2011 Seventh International Conference on Computational Intelligence and Security

A New Improved Particle Swarm Optimization Algorithm

Yuhong Duan
School of Mathematics and Computer
Ningxia University
Yin Chuan 750021, China
Email:duanyuhong2005@163.com

Yuelin Gao
Institute of Information and System Science
North National University
Yin Chuan 750021, China
Email:gaoyuelin@263.net

Abstract— **To improve PSO, differential evolution (DEA) and ant colony strategy are involved into PSO algorithm, and new PSO(DAPSO) is presented. Handling the current optimal positions of particles with differential evolution, the detecting and exploitation ability of both PSO and DEA are utilized effectively, and some potential evolution directions are constructed for each particle in PSO, at the same time a strategy is presented to choose which one may be the local best for PSO evolution process just like pheromone table in ant colony algorithm. It is shown by tested with well-known benchmark functions that DAPSO algorithm is better than PSO algorithm with linearly decreasing weight and differential evolution algorithm.**

Keywords- Particle swarm optimization; differential evolution; ant colony algorithm; local searching

I. INTRODUCTION

Particle swarm optimization (PSO) was presented by Kennedy and Eberhart in 1995[1]. It is a random optimization algorithm based on swarm aptitude. Its thought comes from the research on the behavior for the bird swarm to catch food. Compared with genetic algorithm and ant algorithm, it is simple in construction and implemented easily and few in adjustable parameters. It is applied to solve the optimization problems to be nonlinear or not to be derivative or to be multipeak value. It is applied to deal with many practical problems too. Characteristic of particle swarm optimization algorithm is with high search efficiency in earlier state. However, as other intelligent algorithms, PSO also has insurmountable successfully to solving traveling salesman problem and has been paid extensive attention to. Characteristic of shortcomings, such as slower convergence rate in latter periods, even failing to local extremes. To overcome its shortcomings, there are many improved PSO, such as hybrid PSO and improved PSO based on subsidiary swarm and adaptive PSO[4-6]. We propose a new improved hybrid PSO algorithm where differential evolution and ant colony pheromone mechanism are combined with PSO (DAPSO). In this algorithm, the rate of progress and efficiency is improved, and some potential evolution directions are constructed for each particle. It is shown by tested with well-known benchmark functions that DAPSO algorithms is very efficacious.

This paper is organized as follows. In section 2, the based particle swarm optimization algorithm is introduced firstly. The specific of a new improved particle swarm optimization algorithm based on differential evolution algorithm (DEA) and ant colony are described in section 3. In section 4, simulation results are shown to demonstrate the effectiveness of the proposed algorithm. Finally some conclusions are included in section 5.

The work is supported by the Natural Science Foundation of China under Grant No.60962006 and the Natural Science Foundation of Ningxia University under Grant No.NDZR10-37.

II. BASED PARTICLE SWARM OPTIMIZATION ALGORITHM

Let n be the dimension of the search space, $x_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ be the current position of the i^{th} particle in swarm, $P_i = (p_{i1}, p_{i2}, \cdots, p_{in})$ be the best position of the i^{th} particle swarm so far, and $p_g = (p_{g^1}, p_{g^2}, \cdots, p_{g^n})$ be the best position of the whole swarm have ever visited. The rate of the velocity for the i^{th} particle is noted as $v_i = (v_{i1}, v_{i2}, \cdots v_{in})$. In based PSO model, the d^{th} dimension position and velocity of the particles are manipulated according to the equations:

$$v_{id}(t+1) = wv_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 \left(p_{gd}(t) - x_{id}(t)\right) \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t) \quad (2)$$

where the t denotes the t^{th} iteration, c_1 and c_2 are positive constants, called the cognitive and social parameter respectively, both are equal to two generally, r_1 and r_2 are random numbers uniformly distributed in the range $(0,1)$, w is inertia weight $d = 1, \cdots, n$. For

978-0-7695-4584-4/11 $26.00 © 2011 IEEE
DOI 10.1109/CIS.2011.18

inertia weight, Shi Y suggest that adopt the linearly decreasing (LEW) strategy[8], That is

$$w(t) = (w_{ini} - w_{end})(T_{\max} - t) / T_{\max} + w_{end} \quad (3)$$

Where t denotes the t^{th} iteration; $T_{\max}$ denotes the most iteration; w_{ini} denotes the original inertia weight; w_{end} denotes the inertia weight value when the algorithm process run the most iterations. Particle swarm optimization with linearly decreasing inertia weight is called LDW-PSO.

III. A NEW IMPROVE PARTICLE SWARM OPTIMIZATION ALGORITHM

A. To Improve PSO Based on Differential Evolution Algorithm(DEA)

Differential evolution algorithm(DEA) is presented by Storn and Price in 1995 [2] to solve continuous global optimization problem.Its idea is to obtain middle population by devotion of present population, and then obtain new population by fitness competing of hybrid individuals composed of farther individual and son individual. The process of DEA is as follows:

Step1:Initialize population size N, hybridization probability p_c, select randomly initialize population $X(0) = (X_1(0), X_2(0), \cdots, X_N(0))$,

And let $X_{best}(0)$ denotes the best individual of $X(0)$, $t := 0$ .Where $X_i(0)$ is n dimension vector.

Step 2:Operate individual $X_i(t)$ of $X(t)$ as follows:

1)Mutating

Let

$$V_i(t) = X_i(t) + \lambda[X_{best}(t) - X_i(t)] + \beta[X_{r2}(t) - X_{r1}(t)] \triangleq (\overline{V}_1, \overline{V}_2, \cdots, \overline{V}_n)^T \quad (4)$$

Where $r1$ and $r2$ are two different random integers of $[1, N]$, λ and β are selection parameter.

2)Hybridizing

Select randomly integer m in $[1, N]$ and integer L by probability $p\{L = k\} = p_c^k$, let $U_i = (\overline{u}_1, \overline{u}_2, \cdots, \overline{u}_n)^T$,

For $j = 1,2,\cdots,N$, where

$$\overline{u}_j = \begin{cases} \overline{V}_j, & if\ j = \langle m \rangle_n, \langle m+1 \rangle_n, \cdots, \langle m+L-1 \rangle_n, \\ [X_i(t)]_j & others, \end{cases} \quad (5)$$

Where $\langle \cdot \rangle_n$ denotes modulus function of modulus n.

3)Selecting

Let

$$X_i(t+1)) = \begin{cases} U_i(t) & \text{if} \quad J(U_i(t)) > J(X_i(t)) \\ X_i(t) & \text{if} \quad J(U_i(t)) \le J(X_i(t)) \end{cases} \quad (6)$$

Where $J(X)$ is fitness function of individual X.

Step3:Let $X(t+1) = (X_1(t+1), X_2(t+1), \cdots, X_N(t+1))$ denotes the new population produced via step 2, and let $X_{best}(t+1)$ denotes the best individual of $X(t+1)$. If it is satisfied with ending condition, then output result, otherwise, let $t := t+1$ go to step 2.

Each particle regard the local optimal solution p_i as evolution direction in PSO. To improve PSO base on differential algorithm, we let all local optimal solution p_i be initialize population, and make them do differential evolution operating which include mutating and hybridizing and selecting, then obtain the better local optimal solution, and thereby make evolution direction better, quicken convergence velocity.

B. To Improve PSO Based on Ant Colony Algorithm

Ant colony algorithm presented by Dorigo in 1996[2] is a kind of bionics algorithm that imitates forage process of ant. In ant colony algorithm, each ant can one select section with greater pheromone trail density according to probability in many sections to obtain optimization solution. But each particle in PSO only retain information of the present optimal solution, and it may fall into local optimization. We lead into pheromone trail model in PSO aroused by ant colony algorithm. In new improved PSO, produce k points in neighborhood of the present optimal solution p_g, and we record these k points by sequence pp_g, that is $pp_g = \{pp_g(1), pp_g(2), \cdots, pp_g(k)\}$, where $pp_g(j) = p_g + rd_j$, r is step length, d_j is direction. Let $pp_g(k+1)$ denotes the present optimal solution p_g

We structure probability of the present optimal solution to select point in sequence pp_g as follows:

$$p_g(j) = \begin{cases} \max\limits_{1 \le j \le k+1} \{Finess(pp_g(j))\} & q \ge q0 \\ \dfrac{Finess(pp_g(j))}{\sum_{l=1}^{k+1} Finess(pp_g(l))} & q < q0 \end{cases} \quad (7)$$

Where $q0$ $(0 \le q0 \le 1)$ is a given parameter, function $Finess(x)$ is fitness of point x. We can find that the possibility of the point selected with greater fitness value is greater according to selection probability. So sequence pp_g obtained by local searching make evolution direction of particle have diversified selection, therefore improve ability to avoid local optimization.

To retain quick convergence velocity of based PSO, we design a optimization strategy based on aggregation degree of particle. Document[7] presented that positions of particles are identical is equal in value to

fitness are same, and aggregation degree is defined by formula (8):

$$\sigma^2 = \sum_{i=1}^{m} \left(\frac{Finess(i) - F_{ave}}{F}\right)^2 \quad (8)$$

Where F which main action is to limit σ^2 is an accommodate factor, and is defined by formula (9):

$$F = \begin{cases} \max_{1 \le i \le m} \{Fitness(i) - F_{ave}\} & , \max_{1 \le i \le m} \{Fitness(i) - F_{ave}\} > 1 \\ 1 & , \max_{1 \le i \le m} \{Fitness(i) - F_{ave}\} \le 1 \end{cases} \quad (9)$$

When the particles distribute dispersedly, the searching ability of swarm particles is powerful, the value of σ^2 is greater, and selecting point with the best fitness in sequence pp_g can ensure convergence velocity. When value of σ^2 change smaller indicate that particle swarm begin with convergence, enlarge selecting range of sequence pp_g step by step to increase diversify of particle. We design piecewise function to define range of k as follows:

$$k = \begin{cases} 0 & , \ \sigma^2 > \sigma_1 \\ \cdots & \cdots \\ K & , \ \sigma^2 \le \sigma_2 \end{cases} \quad (10)$$

Experiments show that too many piecewise are not obvious improving to function, so we certain that $\sigma_1 = \sigma_2 \in (0,1)$ can obtain good affection.

C. Describing New Improved Particle Swarm Optimization Algorithm

Based on discussing upwards, we now give a new improved particle swarm optimization algorithm based on differential evolution algorithm (DEA) and ant colony algorithm (DAPSO). Process of DAPSO is as follows:

Step1: Initialize a population of particles with random positions x_i and velocities v_i on n dimensions in the problem space.

Step2: Initialize the solution of each particle P_i, for each particle, compute the desired optimization fitness function in n variables so that obtain global best solution P_g

Step3: If the algorithm satisfies the convergence rule or reach the most iteration, then loop to step8, otherwise, go to step 4.

Step4: Change the velocity and position of the particle according to equations (1) and (2), for each particle, compute the desired optimization fitness function in n variables, change P_i and p_g.

Step5: Let all local optimal solution p_i be initialized population, and make them do differential evolution according to equations (4) -(6), replace p_i and P_g with population and the best individual obtained by differential evolution operating.

Step6: Compute aggregation degree σ^2 according to equations(8)and(9).If $\sigma^2 > \sigma_1$, then loop to step7, otherwise, search locally in neighborhood of the present optimal solution p_g of swarm, compute selection probability according to equation (7), select suitable point from sequence pp_g.

Step7: Set iteration t to $t = t + 1$, and loop to step 3.

Step8: Input the global optimal position and its fitness value.

IV. NUMERICAL TEST AND ANALYSIS

We test capability of LDW-PSO and DEA and DAPSO by classical functions (see table 1). In experiment , we assign the particle number is 30, $c_1 = c_2 = 1.7,$ $w = (0.9 - 0.4)(T_{\max} - t)/T_{\max} + 0.4,$ and $\lambda = \beta = 0.8, p_c = 0.3.$ Three algorithm run 50 times for each function, we record the best fitness value and evaluate fitness value average , the numerical results are listed in table 2-6 .

We can find that from table 2-6 that whether fitness(function)value average or best value of DAPSO is better than DEA and LDW-PSO, and computation precision have clearly improved for 5 functions. DAPSO can obtain the optimal value1, but DEA and LDW-PSO can not for f_1 and f_5 .We draw five evolution diagrams of curve to make clear capability of three algorithm. For data differ too big ,abscissas axis fetch iteration and ordinate axis fetch common logarithmic of fitness(function)value average .We can find from Figure1-5 DAPSO can obtain better optimal value than DEA and LDW-PSO in same iteration, and DAPSO descent quickly than DEA and LDW-PSO, and DAPSO have more quick convergence velocity.

V. CONCLUSION

DEA is similar to genetic algorithm, we lead into differential evolution strategy in PSO, operate local optimal positions of all particles by mutating and hybridizing and selecting, therefore obtain the better local optimal positions, thereby optimize evolution direction of particle, and quicken convergence velocity. Moreover, we leap into ant colony algorithm model by analyzing shortcoming of PSO to solve premature problem. Numerical experiment shows that improved PSO based on DEA and ant colony is efficacious and feasible.

REFERENCES

[1] Eberhart R C ,Shi Y H,“ Particle Swarm optimization: development ,applications and resource, Proceedings of the IEEE Congress on Evolutionary Computation. Piscataway,USA: IEEE Service Center,2001,81-86.

[2] Dorigo M, Maniezzo V, Colorni A, Ant System: Optimization by a Colony of Cooperating agents”, IEEE Transaction on System Man *and Cybernetics-Parts B*,1996,26(1):29-41.

[3] Xiaofeng Xie, Wenjun Zhang, Zhilian Yang. Overview of particle swarm optimization [M].Control and decision making.2003, February,V0l.18:129-134.

[4] Robinson J, Sinton S, Rahmat-Samii Y. Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna[C], IEEE Antennas and Propagation Society Intern-ational Symposium and URSI National Radio Science Meeting, San Antonio, TX 2002.

[5] Li Aihuo. A Cooperative Particle Swarm Optimization Algorithm, Transaction of Fudan University,2004,43(5):923-925

[6] Zhang Xuanping, Yu Du, Guoqiang Qin. An adaptive Particle Swarm Algorithm with Dynamically Changing Inertia Weight [A]. Transaction of Xi’an Jiaotong University. 2005,October, V0l.39:1039-1042.

[7] Lu Z S, Hou Z R. Particle Swarm Optimization with Adaptive Mutation [J]. Acta Electronica Sinica, 2004, 32(3):416-420.

[8] Zongben Xu.Intelligence Computation-analog evolution com-Putation.Higher education press,2005:116-120.

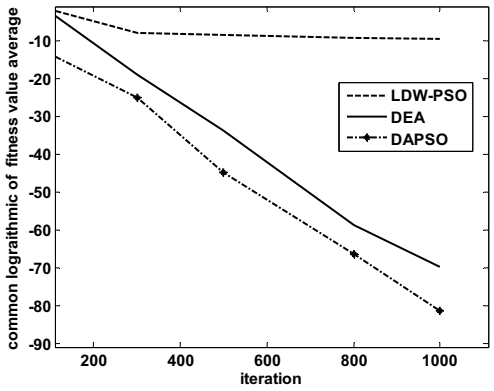

Figure1 envolution curve of fitness value average logarithmic of f_1

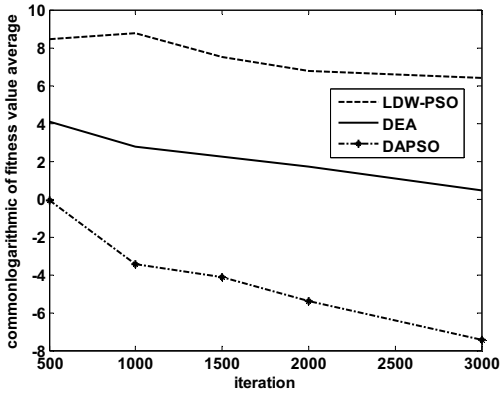

Figure2 envolution curve of fitness value average logarithmic of f_2

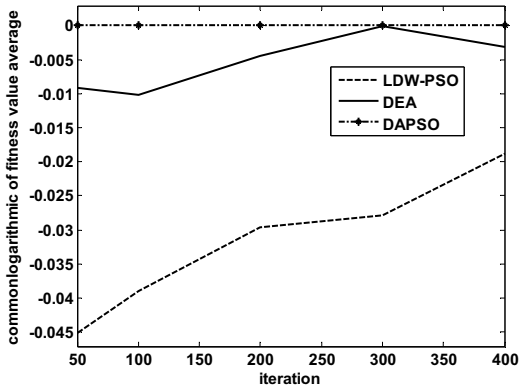

Figure 3 envolution curve of fitness value average logarithmic of f_3

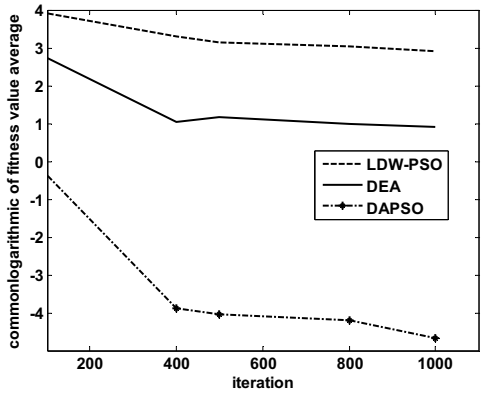

Figure 4 envolution curve of fitness value average logarithmic of f_4

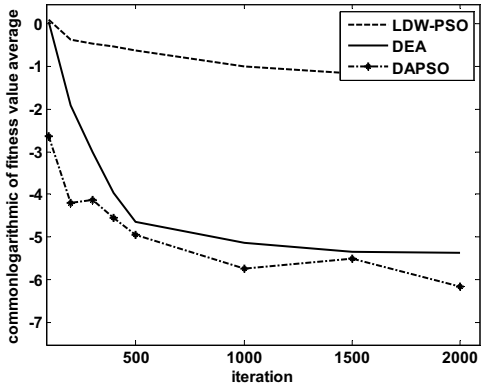

Figure 5 envolution curve of fitness value average logarithmic of f_5

Table1 five normal nonlinear testing functions

Sphere Model： $f_1(x)=\sum_{i=1}^{10}x_i^2, x_i\in[-10,10]$， global optimal function value is 0
Generalized Rosenbrock's function: $f_2(x)=\sum_{i=1}^{10}[100(x_{i+1}-x_i)^2+(x_i-1)^2], x_i\in[-100,100]$， global optimal function value is 0
J.D.Schaffer'function: $f_3(x)=0.5-\frac{\sin^2(x_1^2+x_2^2)^{1/2}-0.5}{(1+0.001\times(x_1^2+x_2^2)^2}$， $x_1,x_2\in[-10,10]$， global optimal function value is 1
Generalized Rastrigin's Function: $f_4(x)=\sum_{i=1}^{10}(x_i^2-10\cos(2\pi x_i)+10), x_i\in[-10,10]$, global optimal function value is 0
Generalized Griewank Function: $f_5(x)=\frac{1}{4000}\sum_{i=1}^{10}x_i^2-\prod_{i=1}^{10}\cos\left\lvert\frac{x_i}{\sqrt{i}}\right\rvert+1$ $x_i\in[-600,600]$, global optimal function value is 0

Table 2 fitness (function)value comparing of three algorithms for f_1

iterations	LDW-PSO		DEA		DAPSO	
	average value	best value	average value	best value	average value	best value
100	0.1550	0.0691	0.0747	0.0312	1.1607×10^{-6}	5.3566×10^{-7}
300	3.4231×10^{-4}	6.2344×10^{-5}	5.4331×10^{-9}	8.6711×10^{-10}	2.1381×10^{-14}	2.3216×10^{-17}
500	2.1011×10^{-4}	1.3111×10^{-5}	2.3312×10^{-15}	8.1803×10^{-17}	3.4059×10^{-20}	1.8376×10^{-26}
800	9.5672×10^{-5}	2.4723×10^{-6}	2.8923×10^{-26}	6.1122×10^{-29}	1.2857×10^{-29}	1.2782×10^{-35}
1000	6.1883×10^{-5}	1.4742×10^{-7}	4.0656×10^{-31}	2.8117×10^{-35}	4.4127×10^{-36}	1.9073×10^{-43}

Table3 fitness (function)value comparing of three algorithms for f_2

iterations	LDW-PSO		DEA		DAPSO	
	average value	best value	average value	best value	average value	best value
500	4693.0237	110.1023	57.7145	8.0156	0.9147	8.2158×10^{-6}
1000	6517.7312	38.6347	16.0179	0.4614	0.0318	4.5148×10^{-15}
1500	1804.6782	37.0123	9.2611	0.1307	1.5709×10^{-2}	8.9619×10^{-21}
2000	853.2203	25.2027	5.5691	0.1130	4.4421×10^{-3}	5.0505×10^{-28}
3000	611.5122	15.7115	1.5430	0.0208	5.7913×10^{-5}	2.2766×10^{-31}

Table4 fitness (function)value comparing of three algorithms for f_3

iterations	LDW-PSO		DEA		DAPSO	
	average value	rate of be1	average value	rate of be1	average value	rate of be1
50	0.9559	0%	0.9909	0%	1.0000	0%
100	0.9618	0%	0.9899	0%	1	100%
200	0.9708	0%	0.9956	0%	1	100%
300	0.9726	0%	0.9999	0%	1	100%
400	0.9814	0%	0.9969	0%	1	100%

Table5 fitness (function)value comparing of three algorithms for f_4

iterations	LDW-PSO		DEA		DAPSO	
	average value	best value	average value	best value	average value	best value
100	50.7032	26.5671	15.5622	4.0318	0.7214	0.1360
400	27.7011	16.98221	2.8942	2.0123	0.0211	3.8701×10^{-7}
500	23.9991	12.5699	3.2794	2.0111	1.8312×10^{-2}	7.366×10^{-12}
800	21.1278	9.8633	2.7511	1.0081	1.5356×10^{-2}	3.5527×10^{-16}
1000	18.8644	8.2341	2.5412	1.9811	9.5122×10^{-3}	7.1034×10^{-17}

Table6 fitness (function)value comparing of three algorithms for f_5

terations	LDW-PSO		DEA		DAPSO	
	average value	rate of be1	average value	rate of be1	average value	rate of be1
200	0.6883	0%	0.1485	0%	0.0151	0%
300	0.6311	0%	0.0487	0%	0.0161	7.25%
400	0.5811	0%	0.0190	0%	0.0105	20.1%
500	0.5351	0%	0.0095	0%	0.0071	39.5%
1000	0.3723	0%	0.0059	0%	0.0032	62%
1500	0.3123	0%	0.0048	0%	0.0040	56.6%
2000	0.3212	0%	0.0046	16.7%	0.0021	68.8%