Journal of Information & Computational Science 11:3 (2014) 691–698 February 10, 2014
Available at http://www.joics.com

A Novel Quantum Particle Swarm Optimization Algorithm Based on Euclidean-distance Division Policy ⋆

Jinyin Chen [a], Ying Li [b,∗], Dongyong Yang [c]

[a] *College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China*

[b] *College of Computer Science, Zhejiang University, Hangzhou 310000, China*

[c] *College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China*

Abstract

On the basis of the standard Particle Swarm Optimization (PSO) algorithm, we put forward a novel Quantum Particle Swarm Optimization (QPSO) algorithm. For the shortcomings of QPSO, we got a new kind of quantum particle swarm optimization with Euclidean-distance division policy and a detailed description for it. The experimental results indicated that the improved algorithm with Euclidean-distance division policy has been greatly improved in performance compared with current other PSOs.

Keywords: QPSO (Quantum Particle Swarm Optimization); Iteration; Division Strategy; Wave Function; Convergence; Global Optimum

1 Introduction

In recent years, evolutionary computation is an important research direction in the field of artificial intelligence research; also it is an important element of intelligent information processing. Swarm intelligence is a kind of evolutionary algorithms to solve the optimization problem especially effective means of complex optimization problems.

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm based on swarm intelligence, which has great potential in function optimization, neural networks, pattern classification, fuzzy system control as well as some other areas. But the search space of the particle is a limited area in PSO algorithm, it is impossible to cover the whole feasible space, so we can't guarantee to converge to the global optimal solution with probability 1.

From the point of view of quantum mechanics, Sun and others put forward a new model of the PSO algorithm after they learned the research on particle convergence behavior by Clerc and others in 2004. They thought the particles have quantum behavior and put forward the Quantum Particle Swarm Optimization (QPSO) according to this new model. QPSO algorithm performance

⋆Project supported by the National Technology Support Program under grant No. 2011BAH16B04.

∗Corresponding author.
Email address: cnliying2013@gmail.com (Ying Li).

DOI: 10.12733/jics20102558

has been greatly improved compared with the PSO algorithm. However QPSO algorithm is easy to fall into local convergence when solving multimodal optimization problems. Therefore, on the basis of the QPSO algorithm combined with the idea of division strategy, we designed a algorithm with Euclidean-distance division policy abbreviated to SD-QPSO. Experiments indicate that, the SD-QPSO algorithm is better than QPSO in searching the global optimum value and has a better performance on convergence.

2 Related Work

2.1 PSO Algorithm

Each particle in the solution space will be random initialized before starting the search of particle swarm in PSO algorithms, and meanwhile put a corresponding speed. Particle group will search in N dimensional space, and each of particle's position $x_i = (x_{i1}, x_{i2}, ..., x_{iN})$ can be expressed as a solution of the problem, the corresponding speed of the particle is $v_i = (v_{i1}, v_{i2}, ..., v_{iN})$. The particles constantly adjust their position to search for a new solution. Every particle will save their optimal solution in the searching, which we denoted $P_i = (P_{i1}, P_{i2}, ..., P_{iN})$ or referred as *Pbest*. Also the particle group's optimal solution when in t searching will be recorded too, denoted by $P_g = (P_{g1}, P_{g2}, ..., P_{gN})$ or referred as *gbest*. In the particle's constant iterative process, the generation of the particles update position and speed according to the next equations:

$$V_{id}(k+1) = \omega \cdot V_{id}(k) + c_1 \cdot r_1 \cdot (P_{id} - x_{id}(k)) + c_2 \cdot r_2 \cdot (P_{gd} - x_{id}(k)) \tag{1}$$

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1) \tag{2}$$

where $i = 1, 2, ..., M$, M is the size of the particle swarm, $V_{id}(k)$ is the d dimensional component of particle i with times of iteration. In the same way, $X_{id}(k)$ is the d dimensional component of particle i with k times of iteration. P_{id} is the optimal solution of particle i, d means which dimensional the component is. P_{gd} is the dimensional component of particle swarm's optimal solution *gbest*. c_1, c_2 is learning factor values, usually is 2, r_1 and r_2 are random values in [0, 1], both of them are independent, ω is the inertia weight function.

In the standard PSO algorithm, the convergence of the particles are in the form of implementation of the track, and the velocity of the particles are always limited, so the search space in the search process of the particle is a limited area, it is impossible to cover the whole possible space. As a result, the standard PSO algorithm can not be guaranteed to converge to the global optimal solution with probability 1, which is the biggest drawback.

2.2 QPSO Algorithm

Many trials have demonstrated that the PSO algorithm can not converge to the global optimal solution, even locally optimal solution, and many scholars have adopted a lot of methods to improve the performance of the particle swarm optimization algorithm. From the point of view of quantum mechanics, Sun and others put forward a new model of the PSO algorithm after they learned the study on particle convergence behavior by Clerc and others in 2004. They thought the particles have quantum behavior and put forward the Quantum Particle Swarm Optimization

(QPSO) according to this new model. In quantum space, the state of aggregation of particles is completely different, it can search the entire feasible solution space, so the QPSO algorithm global search performance is far better than the standard PSO algorithm. In the quantum space, the speed and position of the particles can't be determined at the same time. We use the wave function to describe the state of the particle, and get the probability functions of the particle appeared in a position by solving the Schrodinger equation, and then get the location of the particle by Monte Carlo simulation. In QPSO algorithm, the particles only have the location information, and the location's update is determined by the following three equations:

$$mbest = \frac{1}{M}\sum_{i=1}^{M} pbest_i \tag{3}$$

$$p_{id} = \phi \cdot pbest_{id} + (1-\phi)\cdot gbest_d \tag{4}$$

$$x_{id} = p_{id} \pm \beta|mbest_d - x_{id}| \cdot \ln(1/\mu) \tag{5}$$

where in Eq. (3), $mbest$ is the center point of the optimal location in all individuals. ϕ is a random number in (0, 1), p_{id} is a random position between and, M is the size of the particle swarm. In Eq. (5), parameter β becomes the expansion-contraction factor, it used to control the speed of convergence of the algorithm, and it's the only parameter in QPSO algorithm that need to be controlled. The way to get the value of β is a liner function of the times of iterations by the algorithm, i.e.

$$\beta = (\beta_{max} - \beta_{min}) \cdot (iteration - iter)/iteration + \beta_{min} \tag{6}$$

where $iter$ is the current iteration number, $iteration$ is the total number of the iteration, β_{max} and β_{min} are two positive number, usually the value is 1.0 and 0.5 respectively.

From the above equation of motion, the obvious difference between QPSO and PSO is that the QPSO algorithm introduced a random exponential distribution of the particle position and put forward the concept of $mbest$ and p_{id}. Because of exponential distribution of the particle position, the particles in each iteration step of the search space is the entire real space, and can cover the entire feasible solution space to increase the ability to search the global optimal solution. In addition, the introduction of $mbest$ makes QPSO's convergent performance greatly improved. The QPSO algorithm processes can be described as:

Step 1: Initialize the particle swarm, and according to Eq. (3) calculate $mbest$'s value.

Step 2: Seeking the fitness value of each particle, get p_{id} with comparing, and compare each particle's p_{id}, then get p_{gd}.

Step 3: Update p_{gd}. Calculate particles in each dimension, according to Eq. (4), get a random point between p_{id} and p_{gd}.

Step 4: According to Eq. (5) to obtain a new position. Repeat Step 2 to Step 3 until the conditions are not met, the iterative process end.

2.3 Strengths and Weaknesses of the QPSO Algorithm

Because of a lot of features, the quantum particle swarm algorithm can overcome defects in convergence performance compared with the general particle swarm algorithm. First, the quantum system is a complex nonlinear system, in line with the state superposition principle. It has more states than a linear system. Second, a particle in the quantum system has a certain probability in the search space in every position because the particles are not a determined trajectory. Finally, by the traditional PSO algorithm, the limited search range made a fixed area for particles. In the QPSO algorithm, the particles appear to a certain probability in the entire feasible search space, even the position far away from the location of the center point. Such a position may also be a better adaptation value than *gbest* in current swarm. In the QPSO algorithm, at the post-process of the searching, the particles gather gradually, and make the search space smaller. Because of the stagnation in local extreme point, groups lose the ability to further expand the search space, and then make the whole group easy to fall into local optimal solution.

3 QPSO Based on Euclidean-distance Division of Labor Policy

We divided the particle swarm into two sub-group according to the Euclidean distance, and calculated the average distance between every particles and. We take the particles as a group which are less than the average, referred as $PNear$, the rest is divided into another group, denoted as $PFar$ in Fig. 1.

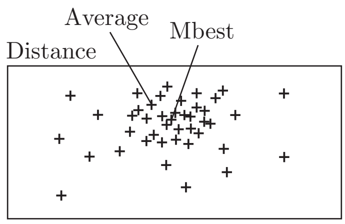

Fig. 1: Division of labor policy for SD-QPSO

Specific algorithm process can be described as follows:

Step 1: In the search space S, generated a group with N particles randomly, and initialize the relevant parameters;

Step 2: Constitute a collection of particles $PNear$ which selected the particles less than the average distance, the rest particles constitute the collection $PFar$, their intersection is empty.

Step 3: Select the appropriate parameters form these 2 sub-groups respectively. Calculate for the next position of the particles, and adjust them;

Step 4: Reconstruct the group equal the union of $PNear$ and $PFar$.

Step 5: If the termination condition is met, the algorithm terminates, otherwise go to Step 2.

4 Test Results of the Improved Algorithm

Select two representative test functions which not only have unimodal function, but also have multimodal function. And the test functions we are used are shown in Table 1. Set the iteration=2000, and the popsize is 20., We group f1, f2 and f4 into one set, f5, f6 and f7 into another group. f3, f8, f9 and f10 will be considered separately.

Table 1: SD-QPSO running results of f1, f2 and f4 compared with other PSOs

Fun	*Algorithms*	*Worst Value*	*Best Value*	*Mean Value*	*Variance*
$f1$	PSO	0.116759	0.0619157	0.068352	0.000343433
$f1$	CPSO	5.58448e-48	1.02756e-53	9.0664e-49	2.60984e-96
$f1$	QPSO	6.00531e-91	2.88245e-112	6.00533e-92	3.20968e-182
$f1$	TD-QPSO	5.5369e-86	1.27615e-94	1.84464e-84	2.79151e-167
$f1$	**SD-QPSO**	**1.29376e-96**	**3.64218e-108**	**2.01455e-97**	**1.53439e-193**
$f2$	PSO	2.24208	2.0619	2.13516	0.044387
$f2$	CPSO	1.84081	1.84081	1.84081	0
$f2$	QPSO	2.4524	1.84081	1.977905	0.0933546
$f2$	TD-QPSO	1.84459	1.84081	1.84411	0.0005125
$f2$	**SD-QPSO**	**2.4524**	**1.84081**	**1.977905**	**0.0933546**
$f4$	PSO	1.30333	0.404795	0.686461	0.0602081
$f4$	CPSO	5.04659e-47	1.49764e-52	5.40778e-48	2.23087e-94
$f4$	QPSO	1.18539e-92	8.66076e-109	1.18621e-93	1.25038e-185
$f4$	TD-QPSO	2.52126e-80	3.12587e-88	2.52126e-81	5.65751e-161
$f4$	**SD-QPSO**	**1.16493e-96**	**1.62628e-105**	**1.34053e-97**	**1.18106e-193**

From Table 1 we can see, SD-QPSO has the best performance compared with other PSOs in optimizing f1 and f4. We choose f1 as the typical example for unimodal function, and we get mean fitness value shown in Fig. 2.

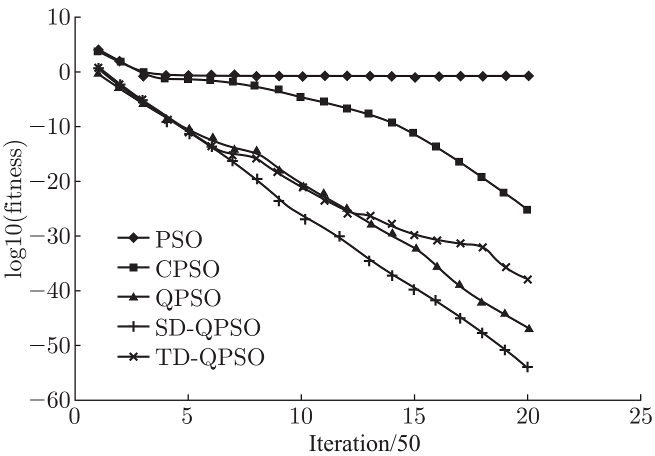

Fig. 2: Mean fitness curse of PSOs on optimizing f1

From Table 1 we can conclude that the smaller population size is, the dimension value is small, QPSO and SD-QPSO have better performances compared with PSO and CPSO, especially SD-QPSO.

From Table 2 we will conclude that all these five algorithms have unsatisfied results for f7, so we will choose f6 as the typical multimodal functions for PSOs to optimize as following Fig. 3.

Fig. 3 shows that for multimodal function, QPSO and SD-QPSO have almost the same performances. In order to test their behavior on different population size, we set population=50 and

Table 2: SD-QPSO running results of f5, f6 and f7 compared with other PSOs

Fun	*Algorithms*	*Worst Value*	*Best Value*	*Mean Value*	*Variance*
$f5$	PSO	30.4153	19.3628	22.7639	25.973
$f5$	CPSO	15.9193	6.96471	9.35261	9.30941
$f5$	QPSO	22.884	2.98488	12.1385	38.0889
$f5$	TD-QPSO	36.8133	2.98488	14.1284	87.1103
$f5$	**SD-QPSO**	**20.8618**	**3.97984**	**9.54837**	**25.6953**
$f6$	PSO	3.76064e-5	1.11782e-5	2.08635e-5	6.2657e-11
$f6$	CPSO	2.43539e-51	1.59699e-54	3.6708e-52	5.04868e-103
$f6$	QPSO	1.02166e-90	1.19873e-110	1.02166e-91	9.2897e-182
$f6$	TD-QPSO	3.89664e-81	1.27279e-99	3.89665e-82	1.35136e-162
$f6$	**SD-QPSO**	**1.04288e-934**	**1.98723e-112**	**2.08133e-94**	**1.68854e-187**
$f7$	PSO	2737.43	1698.16	2143.11	1.67205e+5
$f7$	CPSO	2829.21	1758.5	2246.09	1.47447e+5
$f7$	QPSO	1905.63	118.438	964.815	3.63131e+5
$f7$	TD-QPSO	2263.98	236.877	1056.22	3.46398e+5
$f7$	**SD-QPSO**	**2383.94**	**118.438**	**1386.84**	**5.71241e+5**

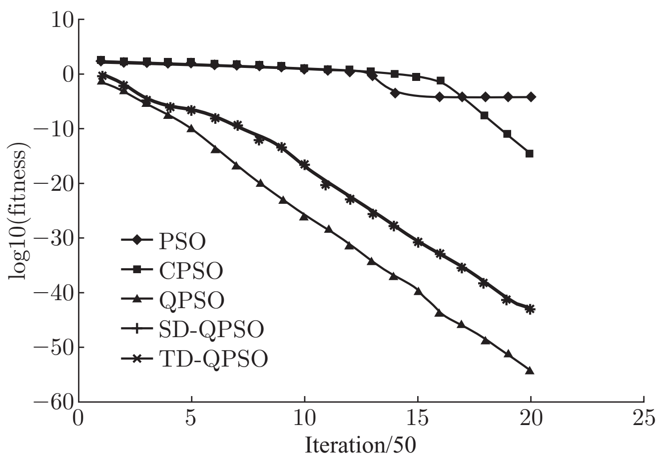

Fig. 3: Mean fitness curse of PSOs on optimizing f6

dimension=20.

In this experiment, CPSO and TD-QPSO runs more than 15 seconds. For TD-QPSO with the increase of population size and dimension, it cannot achieve optima at last. From the above data, We can obtain that when the particle swarm is small and the dimension is low, the SD-QPSO algorithm is much better than the PSO algorithm in terms of both unimodal and multimodal function on the accuracy. With the expansion of the swarm size and the increase of dimension, the accuracy of both algorithm is very similar when the QPSO algorithm does not fall into a local optimum. For the convergence rate, SD-QPSO algorithm is the fastest in the three algorithms when particle swarm's scale is small and the dimension is low, PSO is the slowest. With the expansion of the swarm size the speed of SD-QPSO algorithm will slow down, the QPSO algorithm will be the fastest when the QPSO algorithm does not fall into local optimal convergence. On

Table 3: SD-QPSO running results of f1, f2 and f4 compared with other PSOs

Fun	*Algorithms*	*Worst Value*	*Best Value*	*Mean Value*	*Variance*
$f1$	PSO	0.808214	0.520761	0.606369	0.00871192
$f1$	CPSO	2.82837e-26	2.3211e-29	3.50873e-27	6.76915e-53
$f1$	QPSO	2.19676e-78	9.56997e-81	4.53926e-79	7.39376e-157
$f1$	TD-QPSO	6.23857e-84	3.69666e-88	7.11973e-85	3.36386e-168
$f1$	**SD-QPSO**	**1.54748e-79**	**1.02038e-85**	**2.20502e-80**	**2.26491e-159**
$f2$	PSO	2.64125	2.41706	2.587	0.063094
$f2$	CPSO	1.84081	1.84081	1.84081	0
$f2$	QPSO	2.17124	1.84081	1.88977	0.0414308
$f2$	TD-QPSO	1.84081	1.84081	1.84411	0
$f2$	**SD-QPSO**	**2.17124**	**1.84081**	**2.17124**	**0.0414308**
$f4$	PSO	17.4101	7.19304	10.3561	10.9724
$f4$	CPSO	8.51387e-25	1.082e-28	2.93817e-26	2.05527e-51
$f4$	QPSO	1.37144e-782	9.82834e-85	1.41735e-79	1.66114e-157
$f4$	TD-QPSO	8.17189e-86	1.71294e-88	4.29714e-86	4.52116e-172
$f4$	**SD-QPSO**	**9.82456e-81**	**7.59344e-87**	**1.7926e-81**	**9.88465e-162**

the time complexity of the problem, the running time of the three algorithms is in the same order of magnitude. Therefore, considering the time of the algorithm complexity, performance, convergence speed and its search on the multimodal function, the SD-QPSO algorithm is the most desirable among the three algorithms.

5 Conclusion

Swarm intelligence algorithm as a new evolution of computing technology, it has become a focus of research. Particle Swarm Optimization is a very important field of swarm intelligence branch, which is originated in the simulation of bird flocking process, and later is discovered to be a good optimization tool. In this thesis, we use a more unified intelligent system to design and optimize, i.e. Quantum behaved Particle Swarm Optimization algorithm (QPSO). This study work around QPSO algorithm mainly summarized as follows: Firstly conduct in-depth study of the basic principles of particle swarm optimization, quantum particle swarm optimization, and provide a theoretical foundation for later learning applications. As the QPSO's diversity loss about later search, and the problem that it can not effectively escape from the local minima as convergence speed is slow.In this paper, we adopt Euclidean-distance division policy in the QPSO, the whole sub-groups are divided into two sub-groups which search individually, and ultimately achieve the purpose of the global search. In order to compare algorithm performance, We test three algorithms under the same operating conditions, and the SD-QPSO algorithm finally obtains a satisfactory and feasible result.

References

[1] M. Mikki, Ahmed A. Kishk, Quantum particle swarm optimization for electromagnetics [J], IEEE Transactions on Antennas and Propagation, 54 (10), October 2006, 2664-2775

[2] Yang Wang, Xiaoyue Feng, Yanxin Huang et al., A novel quantum swarm evolutionary algorithm and its applications [J], Neurocomputing, 2007, 23-25

[3] Jun Sun, Bin Feng, Wenbo Xu, Particle swarm optimization with particles having quantum behavior [C], Proc. of 2004 Congress on Evolutionary Computation, 2004, 325-331

[4] Jun Sun, Wenbo Xu, A global search strategy of quantum-behaved particle swarm optimization [C], Proc. of IEEE Conference on Cybemetics and Intelligent Systems, Washington, 2004, 111-116

[5] Y. Shi, R. C. Eberhart, A modified particle swarm [C], Proc. of IEEE International Conference on Evolutionary Computation, 1998

[6] Maurice Clerc, James Kennedy, The particle swarm - Explosion, stability, and convergence in a multidimensional complex space [J], IEEE Transactions on Evolutionary Computation, 6 (1), February 2002, 58-73