An Improved Particle Swarm Optimization Algorithm and Its Application

Xuesong Yan[1], Qinghua Wu[2,3], Hanmin Liu[4] and Wenzhi Huang[2,3]

[1] **School of Computer Science, China University of Geosciences
Wuhan, Hubei 430074, China**

[2] **Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology
Wuhan, Hubei 430073, China**

[3] **School of Computer Science and Engineering, Wuhan Institute of Technology
Wuhan, Hubei 430073, China**

[4] **Wuhan Institute of Ship Building Technology
Wuhan, Hubei 430050, China**

Abstract

In this paper, aim at the disadvantages of standard Particle Swarm Optimization algorithm like being trapped easily into a local optimum, we improves the standard PSO and proposes a new algorithm to solve the overcomes of the standard PSO. The new algorithm keeps not only the fast convergence speed characteristic of PSO, but effectively improves the capability of global searching as well. Compared with standard PSO on the Benchmarks function, the results show that the new algorithm is efficient, we also use the new algorithm to solve the TSP and the experiment results show the new algorithm is effective for the this problem.

Keywords: *Particle Swarm Optimization, Traveling Salesman Problem, Particle, Convergence.*

1. Introduction

Particle Swarm Optimization (PSO) algorithm was an intelligent technology first presented in 1995 by Eberhart and Kennedy, and it was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities [1]. If we compare PSO with Genetic Algorithms (GAs), we may find that they are all maneuvered on the basis of population operated. But PSO doesn't rely on genetic operators like selection operators, crossover operators and mutation operators to operate individual, it optimizes the population through information exchange among individuals. PSO achieves its optimum solution by starting from a group of random solution and then searching repeatedly. Once PSO was presented, it invited widespread concerns among scholars in the optimization fields and shortly afterwards it had become a studying focus within only several years. A number of scientific achievements had emerged in these fields [2-4]. PSO was proved to be a sort of high efficient optimization algorithm by numerous research and experiments [5]. PSO is a meta-heuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, meta-heuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-Newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc.

The traveling salesman problem (TSP) [6] is one of the most widely studied NP-hard combinatorial optimization problems. Its statement is deceptively simple, and yet it remains one of the most challenging problems in Operational Research. The simple description of TSP is: Give a shortest path that covers all cities along. Let $G=(V;E)$ be a graph where V is a set of vertices and E is a set of edges. Let $C=(c_{ij})$ be a distance (or cost) matrix associated with E. The TSP requires determination of a minimum distance circuit (Hamiltonian circuit or cycle) passing through each vertex once and only once. C is said to satisfy the triangle inequality if and only if $c_{ij}+c_{jk}\geq c_{ik}$ for $i,j,k\in V$.

Due to its simple description and wide application in real practice such as Path Problem, Routing Problem and Distribution Problem, it has attracted researchers of various domains to work for its better solutions. Those traditional algorithms such as Cupidity Algorithm, Dynamic Programming Algorithm, are all facing the same obstacle, which is when the problem scale N reaches to a certain degree, the so-called "Combination Explosion" will occur. For example, if $N = 50$, then it will take 5×10^{48} years under a super mainframe executing 100 million instructions per second to reach its approximate best solution.

A lot of algorithms have been proposed to solve TSP [7-12]. Some of them (based on dynamic programming or branch and bound methods) provide the global optimum solution. Other algorithms are heuristic ones, which are much faster, but they do not guarantee the optimal solutions. There are well known algorithms based on 2-opt or 3-opt change operators, Lin-Kerninghan algorithm (variable change) as well algorithms based on greedy principles (nearest neighbor, spanning tree, etc). The TSP was also approached by various modern heuristic methods, like simulated annealing, evolutionary algorithms and tabu search, even neural networks.

This paper improves the disadvantages of standard PSO being easily trapped into a local optimum and proposed a new algorithm called improved PSO (IPSO) which proves to be more simply conducted and with more efficient global searching capability, then use the new algorithm for traveling salesman problem.

2. Basic Particle Swarm Optimization Algorithm

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered. Formally, let $f : R^n \rightarrow R$ be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the objective function value of the given candidate solution. The gradient of f is not known. The goal is to find a solution a for which $f(a) \le f(b)$ for all b in the search-space, which would mean a is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

PSO was presented under the inspiration of bird flock immigration during the course of finding food and then be used in the optimization problems. In PSO, each optimization problem solution is taken as a bird in the searching space and it is called "particle". Every particle has a fitness value which is determined by target functions and it has also a velocity which determines its destination and distance. All particles search in the solution space for their best positions and the positions of the best particles in the swarm. PSO is initially a group of random particles (random solutions), and then the optimum solutions are found by repeated searching. In every iteration, a particle will follow two bests to renew itself: the best position found for a particle called p_{best}; the best position found for the whole swarm called g_{best}. All particles will determine following steps through the best experiences of individuals themselves and their companions.

For particle id, its velocity and its position renewal formula are as follows:

$$V_{id}' = \omega V_{id} + \eta_1 rand()(P_{idb} - X_{id}) + \eta_2 rand()(P_{gdb} - X_{id}) \quad (1)$$

$$X_{id}' = X_{id} + V_{id}' \quad (2)$$

In here: ω is called inertia weight, it is a proportion factor that is concerned with former velocity, $0 < \omega < 1$, η_1 and η_2 are constants and are called accelerating factors, normally $\eta_1 = \eta_2 = 2$, $rand()$ are random numbers, X_{id} represents the position of particle id; V_{id} represents the velocity of particle id; P_{id}, P_{gd} represent separately the best position particle id has found and the position of the best particles in the whole swarm.

In formula(1), the first part represents the former velocity of the particle, it enables the particle to possess expanding tendency in the searching space and thus makes the algorithm be more capable in global searching; the second part is called cognition part, it represents the process of absorbing individual experience knowledge on the part of the particle; the third part is called social part, it represents the process of learning from the experiences of other particles on the part of certain particle, and it also shows the information sharing and social cooperation among particles.

The flow of PSO can briefly describe as following: First, to initialize a group of particles, e.g. to give randomly each particle an initial position X_i and an initial velocity V_i, and then to calculate its fitness value f. In every iteration, evaluated a particle's fitness value by analyzing the velocity and positions of renewed particles in formula (1) and (2). When a particle finds a better position than previously, it will mark this coordinate into vector P1, the vector difference between P1 and the present position of the particle will randomly be added to next velocity vector, so that the following renewed particles will search around this point, it's also called in formula (1) cognition component. The weight difference of the present position of the particle swarm and the best position of the swarm P_{gd} will also be added to velocity vector for adjusting the next population velocity. This is also called in formula (1) social component. These two adjustments will enable particles to search around two bests.

The most obvious advantage of PSO is that the convergence speed of the swarm is very high, scholars like Clerc [13] has presented proof on its convergence. In order to verify the convergence speed of the PSO algorithm, we selected four benchmarks function and compared the results with traditional genetic algorithm (GA).

F1: Schaffer function

$$\min f(x_i) = 0.5 - \frac{(\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5)}{[1 + 0.001(x_1^2 + x_2^2)]^2},$$
$$-10 \le x_i \le 10$$

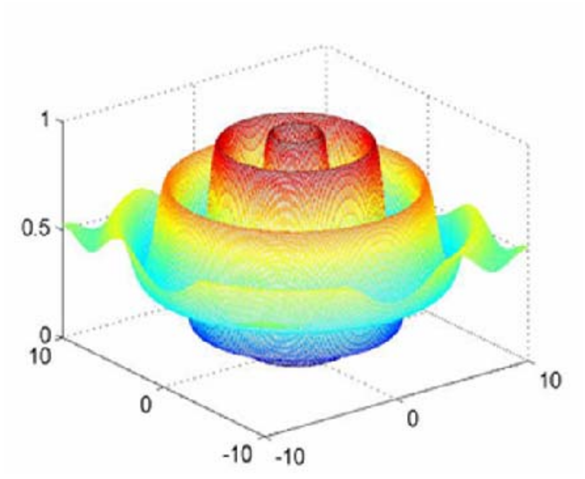


Fig. 1 Schaffer function

In this function the biggest point is in the situation where xi= (0, 0) and the global optimal value is 1.0, the largest in the overall points for the center, and 3.14 for the radius of a circle on the overall situation from numerous major points of the uplift. This function has a strong shock; therefore, it is difficult to find a general method of its global optimal solution.

F2: Shubert function

$$\min f(x, y) = \left\{ \sum_{i=1}^{5} i \cos\left[(i+1)x + i\right] \right\} \times$$
$$\left\{ \sum_{i=1}^{5} i \cos\left[(i+1)y + i\right] \right\}, x, y \in [-10, 10]$$

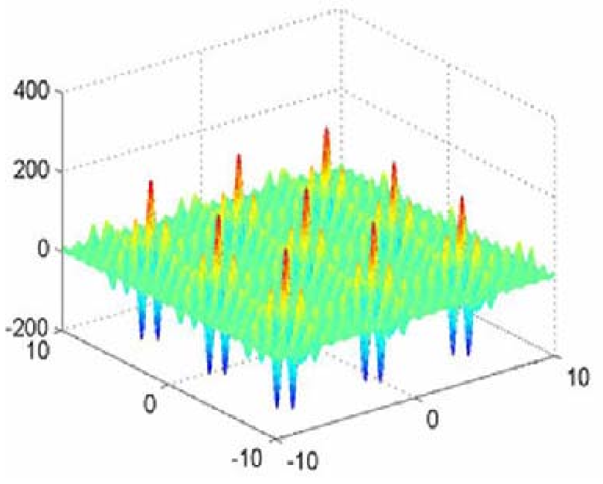


Fig. 2 Shubert function

This function has 760 local minimum and 18 global minimum, the global minimum value is -186.7309.

F3: Hansen function

$$\min f(x, y) = \sum_{i=1}^{5} i \cos((i-1)x + i) \sum_{j=1}^{5} j \cos((j+1)y + j),$$
$$x, y \in [-10, 10]$$

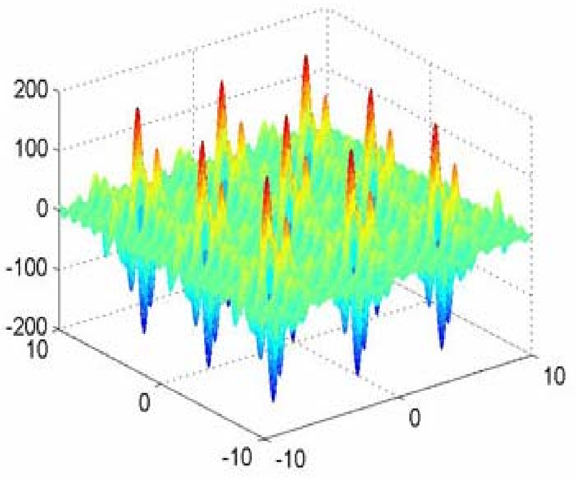


Fig. 3 Hansen function

This function has a global minimum value -176.541793, in the following nine point (-7.589893, -7.708314)、(-7.589893, -1.425128)、(-7.589893, 4.858057)、(-1.306708, -7.708314)、(-1.306708, -1.425128)、(-1.306708, 4.858057)、(4.976478, -7.708314)、(4.976478, -7.708314)、(4.976478, 4.858057) can get this global minimum value, the function has 760 local minimum.

F4: Camel function

$$\min f(x,y)=\left(4-2.1x^2+\frac{x^4}{3}\right)x^2+xy+\left(-4+4y^2\right)y^2,$$

$$x,y\in[-100,100]$$

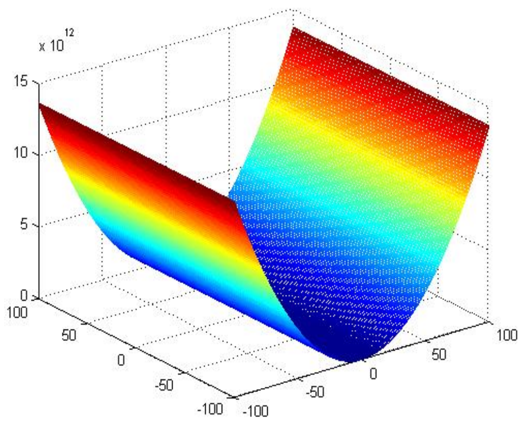


Fig. 4 Camel function

Camel function has 6 local minimum (1.607105, 0.568651)、(-1.607105, -0.568651)、(1.703607, -0.796084)、(-1.703607, 0.796084)、(-0.0898,0.7126) and (0.0898,-0.7126), the (-0.0898,0.7126) and (0.0898,-0.7126) are the two global minimums, the value is -1.031628.

Table 1: Experiment results comparison (100 runs for each case)

Function	***Algorithm***	***Convergence Times***	***Optimal Solution***
F1	GA	72	1.0000000
	PSO	75	1.0000000
F2	GA	75	-186.730909
	PSO	80	-186.730909
F3	GA	85	-176.541793
	PSO	90	-176.541793
F4	GA	23	-1.031628
	PSO	56	-1.031628

In the experiment, each case is repeated for 100 times. Table 1 shows the statistics of our experimental results in terms of accuracy of the best solutions. GA found the known optimal solution to F1 72 times out of 100 runs, found the known optimal solution to F2 75 times out of 100 runs, found the known optimal solution to F3 85 times out of 100 runs, found the known optimal solution to F4 23 times out of 100 runs; PSO algorithm is efficiency for the four cases: found the known optimal solution to F1 75 times out of 100 runs, found the known optimal solution to F2 80 times out of 100 runs, found the known optimal solution to F3 90 times out of 100 runs and found the known optimal solution to F4 56 times out of 100 runs.

3. Improved Particle Swarm Optimization Algorithm

In the standard PSO algorithm, the convergence speed of particles is fast, but the adjustments of cognition component and social component make particles search around P_{gd} and P_{id}. According to velocity and position renewal formula, once the best individual in the swarm is trapped into a local optimum, the information sharing mechanism in PSO will attract other particles to approach this local optimum gradually, and in the end the whole swarm will be converged at this position. But according to velocity and position renewal formula (1), once the whole swarm is trapped into a local optimum, its cognition component and social component will become zero in the end; still, because $0<\omega<1$ and with the number of iteration increase, the velocity of particles will become zero in the end, thus the whole swarm is hard to jump out of the local optimum and has no way to achieve the global optimum. Here a fatal weakness may result from this characteristic. With constant increase of iterations, the velocity of particles will gradually diminish and reach zero in the end. At this time, the whole swarm will be converged at one point in the solution space, if g_{best} particles haven't found g_{best}, the whole swarm will be trapped into a local optimum; and the capacity of swarm jump out of a local optimum is rather weak. In order to get through this disadvantage, in this paper we presents a new algorithm based on PSO. In order to avoid being trapped into a local optimum, the new PSO adopts a new information sharing mechanism. We all know that when a particle is searching in the solution space, it doesn't know the exact position of the optimum solution. But we can not only record the best positions an individual particle and the whole swarm have experienced, we can also record the worst positions an individual particle and the whole swarm have experienced, thus we may make individual particles move in the direction of evading the worst positions an individual particle and the whole flock have experienced, this will surely enlarge the global searching space of particles and enable them to avoid being trapped into a local optimum too early, in the same time, it will improve the possibility of finding g$_{best}$ in the searching space. In the new strategy, the particle velocity and position renewal formula are as follows:

$$V_{id}^{'} = \omega V_{id} + \eta_1 rand()(X_{id} - P_{idw}) + \eta_2 rand()(X_{id} - P_{gdw}) \tag{3}$$

$$X_{id}^{'} = X_{id} + V_{id}^{'} \tag{4}$$

In here: P_{idw}, P_{gdw} represent the worst position particle id has found and the worst positions of the whole swarm has found.

In standard PSO algorithm, the next flying direction of each particle is nearly determined; it can fly to the best individual and the best individuals for the whole swarm. From the above conclusion we may easily to know it will be the danger for being trapped into a local optimum. In order to decrease the possibility of being trapped into the local optimum, the new PSO introduces genetic selection strategy: To set particle number in the swarm as m, father population and son population add up to 2m. To select randomly q pairs from m; as to each individual particle i, if the fitness value of i is smaller than its opponents, i will win out and then add one to its mark, and finally select those particles which have the maximum mark value into the next generation. The experiments conducted show that this strategy greatly reduces the possibility of being trapped into a local optimum when solving certain functions.

The flow of the IPSO is as follows:

Step 1: to initialize randomly the velocity and position of particles;

Step 2: to evaluate the fitness value of each particle;

Step 3: as to each particle, if its fitness value is smaller than the best fitness value P_{idb}, renew the best position P_{idb} of particle id; or else if its fitness value is bigger than the worst fitness value P_{idw}, renew P_{idw};

Step 4: as to each particle, if its fitness value is smaller than the best whole swarm fitness value P_{gdb}, renew the best fitness value P_{gdb} of particle id; or else if bigger than the worst whole swarm fitness value P_{gdw}, renew P_{gdw};

Step 5: as to each particle,

1) To produce new particle t by applying formula (1) (2),

2) To produce new particle t' by applying formula (3) (4),

3) To make a comparison between t and t', then select the better one into the next generation;

Step 6: to produce next generation particles according to the above genetic selection strategy;

Step 7: if all the above steps satisfy suspension needs, suspend it; or turn to Step 3.

In order to verify the improvement of the new algorithm based on PSO, we select the same benchmark function the above have described. We run our algorithm and compare the results with traditional PSO. In the experiment, each case is repeated for 100 times. Table 2 shows the statistics of our experimental results in terms of accuracy of the best solutions. For the four cases PSO algorithm can found the known optimal solution to F1 75 times out of 100 runs, found the known optimal solution to F2 80 times out of 100 runs, found the known optimal solution to F3 90 times out of 100 runs and found the known optimal solution to F4 56 times out of 100 runs. Our new algorithm improved PSO algorithm is efficiency for the four cases: found the known optimal solution to F1 and F2 100 times out of 100 runs, found the known optimal solution to F3 97 times out of 100 runs and found the known optimal solution to F4 65 times out of 100 runs.

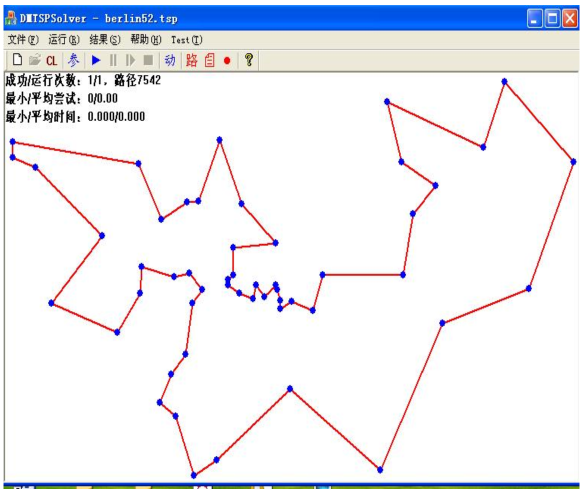
Table 2: Experiment results comparison (100 runs for each case)

Function	*Algorithm*	*Convergence Times*	*Optimal Solution*
F1	PSO	75	1.0000000
	IPSO	100	1.0000000
F2	PSO	80	-186.730909
	IPSO	100	-186.730909
F3	PSO	90	-176.541793
	IPSO	97	-176.541793
F4	PSO	56	-1.031628
	IPSO	65	-1.031628

4. Improved PSO for TSP

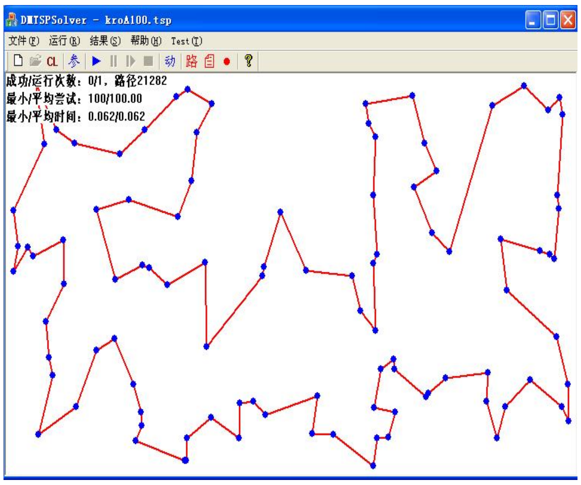
In order to verify the proposed algorithm is useful for the TSP, the experiment test we select 10 TSP test cases: berlin52, kroA100, kroA200, pr299, rd400, ali535, d657, rat783, u1060 and u1432. All experiments are performed on Intel Core(TM)2 Duo CPU 2.26GHz/4G RAM Laptop. In the experiments all test cases were chosen from TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95), and the optimal solution of each test case is known in the website.

We list the test cases' optimal solutions and compared with traditional genetic algorithm and PSO algorithm [19], the comparison results shown in Table 3. The comparison results demonstrate clearly the efficiency of our algorithm. Note that for the 10 test cases the optimum was found in all ten runs. The number of cities in these test cases varies from 52 to 1432 and Fig. 5 to Fig. 14 is the best solution with IPSO.


DMTSPSolver - berlin52.tsp
文件(F) 运行(R) 结果(S) 帮助(H) Test(T)
CL 参 动 路

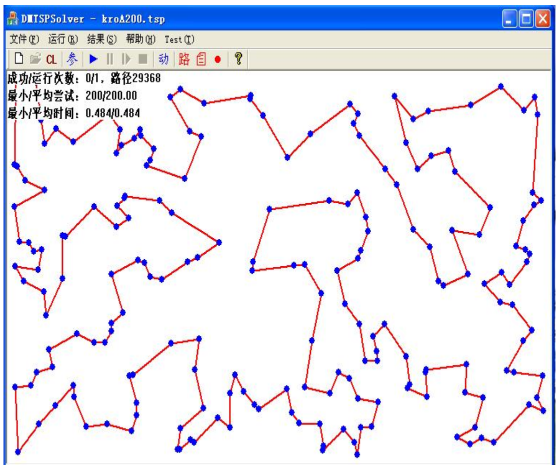
```
成功/运行次数: 1/1, 路径7542
最小/平均尝试: 0/0.00
最小/平均时间: 0.000/0.000
```

Fig. 5 berlin52


DMTSPSolver - kroA100.tsp
文件(F) 运行(R) 结果(S) 帮助(H) Test(T)
CL 参 动 路

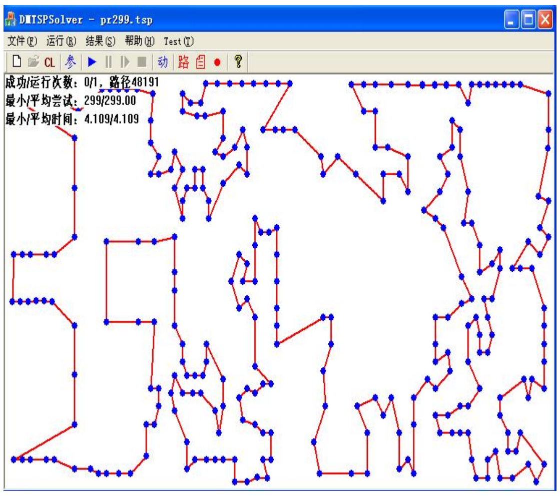
```
成功/运行次数: 0/1, 路径21282
最小/平均尝试: 100/100.00
最小/平均时间: 0.062/0.062
```

Fig. 6 kroA100


DMTSPSolver - kroA200.tsp
文件(F) 运行(R) 结果(S) 帮助(H) Test(T)
CL 参 动 路

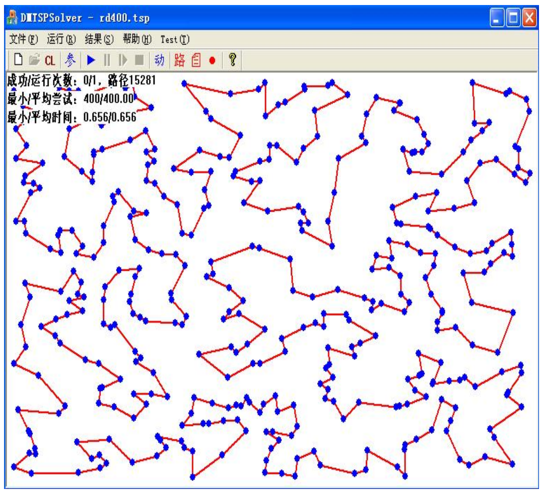
```
成功/运行次数: 0/1, 路径29368
最小/平均尝试: 200/200.00
最小/平均时间: 0.484/0.484
```

Fig. 7 kroA200



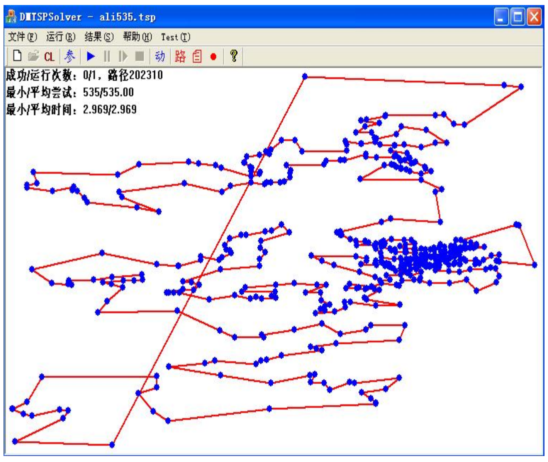
```
成功/运行次数: 0/1, 路径48191
最小/平均尝试: 299/299.00
最小/平均时间: 4.109/4.109
```

Fig. 8 pr299



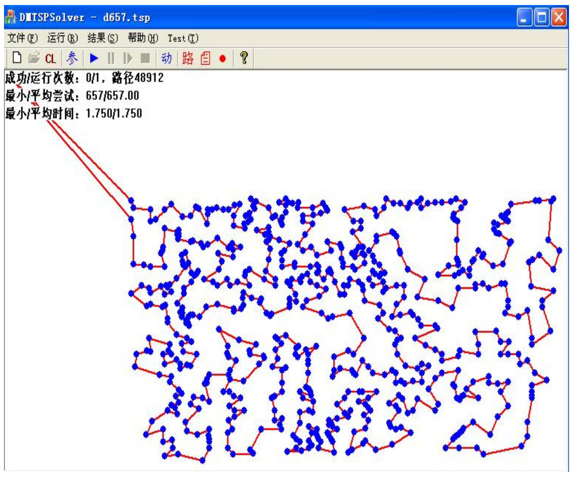
```
成功/运行次数: 0/1, 路径15281
最小/平均尝试: 400/400.00
最小/平均时间: 0.656/0.656
```

Fig. 9 rd400



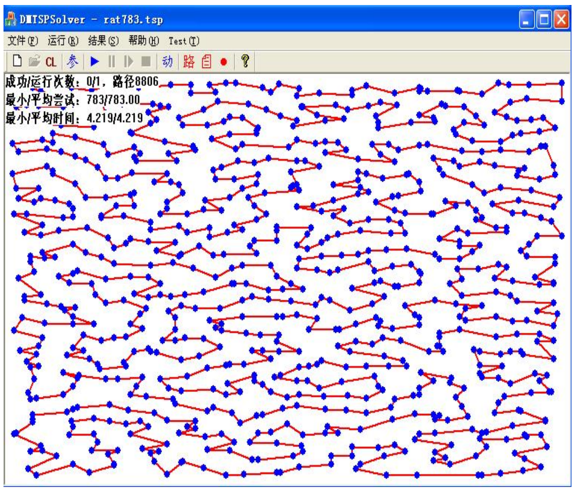
```
成功/运行次数: 0/1, 路径202310
最小/平均尝试: 535/535.00
最小/平均时间: 2.969/2.969
```

Fig. 10 ali535



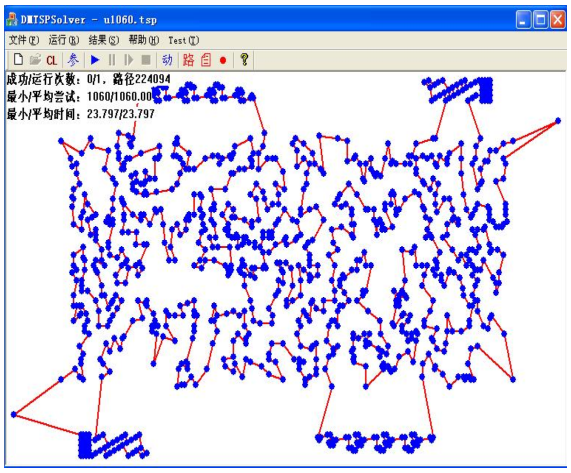
```
成功/运行次数：0/1，路径48912
最小/平均尝试：657/657.00
最小/平均时间：1.750/1.750
```

Fig. 11 d657



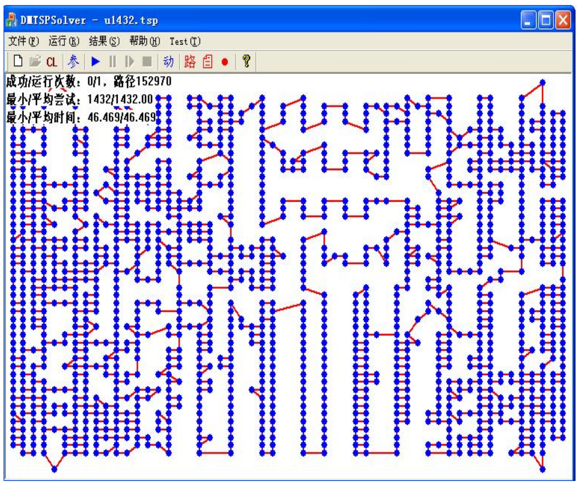
```
成功/运行次数：0/1，路径8806
最小/平均尝试：783/783.00
最小/平均时间：4.219/4.219
```

Fig. 12 rat783



```
成功/运行次数：0/1，路径224094
最小/平均尝试：1060/1060.00
最小/平均时间：23.797/23.797
```

Fig. 13 u1060



```
成功/运行次数：0/1，路径152970
最小/平均尝试：1432/1432.00
最小/平均时间：46.469/46.469
```

Fig. 14 u1432

We also compare our algorithm with other algorithms: Genetic Algorithm (GA), Ant Colony Optimization (ACO) and Ant System-Assisted Genetic Algorithm (ASAGA), the three algorithms are introduced in paper [14]. In the experiment, each case is repeated for 10 times. Table 4 shows the statistics of our experimental results in terms of accuracy of the best solutions. The known optimal solutions are taken from the TSP Library website the above has introduced. ACO failed in reaching the known optimal solution to any case. GA found the known optimal solution to Berlin52 7 times out of ten runs, but could not reach that for the other larger cases. ASAGA found the known optimal solution to berlin52 7 times out of ten runs, found the known optimal solution to kroA100 5 times out of ten runs and found the known optimal solution to kroA200 only one time. Our algorithm is efficiency for the three cases: found the known optimal solution to berlin52 10 times out of ten runs, found the known optimal solution to kroA100 9 times out of ten runs and found the known optimal solution to kroA200 only 8 times.

4. Conclusions

This paper introduce a new algorithm based on the standard PSO algorithm, for the standard PSO algorithm the new algorithm has done two improvements: 1. By introducing a new information sharing mechanism, make particles moved on the contrary direction of the worst individual positions and the worst whole swarm positions, thus enlarge global searching space and reduce the possibility of particles to be trapped into a local optimum; 2. By introducing genetic selection strategy, decreased the possibility of being trapped into a local optimum. Compared with the standard PSO algorithm, the new algorithm enlarges the searching space and the complexity is not high. We use the proposed algorithm for solving the combinatorial problem: TSP, the new algorithm shows great efficiency in solving TSP with the problem scale from 52 to 1432. By analyzing the testing results, we reach the conclusion: in the optimization precision and the optimization speed, the new algorithm is efficiency than the traditional PSO algorithm and the new algorithm is more efficient than traditional algorithms in coping with the TSP.

Acknowledgments

This paper is supported by Natural Science Foundation of China. (No.61272470 and No.61203307), National Civil Aerospace Pre-research Project of China, the Provincial Natural Science Foundation of Hubei (2012FFB04101), the Fundamental Research Founds for National University, China University of Geosciences(Wuhan), Science Research Foundation of Wuhan Institute of Technology (No.12126011) and Youth Foundation of Wuhan Institute of Technology (Q201107).

References

[1] J. Kennedy and R. C.Eberhart, “Particle Swarm Optimization”, IEEE International Conference on Neural Networks, 1995, pp.1942-1948.
[2] Clare M, Kennedy J, “The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space”, IEEE Trans. on Evolution2ary Computation, vol.6(1), 2002, pp.58-73.
[3] C.A.Coello and M.S.Lechuga, Mopso, “A proposal for multiple objective particle swarm optimization”, In IEEE Proceedings World Congress on Computational Intelligence, 2002, pp.1051-1056.
[4] J.Kennedy, “The particle swarm: social adaptation of knowledge”, In Proc. IEEE Conf. on evolutionary computation, 1997, pp.3003-3008.
[5] E. Oscan and C. K.Mohan, “Analysis of A Simple Particle Swarm Optimization System”, Intelligence Engineering Systems Through Artificial Neural Networks, 1998, pp.253-258.
[6] Durbin R, Willshaw D, “An Anlaogue Approach to the Traveling Salesman Problem Using an Elastic Net Approach”, Nature, 326, 6114, 1987, pp.689-691.
[7] Tao Guo and Zbigniew Michalewize, “Inver-Over operator for the TSP”, In Parallel Problem Sovling from Nature(PPSN V), Springer-Verlag press, 1998, pp.803-812.
[8] Zhangcan Huang, Xiaolin Hu and Siduo Chen, “Dynamic Traveling Salesman Problem based on Evolutionary Computation”, In Congress on Evolutionary Computation(CEC’01), IEEE Press, 2001, pp.1283-1288.
[9] Aimin Zhou, Lishan Kang and Zhenyu Yan, “Solving Dynamic TSP with Evolutionary Approach in Real Time”, In Congress on Evolutionary Computation(CEC’03), 2003, pp.951-957.
[10] Hui.Yang, Lishan.Kang and Yuping.Chen, “A Gene-pool Based Genetic Algorithm for TSP”, Wuhan University Journal of Natural Sciences, 8(1B), 2003, pp.217-223.
[11] Xuesong Yan, Lishan.Kang, “An Approach to Dynamic Traveling Salesman Problem”, Proceedings of the Third International Conference on Machine Learning and Cybernetics,2004, pp. 2418-2420.
[12] Xuesong Yan, Aimin Zhou, Lishan Kang, “TSP Problem Based on Dynamic Environment”, Proceedings of the 5th World Congress on Intelligent Control and Automation, 2004, pp.2271-2274.
[13] M.Clerc and J.Kennedy, “The Particle Swarm: Explosion, Stability and Convergence in a Multi-Dimensional Complex Space”, IEEE Trans. on Evolutionary Computation, Vol.6, 2002, pp.58-73.
[14] Gaifang Dong, Xueliang Fu, Heru Xue, "An Ant System-Assisted Genetic Algorithm For Solving The Traveling Salesman Problem", International Journal of Advancements in Computing Technology, Vol. 4, No. 5, 2012, pp.165 -171.
[15] Xuesong Yan, Hui Li, “A Fast Evolutionary Algorithm for Combinatorial Optimization Problems”, Proceedings of the Fourth International Conference on Machine Learning and Cybernetics,2005, pp.3288-3292.
[16] Xuesong Yan, Qing Hua Wu, “A New Optimizaiton Algorithm for Function Optimization”, Proceedings of the 3rd International Symposium on Intelligence Computation & Applications, 2009, pp. 144-150.
[17] Xue Song Yan, Qing Hua Wu, Cheng Yu Hu, Qing Zhong Liang, “Circuit Design Based on Particle Swarm Optimization Algorithms”, Key Engineering Materials, Vols. 474-476, 2011, pp.1093-1098.
[18] Xuesong Yan, Qinghua Wu, Can Zhang, Wei Chen Wenjing Luo, Wei Li, “An Efficient Function Optimization Algorithm based on Culture Evolution”, International Journal of Computer Science Issues, Vol. 9, Issue 5, No. 2, 2012, pp.11-18.
[19] Xuesong Yan, Can Zhang, Wenjing Luo, Wei Li, Wei Chen, Hanmin Liu, “Solve Traveling Salesman Problem Using Particle Swarm Optimization Algorithm”, International Journal of Computer Science Issues, Vol. 9, Issue 6, No. 2, 2012, pp.264-271.

Xuesong Yan associate professor received him B.E. degree in Computer Science and Technology in 2000 and M.E. degree in Computer Application from China University of Geosciences in 2003, received he Ph.D. degree in Computer Software and Theory from Wuhan University in 2006. He is currently with School of Computer Science, China University of Geosciences, Wuhan, China and now as a visiting scholar with Department of Computer Science, University of Central Arkansas, Conway, USA. He research interests include evolutionary computation, data mining and computer application.

Qinghua Wu lecturer received her B.E. degree in Computer Science and Technology in 2000, M.E. degree in Computer Application in 2003 and Ph.D. degree in Earth Exploration and Information Technology Theory from China University of Geosciences in 2011. She is currently with School of Computer

Science and Engineering, Wuhan Institute of Technology, Wuhan, China. Her research interests include evolutionary computation, image processing and computer application.

Hanmin Liu associate professor. He is currently as a Ph.D candidate of School of Computer Science, China University of Geosciences, Wuhan, China. He research interests include evolutionary computation and applications.

Wenzhi Huang Lecturer. She is currently with School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China. Her research interests include image processing and computer application.

Table 3: Optimal results comparison

Test Cases	*Optimal in TSPLIB*	*GA*	*PSO*	*IPSO*
Berlin52	7542	7542	7542	7542
kroA100	21282	21315	21310	21282
kroA200	29368	30168	29968	29368
Pr299	48191	48568	48540	48191
Rd400	15281	15135	15135	15281
Ali535	202310	242310	231120	202310
D657	48912	50912	50612	48912
Rat783	8806	8965	8905	8806
U1060	224094	279094	269908	224094
U1432	152970	182780	177890	152970

Table 4: Experiment results comparison (10 runs for each case)

Test Cases	*Known Optimal*	*GA*	*ACO*	*ASAGA*	*Our Algorithm*
Berlin52	7542	7542(7)	7784	7542(7)	7542(10)
kroA100	21282	21315	21637	21282(5)	21282(9)
KroA200	29368	29694	30143	29638(1)	29368(8)