods demand of the i^{th} customer is g_i, capacity of the delivery vehicle is q.

Define the following variables:

$$x_{ijk} = \begin{cases} 1 & vehicle\ k\ \ from\ i\ to\ j \\ 0 & else \end{cases} \tag{13}$$

$$x_{ijk} = \begin{cases} 1 & vehicle\ k\ serve\ customer\ i \\ 0 & else \end{cases} \tag{14}$$

Modelling goal is to minimize total cost of transport. Transport costs is proportional to the vehicle path, the shorter driving route, the less fuel consumption of the vehicle, the less driver's working time, of course, the less total transportation cost [11]. Here is a mathematical model of the objective function established for the logistics distribution vehicle routing problem with the shortest path:

$$\min z = \sum_{i=0}^{L} \sum_{j=0}^{L} \sum_{k=1}^{K} c_{ij} x_{ijk} \tag{15}$$

$$\sum_{i=1}^{L} g_i y_{ik} \le q; \forall k \tag{16}$$

$$\sum_{k=1}^{K} y_{ik} = \begin{cases} 1 & i = 1,2,\ldots,L \\ K & i = 0 \end{cases} \tag{17}$$

$$\sum_{i=0}^{L} x_{ijk} = y_{ik} \qquad j = 0,1,\ldots,L; \forall k \tag{18}$$

$$\sum_{j=0}^{L} x_{ijk} = y_{jk} \qquad i = 0,1,\ldots,L; \forall k \tag{19}$$

$$x_{ijk} = 0\ or\ 1 \qquad i, j = 0,1,\ldots L; \forall k \tag{20}$$

$$y_{ik} = 0\ or\ 1 \qquad i = 0,1,\ldots L; \forall k \tag{21}$$

In the model: eq. (15) is the objective function; eq. (16) represents the vehicle capacity constraints, total freight loaded by each vehicle shall not exceed the maximum loading capacity; eq. (17) means that only one car service for each customer, all of the tasks completed by the car K together; eq. (18) means only one vehicle reaches a certain customer; eq. (19) indicates only one vehicle left a customer; eqs. (20) and (21) are integer constraints.

4.2 Particle coding

For logistics distribution vehicle routing problem, particle coding is the key to realize the algorithm. For logistics distribution vehicle routing problem, genetic algorithms generally use natural number coding. In the use of particle swarm algorithm, this paper transforms coding of the logistics distribution vehicle routing problem into real number coding, using vector expression forms for particle coding, sorting each element by size, which means service order of vehicles to each customer [12, 13]. For example, assume within a distribution vehicle routing problem, the number of vehicles dispatched from the distribution centre is 3, the number of customers is 8, if a particle's position vector X is:

Customer number: 1234567800

X: 3.2 1.9 4.5 2.3 7.8 6.4 5.2 5.8 2.4 5.0

Customer number 0 indicates the distribution centre, reorder customer number in accordance with the size of the corresponding element X, get total path of all the vehicles. The results were as following:

Gross vehicle path: 2401307865

As the vehicle starts from the distribution centre and then returns to the distribution centre. Therefore, access orders of three corresponding vehicles are:

Vehicle 1: Distribution Centre → Customer 2 → Customer 4 → Distribution Centre

Vehicle 2: Distribution Centre → Customer 1 → Customer 3 → Distribution Centre

Vehicle 3: Distribution Centre → Customer 7 → Customer 8 → Customer 5 → Customer 6 → Distribution Centre

Dimension of the position vector X of the particle is related with the number of vehicles K and the number of customers L, and $D = K + L - 1$, the dimension of the velocity vector V and the position vector X of the particle is the same. After order particles according to the value of each dimension, it usually appears that alpha and omega is 0 and two 0 neighbouring circumstances, such particles are infeasible. Before calculating the fitness value, judge whether the particle is 0 alpha and omega or with two adjacent 0, and if so, then set the adaptive value to infinity, the purpose of doing so is to eliminate infeasible solutions quickly and save computing time.

4.3 The fitness function

For general logistics distribution vehicle routing problem, use the following transform to make capacity constraint eq. (22) becomes a part of the objective function:

$$\min z = \sum_{i=0}^{L}\sum_{j=0}^{L}\sum_{k=1}^{K} c_{ij}x_{ijk} + R \times \sum_{k=1}^{K} \max\left(\sum_{i=1}^{L} g_i y_{ik} - q, 0\right) \tag{22}$$

$R \times \sum_{k=1}^{K} \max\left(\sum_{i=1}^{L} g_i y_{ik} - q, 0\right)$ means the penalty value punishable by capacity constraint violation. In order to meet stringent capacity constraints, R should tend to infinity. However, considering the convenience of computer handling, R could be an appropriate positive number, this paper taking $R = 10^{10}$. The aim is to make the infeasible solutions to be eliminated in an iterative process.

5. RESULTS AND DISCUSSION

5.1 Example A

In order to facilitate comparative analysis, the problem is the distribution system of eight stores and one distribution centre. The number of distribution vehicles by the distribution

centre is 2, the vehicle capacity is 8 tons. The distance between the chains (km) and its demand (ton) was shown in Table II, the distribution centre number is 0, required to arrange a suitable route, minimize the total transport mileage.

Table II: The distance between the demand chains.

Chain number	**Demand**	**Distance**								
		0	1	2	3	4	5	6	7	8
0	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	9	15
4	2	9	10	10	10	0	10	7.5	7.5	10
5	1	20	5	10	5	10	0	7	9	7.5
6	4	10	7.5	7.5	9	7.5	7	0	7	10
7	2	16	11	7.5	9	7.5	9	7	0	10
8	2	8	10	7.5	15	10	7.5	10	10	0

Algorithm parameter setting: the dimensions of the particle $D = 8 + 2 - 1 = 9$, particle number as 40, $c_1 = 1.5$, $c_2 = 1.5$. Evolution algebra takes 500; optimal total path obtained through simulation is: 6 → 7 → 4 → 0 → 1 → 3 → 5 → 8 → 2, corresponding routes as following:

Vehicle 1: Distribution Centre → Chain Store 6 → Chain Store 7 → Chain Store 4 → Distribution Centre

Vehicle 2: Distribution Centre → Chain Store 1 → Chain Store 3 → Chain Store 5 → Chain Store 8 → Chain Store 2 → Distribution Centre.

Obtained the shortest total driving distance is 71.5 km.

Since weight ω improvements, the performance of the PSO algorithm is greatly improved. Standard PSO, LinWPSO, SAPSO and RandWPSO's optimal particle fitness value changes with iteration, which are shown in Figs. 14 to 17.

Standard PSO algorithm, LinWPSO, SAPSO and RandWPSO all carried 20 times calculation; calculation results are shown in Table III. From the table, as seen from the test results, in the case of the same iteration time, the improved PSO algorithm outperforms the standard PSO, because the improved PSO adjusted the weight w value dynamically to each generation groups, which improved the local search ability of the standard PSO. SAPSO calculation results are better than LinWPSO and RandWPSO significantly.

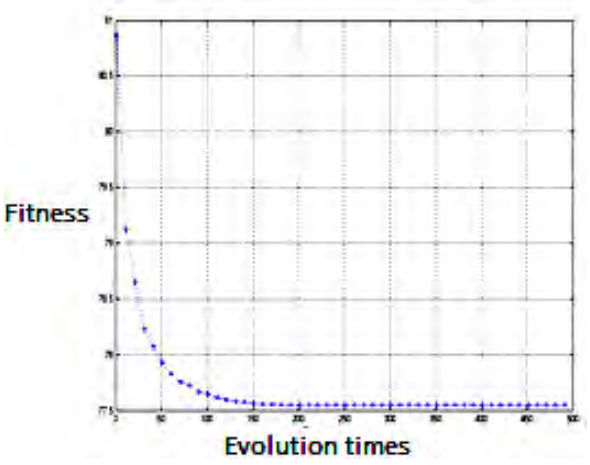

Figure 14: Standard PSO evolution.

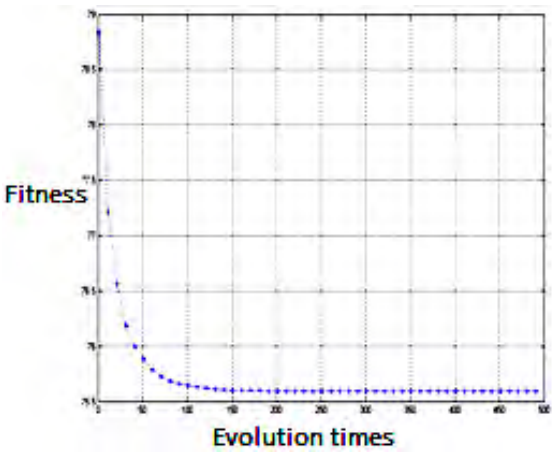

Figure 15: LinWPSO evolution.

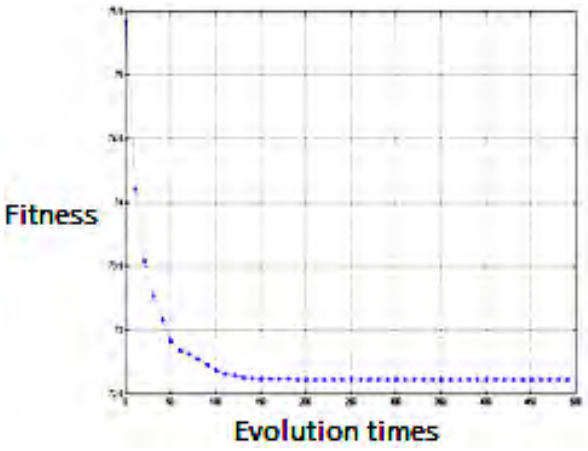

Figure 16: SAPSO evolution.

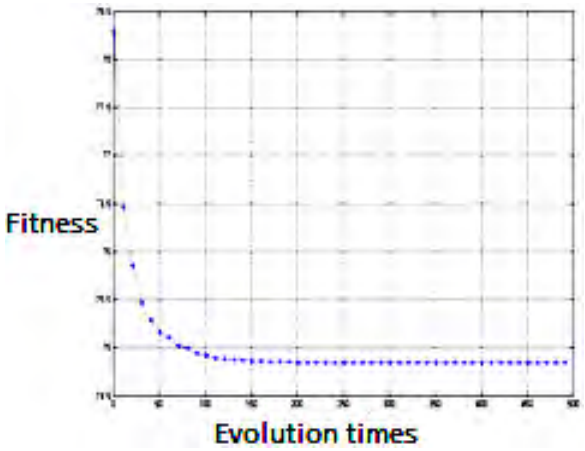

Figure 17: RandWPSO evolution.

We use a standard genetic algorithm (population size is 40, crossover probability is 0.8, mutation probability is 0.05) to carry out 20 times calculations to the problem. Algorithm iteration times are all 50, the standard genetic algorithm achieved the known optimal solution at one time. It is obvious from Table III, in case of roughly the same population size, SAPSO equally superior to standard GA.

Analysis showed that, SAPSO can solve the logistics distribution vehicle routing problem quickly and efficiently. The algorithm is simple, convenient, and the programming is easy, it is one of practical and feasible optimization method to solve the logistics distribution vehicle routing problem.

Table III: Results comparison of the Standard GA, Standard PSO, LinWPSO, SAPSO, RandWPSO.

Number of operations	Standard GA	Standard PSO	LinWPSO	SAPSO	RandWPSO
1	84	78	75.5	72	76
2	85	77.5	75	71.5	74.5
3	81.5	75.5	75	75	74.5
4	82	79	75.5	73	76
5	83.5	77	76	71.5	73.5
6	85	78.5	76	74	75.5
7	83	80	77.5	71.5	74.5
8	82.5	75.5	73.5	73	75.5
9	85.5	79.5	78	71.5	73.5
10	85.5	77	75	73	74.5
11	82	75.5	73.5	71.5	74.5
12	79	77	78	73	75.5
13	83	79	78.5	73	74.5
14	85.5	78	73.5	71.5	76
15	82	75.5	76	73	76
16	83	78.5	76	71.5	74.5
17	85	77	73.5	72	75
18	83.5	77.5	77.5	73	75.5
19	81.5	79	73.5	71.5	73.5
20	85	77	75	72	74.5
Average value	83.35	77.575	75.6	72.4	74.875

5.2 Example B

Example A's distribution points are only eight, in order to further test the LinWPSO, SAPSO and RandWPSO performance, the paper randomly generated a problem with the size of 20 distribution systems. Distribution centre coordinates (50 km, 50 km), the position coordinates of the customer (x km, y km), x, y is real number between 0–100, goods demand is the random number between 0 to 2, coordinates of 20 customers of their cargo demand are shown in Table IV. The deadweight of the vehicle of the distribution centre is 8 tons, the number of vehicles of the distribution centre is 3. Required are reasonable arrangements for vehicle delivery routes to get the shortest delivery mileage. For simplicity, the distance between customers and from the customer to the distribution centre is calculated using straight-line distance.

Table IV: Customers coordinates and their demand.

Customer number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Abscissa x (km)	42	57	41	70	96	91	62	72	76	26
Ordinate y (km)	14	32	99	47	58	88	79	10	8	54
Demand q (t)	0.3	0.4	1.2	1.5	0.8	1.3	1.1	0.6	1.2	0.4

Customer number	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Abscissa x (km)	55	93	45	28	78	10	16	11	97	56
Ordinate y (km)	39	28	74	96	9	27	71	55	31	94
Demand q (t)	0.9	1.3	0.7	1.9	1.7	1.1	1.5	1.6	1.2	1.5

The study included 20 clients; the full array number of customers up to 2.433×10^{18}, limited by the time, the problem cannot be solved simply by the Exhaustive method.

This calculation example is solved using 4 kinds of algorithms of the standard PSO, LinWPSO, SAPSO and RandWPSO respectively. Parameters set is as following: the particle dimension $D = 20 + 3 - 1 = 22$, the number of particles is 40. Penalty function $R = 10^{10}$, evolution algebra takes 500 iterations.

Standard PSO, LinWPSO, SAPSO and RandWPSO, these four algorithms all conduct 10 times calculation. Result of the operation is shown in Fig. 18, which shows that the average of 10 times calculation obtained from the standard PSO was 621.8; the average of 10 times calculation obtained from the LinWPSO was 616.6; the average of 10 times calculation obtained from the SAPSO was 615.9; the average of 10 times calculation obtained from the RandWPSO was 616.3.

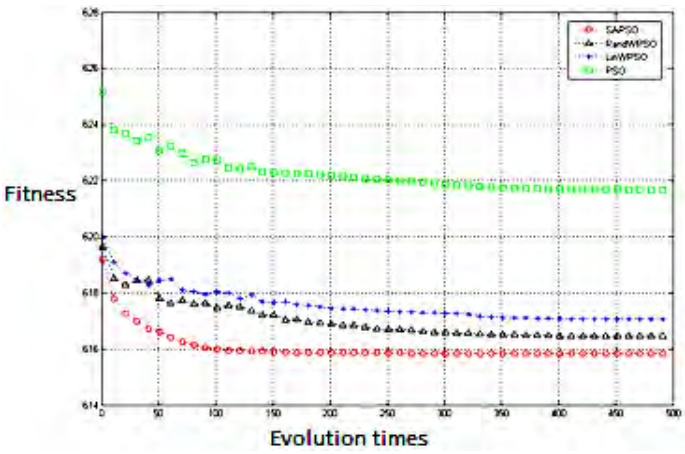

Figure 18: Results comparison chart of Standard PSO, LinWPSO, SAPSO and RandWPSO.

Fig. 18 shows that when the problem size increases, performance of LinWPSO, SAPSO and RandWPSO is superior to standard PSO, at the same time SAPSO has the most optimal performance. Use of the SAPSO algorithm for solving the distribution vehicle routing problem can greatly shorten delivery mileage, which is an effective way to solve the distribution vehicle routing problem.

The optimal path obtained by using SAPSO algorithm is:

Vehicle 1: Distribution Centre → Customer 1 → Customer 8 → Customer 15 → Customer 12 → Customer 19 → Customer 4 → Customer 2 → Customer 11 → Distribution Centre

Vehicle 2: Distribution Centre → Customer 13 → Customer 20 → Customer 7 → Customer 6 → Customer 5 → Customer 9 → Distribution Centre

Vehicle 3: Distribution Centre → Customer 10 → Customer 16 → Customer 18 → Customer 17 → Customer 14 → Customer 3 → Distribution Centre

The total distribution mileage is 615.9 km.

Fig. 19 shows the vehicles path.

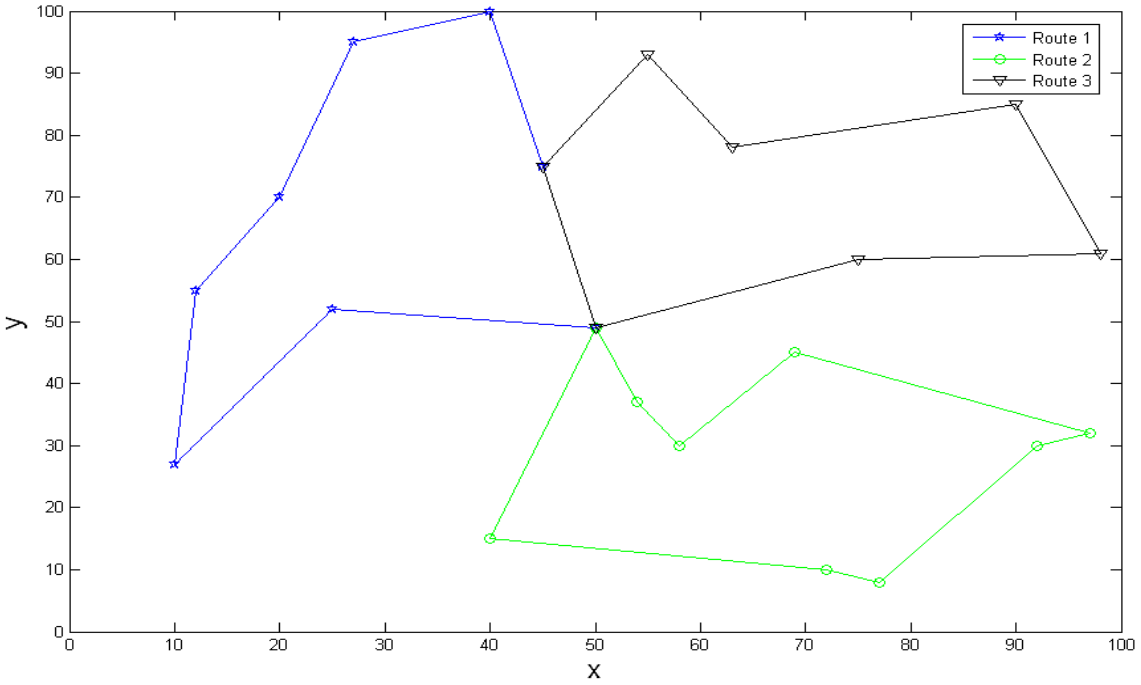

Figure 19: Distribution roadmap.

6. CONCLUSION

This paper mainly discusses the application of the PSO for logistics distribution vehicle routing problem. Combined with the characteristics of logistics and distribution a mathematical model of the distribution vehicle routing problem was established. Three kinds of optimization strategy to optimize the particle swarm algorithm are introduced; constructed are three different improved particle swarm algorithms of LinWPSO, SAPSO and RandWPSO. Respectively, using standard PSO, LinWPSO, SAPSO and RandWPSO to solve example of the logistics distribution routing problem, example analysis shows that the performance when using the LinWPSO, SAPSO and RandWPSO proposed in this paper to solve vehicle routing problem is better than standard PSO. SAPSO has the optimal performance, which can effectively solve the logistics distribution vehicle routing problem, after the problem size increases, optimization advantage of the SAPSO was fully displayed, and solving vehicle routing problem with SAPSO can greatly shorten the delivery mileage.

7. ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation (No. 71240002), the "11th Five-Year Plan" project of Beijing Philosophy and Social Science (No. 10BaJG384), the Beijing Natural Science Foundation (No. 9123025), the Beijing Philosophical Social Science Project (No. 11JGB077), the Beijing Natural Science Foundation (No. 9122003), the Beijing Municipal Education Commission Foundation of China (No. KM201110038002), the Scientific Research Project of Capital University of Economics and Business (No. 2013XJG022), the 2013 Teaching Reform Project of Capital University of Economics and Business, the Scientific Research Improvement Project of the Beijing Municipal Education Commission, the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions (Project name: Decision tree generation algorithm and its optimization of incomplete information systems).

REFERENCES

[1] Xiang, T.; Liao, X. F.; Wong, K. W. (2007). An improved particle swarm optimization algorithm combined with piecewise linear chaotic map, *Applied Mathematics and Computation*, Vol. 190, No. 2, 1637-1645, doi:10.1016/j.amc.2007.02.103

[2] Jie, J.; Zeng, J. C.; Han, C. Z. (2008). Self-organized particle swarm optimization based on feedback control of diversity, *Journal of Computer Research and Development*, Vol. 45, No. 3, 464-471

[3] Shen, Y. X.; Wang, G. Y.; Zeng, C. H. (2011). Correlative particle swarm optimization model, *Journal of Software*, Vol. 22, No. 4, 695-708, doi:10.3724/SP.J.1001.2011.03728

[4] Fan, S.-K. S.; Zahara, E. (2007). A hybrid simplex search and particle swarm optimization for unconstrained optimization, *European Journal of Operational Research*, Vol. 181, No. 2, 527-548, doi:10.1016/j.ejor.2006.06.034

[5] Liu, L.; Zhong, W. M.; Qian, F. (2010). An improved chaos-particle swarm optimization algorithm, *Journal of East China University of Science and Technology: Natural Science Edition*, Vol. 36, No. 2, 267-272

[6] Chang, Y. P. (2010). Integration of SQP and PSO for optimal planning of harmonic filters, *Expert Systems with Applications*, Vol. 37, No. 3, 2522-2530, doi:10.1016/j.eswa.2009.08.025

[7] Xiao, J. M.; Li, J. J.; Wang, X. H. (2009). Convergence analysis of particle swarm optimization and its improved algorithm based on gradient, *Control and Decision*, Vol. 24, No. 4, 560-564

[8] Zhao, X. C. (2010). A perturbed particle swarm algorithm for numerical optimization, *Applied Soft Computing*, Vol. 10, No.1, 119-124, doi:10.1016/j.asoc.2009.06.010

[9] Romeijn, H. E.; Shu, J.; Teo, C.-P. (2007). Designing two-echelon supply networks, *European Journal of Operational Research*, Vol. 178, No. 2, 449-462, doi:10.1016/j.ejor.2006.02.016

[10] Ko, H. J.; Evans, G. W. (2007). A genetic algorithm-based heuristic for the dynamic integrated forward/reverse logistics network for 3PLs, *Computers & Operations Research*, Vol. 34, No. 2, 346-366, doi:10.1016/j.cor.2005.03.004

[11] Amiri, A. (2006). Designing a distribution network in a supply chain system: Formulation and efficient solution procedure, *European Journal of Operational Research*, Vol. 171, No. 2, 567-576, doi:10.1016/j.ejor.2004.09.018

[12] Gabor, A. F.; van Ommeren, J.-K. C. W. (2006). Approximation algorithms for facility location problems with a special class of subadditive cost functions, *Theoretical Computer Science*, Vol. 363, No. 3, 289-300, doi:10.1016/j.tcs.2006.04.013

[13] Keskin, B. B.; Uster, H. (2007). Meta-heuristic approaches with memory and evolution for a multi-product production/distribution system design problem, *European Journal of Operational Research*, Vol. 182, No. 2, 663-682, doi:10.1016/j.ejor.2006.07.034