Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 24 (2013) 54 – 65

Procedia Computer Science

17th Asia Pacific Symposium on Intelligent and Evolutionary Systems, IES2013

Dynamic Diversity Enhancement in Particle Swarm Optimization (DDEPSO) algorithm for preventing from premature convergence

Omid Mohamad Nezami[a*], Anvar Bahrampour[b], Paria Jamshidlou[c]

[a]*Department of Computer, Bijar Branch, Islamic Azad University, Bijar, Iran*
[b]*Department of Computer, Sanandaj Branch, Islamic Azad University, Sanandaj, Iran*
[c]*Sharif University of Technology, Tehran, Iran*

Abstract

The problem of early convergence in the Particle Swarm Optimization (PSO) algorithm often causes the search process to be trapped in a local optimum. This problem often occurs when the diversity of the swarm decreases and the swarm cannot escape from a local optimal. In this paper, a novel dynamic diversity enhancement particle swarm optimization (DDEPSO) algorithm is introduced. In this variant of PSO, we periodically replace some of the swarm's particles by artificial ones, which are generated based on the history of the search process, in order to enhance the diversity of the swarm and promote the exploration ability of the algorithm. Afterwards, we update the velocity of the artificial particles in corresponding generating period by a new velocity equation with the minimum inertia weight in order to enhance the exploitation potentiality of the swarm. The performance of this approach has been tested on the set of twelve standard unimodal and multimodal (Rotated or unrotated) benchmark problems and the results have been compared with our previous work as well as four other variants of the PSO algorithm. The numerical results demonstrate that the proposed algorithm outperforms others in most of the test cases taken in this study.

Keywords: Particle Swarm Optimization (PSO) Algorithm; Population Diversity; Premature Convergence.

[*]Corresponding author. Tel.: +98-918-8723586; fax: +98-872-4237541.
*E-mail address:*Nezami@iaubijar.ac.ir.

doi:10.1016/j.procs.2013.10.027

1. Introduction

Particle Swarm Optimization (PSO), which is one of the swarm intelligence computation techniques, was invented by Russ Eberhart and James Kennedy in 1995 [2-3]. It is an evolutionary algorithm that inspired from social behaviors in birds flocking and fish schooling to guide particles to search for global optimum in optimization problems. Each particle has an associated point in the search space which represents a solution, and flies through the search space to find globally optimal solution with an associated velocity which dynamically adjusted through the search process. The current velocity of a particle is typically a linear combination of three elements: (1) its previous velocity (2) the distance from the best point found by the particle and (3) the distance form the best point found by the swarm. The PSO algorithm iteratively modifies the point and the velocity of each particle as it looks for the optimal solution. The rate of convergence of particles in PSO is good through the fast information flowing among particles, so its diversity decreases very quickly in the successive iterations and lead to a suboptimal solution. This situation was said that an evolutionary process was trapped in a local optimal or premature convergence of the process.

The standard PSO algorithm can easily get trapped in the local optimal when solving complex multimodal problems. These weaknesses have restricted wider applications of the PSO [5- 6- 7]. Some reasons cause to this problem, one of that is decreasing diversity of population. A number of variants of PSO algorithm have been proposed to overcome the problem of diversity loss. One of the common methods to increase the diversity is mutation. Mutation causes an improvement in exploration abilities, which can be applied to different elements of a particle swarm. The effect of mutation depends on which elements of the swarm are mutated [8]. Velocity vector mutation is equivalent to particle's position vector mutation, under the condition that the same mutation operator is considered.

In [8] a negative feedback mechanism into particle swarm optimization has proposed and developed an adaptive PSO. This mechanism takes advantage of the swarm-diversity to control the tuning of the inertia weight (PSO-DCIW), which in turn can adjust the swarm-diversity adaptively and contribute to a successful global search. Some other methods exist that using diversity measuring and mutation in the particle's position, to promote the performance of the algorithm include Gaussian Mutation [8-10-11-12-13], Cauchy [13-14], and Chaos Mutation [15-16-17].

There are other different ways of introducing diversity and controlling the degree of diversity; Riget and Vesterstorm proposed an algorithm named ARPSO. In ARPSO if diversity is above the predefined threshold d_{high} then particles attract each other, and if it is below d_{low} then the particle repel each other until they meet the required high diversity d_{high} [18]; repulsion to keep particles away from the optimum proposed by Parsopoulos and Vrahatis [19]; LoZvbjerg and Krink made dispersion between particles that are too close to one another [20]; and Blackwell and Bentley have reduced the attraction of the swarm center to prevent the particles clustering too tightly in one region of the search space in order to escape from local optimal [21]. J. J. Liang and P. N. Suganthan proposed a dynamic multi-swarm particle swarm optimizer (DMS-PSO)[22]. In this method whole population is divided into many small swarms, these swarms are regrouped frequently by using various regrouping schedules and information is exchanged among the swarms.

In this paper we propose a mechanism to enhance the diversity of the swarm based on particle replacement periodically, which in turn try to detect suitable positions (new particles) of the search space (points with fairly good fitness, and good distance from current distribution of the swarm particles) to replacing some of existing particles by new ones, hoped to increase diversity level of the swarm and escape from local optimal by detecting better area of the search space.

The rest of the paper is organized as follows: Section 2 describes the basic Particle Swarm Optimization algorithm. In section 3, we have a definition of Particle swarm optimization based on diversified artificial particles. Dynamic diversity enhancement mechanism described in section 4. Experimental results are discussed in section 5. Finally, this paper concludes in section 6.

2. Particle swarm optimization

Particle Swarm Optimization or PSO is a population based evolutionary algorithm developed by Eberhart and Kennedy in 1995, inspired from social behavior of natural species such as bird flocking or fish schooling. Each particle represents a point of multidimensional search space or a solution of the multidimensional problem that flies

through a search space, denoted by $X_i = (x_{i1}, x_{i2},...,x_{id} ,..., x_{iD})$ for a D-dimensional problem, and looking for a globally optimal solution. So each member of the swarm adjusts its position in the search space from time to time according to the flying experience of its own and of its neighbors (or colleagues). In PSO, best position that the particle i meets in previous generations is denoted by $P_i = (p_{i1}, p_{i2},..., p_{id} ,..., p_{iD})$, and the best one among all the particles in the swarm is represented as $P_g = (p_{g1}, p_{g2},..., p_{gd} ,..., p_{gD})$. The two basic equations of PSO which implement flying of particles are the velocity equation and the position one as the Equation (1).

$$V_{id} = \omega * V_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id})$$
$$X_{id} = X_{id} + V_{id} \quad (1)$$

Where Vi in the first equation is the velocity of Particle i that represented as Vi = (vi1, vi2,..., vid ,..., viD). The first part of the Equation (1) is the inertia of the previous velocity, ω is predefined by the user, and the second part represents the personal thinking of the particle. The third part is the social component. In this equation c1 and c2 are acceleration constants. They represent the weighting of the stochastic acceleration terms that pull each particle toward personal best and global best positions. The constants r1, r2 are the uniformly generated random numbers in the range of [0, 1]. Velocity clamp (A particle's velocity on each dimension is clamped to a maximum magnitudeV_{max}) was firstly used to adjust the ability between exploration and exploitation [4, 23]. However, velocity clamp only makes algorithm less like to diverge, it cannot help algorithm "jump out" a local optimum or refine the candidate solution. In each iteration a new velocity value for each particle is calculated based on its current velocity, the distance from the local best position, and the global best position. The new velocity value is then used to calculate the next position of the particle in the search space according to Equation (1). This process is then iterated a number of times or until a certain condition occurs.

3. Particle swarm optimization based on diversified artificial particles

The basic idea of our previous work [1] was to measure the diversity level [4] of the swarm during the evolutionary process and once the diversity of the population drops down to the predefined threshold level *d*, then the system of generating diversified artificial particle (DAP system) activated, and start to replacement some of the particles of the swarm that have relatively bad fitness with the particles generated by the DAP system which have high diversity and fairly good fitness. Therefore, the diversity level of the swarm would be increased up to certain degree; this process should be repeated once the diversity level drops down the *d* value again. Since in the PSO algorithm the speed of convergence is very high, the diversity control mechanism and generating DAP particles should be repeated very often. Therefore, we introduced a new parameter *T*, to define the duration that the DAP system should be passive after each replacement process. The proposed method for generating DAP particles, which illustrated in [1] and replacement of the swarm's particles was implemented independently from the problem characteristics to improve the global convergence behavior of PSO algorithms.

In this research we use similar process, and since the artificial particles generated in new regions of the search space at positions with good fitness and have far distance from current distribution of the converged swarm, we try to search very carefully in new regions by new artificial particles in corresponding generating period, In order to search more thoroughly and precisely.

4. Dynamic diversity enhancement mechanism

As mentioned above the basic idea of this research is enhance the diversity of the swarm dynamically by introducing new artificial particles into swarm and search carefully new regions of the search space, which new artificial particles located in. In order to generate artificial particles we a system of generating diversified artificial particle (DAP system), which uses the history of the search process by making an external archive that consists the best particles of previous generations. The following section illustrates the new variant of PSO, and section 4.2 shows how we generate diversified artificial particles from external archive.

4.1. DDEPSO algorithm

Fig. 1 shows the process of Dynamic Diversity Enhancement Particle Swarm Optimization algorithm (DDEPSO); the steps of this process are the same as the steps of the standard PSO except steps 4, 5, and in step 6 we use two different velocity equations. In step 4 we update External Archive in order to use as historical information in generating diversified artificial particles. In step 5 we have a diversified artificial particles mechanism (DAP), and should be executed if duration T from last Replacement time was elapsed. DAP process will enhance the swarm diversity by replacement some of converged bad fitness particles with the new diversified artificial particles. Through this step strongly prevent premature convergence and extend the searching space for locating a better solution. Although, the fitness of the DAPs may not as good as those within the population before DAP process, but the diversity of new artificial particles is pretty high, and the exploration of the evolutionary process can restart again.

In step 6 we use two different velocity equations, the equation (1) and the following equation, for update velocity of the swarm's particle. The following velocity equation is used just for artificial particles which generated in the latest replacement process. Since we want to search more carefully in new regions by new artificial particles, we use the minimum inertia weight. In addition to the above, since new artificial particles locate in regions more likely with far distance from converged particles, then probably we have a big component of velocity to the global best particle in velocity equation, so we use a random coefficient r_0 in new velocity equation.

$$V_{id} = r_0(\omega_{\min} * V_{id} + c_1 r_1(p_{id} - x_{id}) + c_2 r_2(p_{gd} - x_{id})) \quad (2)$$
$$X_{id} = X_{id} + V_{id}$$

4.2. An artificial particle generation

As mentioned in the previous sections, high diversity and variety of particles could influence the convergence and the ability to escape from local optimal solution. In this section we describe a diversified artificial particle generating mechanism in order to generate artificial particles with high quality and good diversity. In the previous section, we discussed development of an external archive to collect particles with high diversity and fairly good fitness in previous generations of PSO process. In this research we established an external archive with 100 particles, and initialized it with random particles. Firstly, we gather particles with best fitness in the first generations (about 100 generations in this research) of the PSO process and replace particles in the external archive which have bad fitness. Then we should establish a replacement policy in order to gather effective particles in external archive and avoid the convergence of external archive particles. In this study after first 100 generations, we only do replacement when best particle found by PSO algorithm (best of all generation up to now) changed, and remove one of the particles with low diversity. For detect low diversity particles to remove from external archive, we use a Euclidean distance[1], and measure distance of each particle from the mean of external archive particles. One of the particles with less distance should be replaced by new particle.

```
Step 1: Initialize PSO parameters, Period T, and Replacement
        Rate R, Generate initial Swarm, External Archive, and
        velocities of all particles

Step 2: Evaluate the Fitness of all particles in the Swarm

Step 3: Update particles personal best position and global best
        position (i.e. Pi and Pg).

Step 4: Update External Archive

Step 5: If (last Replacement took place more than T generations
        ago)
            Generate Diversified Artificial Particles (DAPs)
            Select artificial particles with high diversity and
              fairly good fitness
            Replace R% of the swarm's particles (with bad
              fitness) by selected DAPs
            Generate velocities, and set personal best position to
              the current position for replaced particles
End if

Step 6: For each particles P of the swarm
        If P is a particle of latest DAPs
            Update particles velocity using equations (2)
        Else
            Update particles velocity using equations (1)
        Update position vector of particle P

Step 7: If (Stopping criteria is reached) then go to step 8
Else go to step 2

Step 8: Print the global best particle and the corresponding
        fitness function value
```

Fig. 1. Steps of DDEPSO Algorithm

For generating artificial particle from information of the external archive, we use a roulette wheel concept to create a new copy of external archive as Roulette, and We add two new rows to the Roulette for mutation purpose, first one is a vector of xMax (maximum value in each dimension), and the second is a vector of xMin(minimum value in each dimension) at the end of the matrix. Therefore, we use this mechanism to determine value of each dimension in an artificial particle by randomly selection one of the values stored in the same dimension of the Roulette. In other words, in order to generate each artificial particle, for each dimension value we select one row of the Roulette randomly, and use value stored in the same dimension of selected particle, then we insert this new particle into a mating pool. Therefore, we have particles which are in the external archive and new generated particles, all in one mating pool; then we apply operators such as selection(Roulette wheel method), crossover and mutation to promote quality and diversity of generated particles. Finally, we select a numbers of particles (45% of the swarm in this research) that have good diversity (distance from mean of the swarm) and fairly good fitness, and replace with the same numbers of existing particles in the swarm with relatively bad fitness particles. This process will increase diversity of the swarm notably and help to escape from local optimal trap. Fig. 2 illustrates the DAP system process. Fig. 3 shows how diversity level of DDEPSO increases in comparison with GPSO(Global star structure PSO) and causes to escape from local optimal for function 10 in Table 1. This experiment is on 3000 iteration by 20 particles of 30 dimensions, and duration 100 for DAP system.

```
Roulette= Roulettewheel(Ex); //External Archive
Roulette (Roulette_Size+1,:) =xMin;
Roulette (Roulette_Size+2,:)= xMax;
 //xMin & xMax Adding for mutation purposes.
For each artificial particle p do
 For each dimension d do
    Par =ceil (rand*( Roulette_Size+1));
    DAP (p, d) =Roulette (par, d);
 End
End
Crossover selected DAP and EX particles and does Mutation
```

Fig. 2. DAP system process

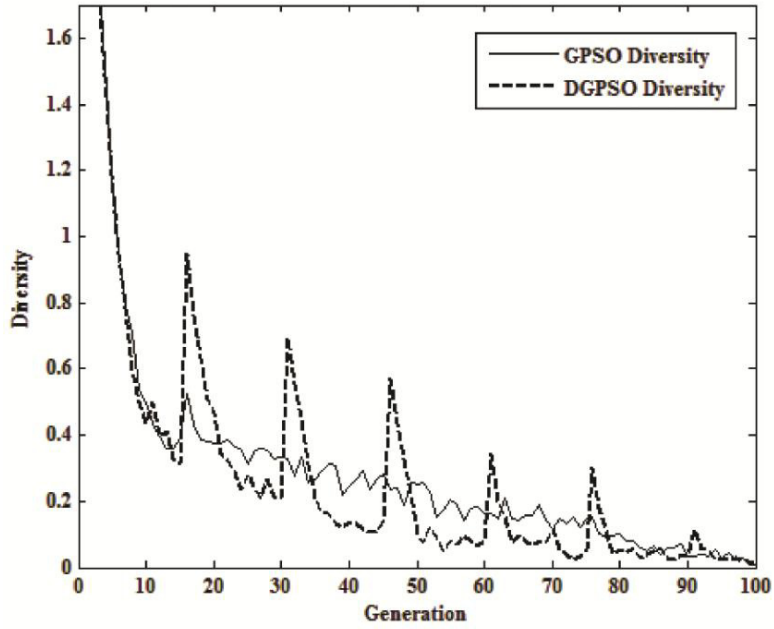

Fig. 3. Diversity level of GPSO and DDEPSO for function f_{10}

5. Experimental settings and numerical results

Here For comparison of PSO and DDEPSO algorithms, we have used a collection of 12 standard benchmark problems. Mathematical models of the problems along with the true optimum value are given in Table 1. In this benchmark problem set, we have a unimodal, and multimodal (rotated and unrotated) functions [28]. The entire set of test problems taken for this study is scalable i.e. the problems can be tested for any number of variables. However, for the present study we have tested the problems for dimensions 30 and 50.

In order to make a fair comparison of proposed DDEPSO with some of other variants of PSO algorithm, we implement standard PSO algorithm in both global star structure and local ring structure named GPSO and LPSO, respectively. In addition to these comparisons we also implement PSO-DCIW and DMS-PSO algorithms, which are proposed in [9- 22], and PSO-DAP(our previous work[1]) and compared with DDEPSO. We use the same conditions and same initial populations for candid comparison between all algorithms. The population size was taken as 20 while we have 30 variables (dimensions) for all the test problems, and 50 when problems should be tested with 50 variables. A linearly decreasing inertia weight is used which starts at 0.9 and ends at 0.4, with the user defined parameters c_1=2.0 and c_2=2.0. For each algorithm, the maximum number of iterations is set as 3000 iterations in the case of 30 dimensions, and 10000 for dimension 50. A new parameter *T* for DDEPSO algorithms is set as 30 and 50 in cases of 30 and 50 dimensions respectively, with the external archive of size 100, dispersion rate *R* is set as 45%. In DMS-PSO we use group size 3 and regroup period 5. A total of 20 runs for each experimental setting were conducted, and the results are given in Table 2 and Table 3, in terms of mean of best fitness, standard deviation, and P-value of algorithms. Fig. 4 through 6 show performance curves DDEPSO in comparison with the basic PSO in both global star and local ring structures, DMS-PSO, PSO-DCIW, PSO-DAP by mean fitness of best particles history found by the swarm in all runs. The numerical results show that the proposed algorithm outperforms the other variants of PSO algorithms in most of the test cases taken in this study.

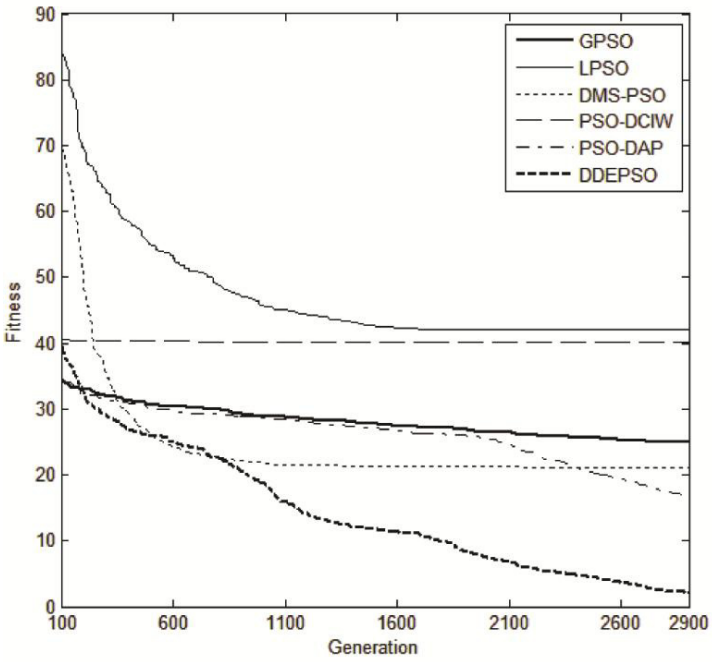

Fig. 4. Performance curves of GPSO, LPSO, DMS-PSO, PSO-DCIW, PSO-DAP, and DDEPSO for function f_8

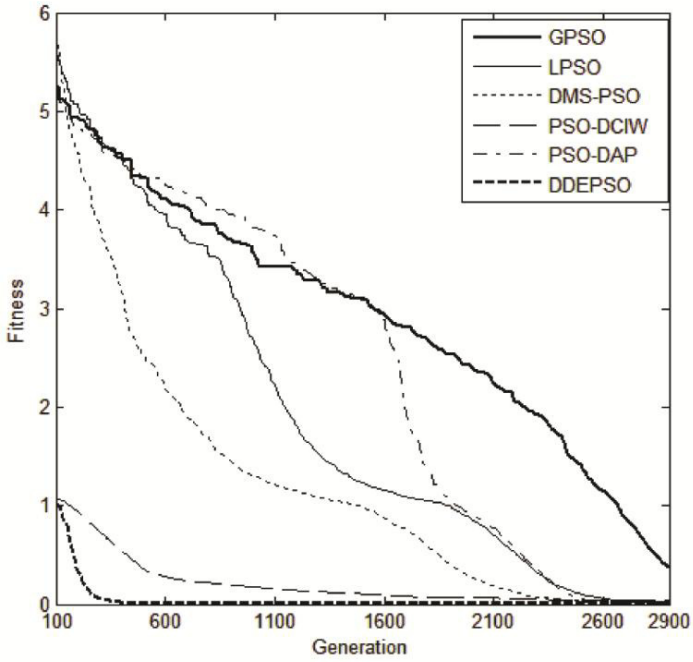

Fig. 5. Performance curves of GPSO, LPSO, DMS-PSO, PSO-DCIW, PSO-DAP, and DDEPSO for function f_{10}

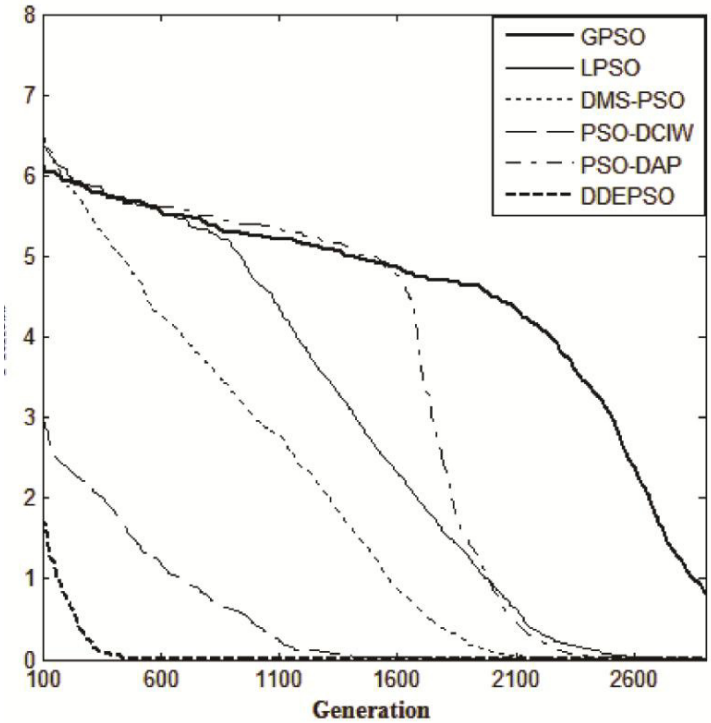

Fig. 6. Performance curves of GPSO, LPSO, DMS-PSO, PSO-DCIW, PSO-DAP, and DDEPSO for function f_{11}

6. Conclusion

Diversity level of the swarm is a very important point in the PSO algorithm and has a significant effect on its behavior; therefore, our approach enhances the diversity of the swarm with the new idea. In our approach, the algorithm can find the appropriate regions of search space by introducing new artificial particles based on historical information. After that, it carefully searches for these regions in which new artificial particles are located by using new formula to update velocities of artificial particles after replacement with converged particles. Subsequently, the algorithm meticulously moves in the search space and finds better solutions for problems. Experimental results illustrate DDEPSO can investigate the search space more appropriately compared to other approaches such as GPSO, LPSO, DMS-PSO, PSO-DCIW and PSO-DAP to find the better solution for different problems with different properties. This proves the potentiality of our approach which is problem-type-independent. Finally, it is concluded that the DDEPSO is an effective algorithm to solve different kinds of problems.

Acknowledgements

The authors would like to thank Dr. Ponnuthurai Nagaratnam Suganthan for sending his implementations which improved the paper's quality.

References

1. MohamadNezami O, Bahrampour A.*Particle Swarm Optimization algorithm based on Diversified Artificial Particles (PSO-DAP)*. The proceeding of 11th Iranian Conference on Intelligent Systems. Tehran. Iran;2013.
2. Eberhart R, Kennedy J.*A new optimizer using particle swarm theory*. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science; 1991. p. 39-43.
3. Kennedy J, Eberhart R. *Particle swarm optimization*. In Proceedings of the IEEE International Conference on Neural Networks (ICNN); 1995. p. 1942–1948.
4. Cheng S, Shi Y.*Diversity control in particle swarm optimization*. In Proceedings of the IEEE Symposium on Swarm Intelligence (SIS); 2011.
5. Zhan ZH, Zhang J, Li Y, Chung HS. Adaptive particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 2009; 39-6, p. 1362-1381.
6. Liang JJ, Qin AK, Suganthan PN, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transaction on Evolutionary Computation 2006;10-3, p. 281-295.
7. Li XD, Engelbrecht AP. *Particle swarm optimization: An introduction and its recent developments*. In Proceedings of the Genetic Evol. Comput.Conf; 2007. p. 3391-3414.
8. Wei C, He Z, Zheng Y, Pi W.*Swarm directions embedded in past evolutionary programming*. In Proceedings of the IEEE Congress on Evolutionary Computation; 2002. p. 1278-1283.
9. Jing J, Jianchao Z, Chongzhao H. *Adaptive Particle Swarm Optimization with Feedback Control of Diversity*. In Proceedings of the ICIC. LNBI 4115; 2006. p. 81-92.
10. Higashi H. Iba H. *Particle swarm optimization with gaussian mutation*. In Proceedings of the IEEE Swarm Intelligence Symposium; 2003. p. 72-79.
11. Secrest BR, Lamont GB.*Visualizing particle swarm optimization–gaussian particle swarm optimization*. In Proceedings of the IEEE Swarm Intelligence Symposium; 2003. p. 198-204.
12. Sriyanyong P.*Solving economic dispatch using particle swarm optimization combined with gaussian mutation*. In Proceedings of the ECTICON; 2008. p. 885-888.
13. Krohling RA. *Gaussian particle swarm with Jumps*. In Proceedings of the IEEE Congress on Evolutionary Computation. UK; 2005. p. 1226-1231.
14. Stacey A, Jancic M, Grundy I. Particle swarm Optimization with Mutation. In Proceedings of the IEEE Congress on Evolutionary Computation 2003; p. 1425-1430.
15. Dong D, Jie J, Zeng J, Wang M. *Chaos-mutation-based particle swarm optimizer for dynamic environment*. In Proceedings of the 3rd Int. Conf. on Intelligent System and Knowledge Engineering; 2008. p. 1032-1037.
16. YangM, Huang H, Xiao G. *A novel dynamic particle swarm optimization algorithm based on chaotic mutation*. In Proceedings of the Workshop on Knowledge Discovery and Data Mining; 2009. p. 656-659.
17. Yue-lin G, Xiao-hui A, Jun-min LA.*Particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation*. In Proceedings of the Conference on Computational Intelligence and Security. p. 61-65.
18. Riget J, Vestrstorm JS.*A Diversity Guided Particle Swarm Optimizer–The ARPSO*. Technical Report. Dept. of computer science, university of Aarhus, Denmark; 2002.
19. Parsopoulos KE, Vrahatis MN. On the computation of all global minimizers thorough Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation 2008; p. 211-224.

20. LoZvbjerg M, Krink T. *Extending particle swarms with self-organized criticality*. In Proceedings of the IEEE Congress on Evolutionary Computation; 2002. p. 1588-1593.
21. Blackwell T, Bentley PG. *Don't push me! collision-avoiding swarms*. In Proceedings of the IEEE Congress on Evolutionary Computation. 2002; p. 1691-1696.
22. Liang J, Ponnuthurai NS. *Dynamic multi-swarm particle swarm optimizer with local search*. In Proceedings of the IEEE Congr. Evol. Comput; 2005. p. 522-528.
23. Eberhart R, Dobbins R, Simpson P. Computational intelligence PC tools. Academic Press Professional 1996.
24. Chang PC, Huang WH, Ting CHJ. Dynamic diversity control in genetic algorithm for mining unsearched solution space in TSP problems. Expert systems with applications 2010; p. 1863-1878.
25. Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation 1992, 3-2, p. 82-102.

Table 1. Benchmark Functions used in our experimental study

Function	Function Definition	Range	Optimum
Sphere Function	$f_1(x)=\sum_{i=0}^{n-1}x_i^2$	[-100,100]	0
Schwefel function 1.2	$f_2(x)=\sum_{i=0}^{n-1}(\sum_{j=0}^{i}x_i)^2$	[-100,100]	0
Schwefel's Problem 1.2 with Noise	$f_3(x)=(\sum_{i=0}^{n-1}(\sum_{j=0}^{i}x_i)^2)*(1+0.4(\lvert N(0,1)\rvert))$	[-100,100]	0
Noisy Function	$f_4(x)=(\sum_{i=0}^{n-1}(i+1)x_i^4)+rand[0,1]$	[-1.28,1.28]	0
Rosenbrock Function	$f_5(x)=\sum_{i=0}^{n-1}100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	[-30,30]	0
Schwefel Function	$f_6(x)=418.9829n-\sum_{i=1}^{n}x_i\sin(\sqrt{\lvert x_i\rvert})$	[-500,500]	0
Rastringin Function	$f_7(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	[-5.12,5.12]	0
NoncontinuousRastringin Function	$f_8(x)=\sum_{i=1}^{n}[y_i^2-10\cos(2\pi y_i)+10]$	[-5.12,5.12]	0
	$y_i=\begin{cases}x_i & \lvert x_i\rvert<\frac{1}{2}\\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\geq\frac{1}{2}\end{cases}$		
Shaffer's Function	$f_9(x)=(\sum_{i=1}^{n}x_i^2)^{1/4}[\sin^2(50(\sum_{i=1}^{n}x_i^2)^{1/10})+1.0]$	[-32.767,32.767]	0
Griewank Function	$f_{10}(x)=\frac{1}{4000}\sum_{i=0}^{n-1}x_i^2+\sum_{i=0}^{n-1}\cos(\frac{x_i}{\sqrt{i+1}})+1$	[-600,600]	0
Rotated Ackley Function	$f_{11}(x)=20+e-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi z_i))$	[-32,32]	0
	$z=x*M$		
Rotated Rastrign's Function	$f_{12}(x)=\sum_{i=1}^{n}(z_i^2-10\cos(2\pi z_i)+10)$		
	$z=x*M$		

Table 2.Comparison results of GPSO, LPSO, DMS_PSO, PSO_DCIW,PSO-DAP, and DDEPSO for 20 particles of 30 dimensions in 3000 iterations

Test Functions		GPSO	LPSO	DMS_PSO	PSO_DCIW	PSO_DAP	DDEPSO
f_1	Mean Best	4.0926e+01	4.5896e-03	5.4071e-06	2.9697e-02	5.9606e-04	**5.3949e-15**

	Std.Dev	1.6831e+02	4.2145e-03	1.1863e-05	4.1784e-02	7.5794e-04	1.2708e-14
	P-value	2.9045e-01	1.0633e-04	5.5683e-02	4.9470e-03	2.3047e-03	7.2912e-02
	Mean Best	3.1620e+02	1.1798e+02	1.2512e+02	9.3692e+01	7.9997e+01	**1.4756e-07**
f_2	Std.Dev	6.7113e+01	3.4445e+01	6.0132e+01	2.5895e+01	8.6594e+01	1.5533e-07
	P-value	1.2322e-14	3.8042e-12	1.6577e-08	1.4412e-12	5.6751e-04	**4.3461e-04**
	Mean Best	4.4608e+02	3.4137e+02	2.9249e+02	2.4689e+02	6.5026e+01	**5.2983e-07**
$f3$	Std.Dev	9.7746e+01	8.5358e+01	1.3377e+02	6.8792e+01	5.9693e+01	5.2191e-07
	P-value	2.2029e-14	2.4028e-13	7.5458e-09	1.6645e-12	1.0596e-04	**2.2390e-04**
	Mean Best	5.4100e-02	4.4738e-02	2.0109e-02	2.8667e-02	2.5739e-02	**1.8287e-03**
f_4	Std.Dev	1.1989e-02	1.1224e-02	7.7876e-03	4.9322e-03	9.0694e-03	6.9651e-04
	P-value	2.7067e-14	2.5509e-13	4.9426e-10	2.5932e-16	9.9952e-11	**3.7381e-10**
	Mean Best	4.5126e+04	1.1850e+02	1.1676e+02	1.1581e+02	5.0003e+01	**3.4267e+01**
f_5	Std.Dev	6.5777e+04	1.0263e+02	1.5563e+02	5.6818e+01	4.4109e+01	2.6641e+01
	P-value	6.3292e-03	5.5224e-05	3.3236e-03	2.2880e-08	6.8048e-05	**1.5247e-05**
	Mean Best	3.5640e+03	3.8471e+03	3.1667e+03	5.2757e+03	**1.8862e+03**	2.4757e+03
f_6	Std.Dev	5.8963e+02	5.4001e+02	4.0226e+02	5.6450e+02	3.3864e+02	2.9547e+02
	P-value	1.2545e-16	5.8938e-18	9.1139e-19	3.6395e-20	**5.6939e-16**	2.8293e-19
	Mean Best	2.3781e+01	4.1652e+01	2.1889e+01	3.0560e+01	1.2132e+01	**2.3061e+00**
f_7	Std.Dev	6.4787e+00	9.0377e+00	7.7069e+00	6.1893e+00	4.7121e+00	3.2331e+00
	P-value	1.1148e-12	1.8419e-14	9.8640e-11	5.2292e-15	5.1899e-10	**4.8221e-03**
	Mean Best	2.4930e+01	4.2019e+01	2.1122e+01	4.0125e+01	1.6865e+01	**2.1043e+00**
f_8	Std.Dev	8.1783e+00	8.8871e+00	5.4456e+00	1.0156e+01	3.1445e+00	2.3881e+00
	P-value	2.9212e-11	1.1553e-14	4.1633e-13	2.9925e-13	1.1437e-15	**8.7738e-04**
	Mean Best	3.7490e+00	1.9214e+00	1.0255e+00	4.8289e+00	1.0704e+00	**3.4247e-01**
f_9	Std.Dev	1.5520e+00	1.1635e+00	1.1425e-01	4.5421e-01	2.2718e-01	4.1894e-01
	P-value	1.4963e-09	5.3775e-07	7.7726e-20	3.2134e-21	1.2301e-14	**1.6806e-03**
	Mean Best	3.7182e-01	1.2591e-02	1.8085e-02	3.4016e-02	6.5733e-03	**1.6003e-03**
f_{10}	Std.Dev	4.4251e-01	1.2069e-02	1.1279e-02	3.4149e-02	7.9336e-03	7.1568e-03
	P-value	1.3324e-03	1.6854e-04	8.1850e-07	2.7172e-04	**1.5014e-03**	3.2988e-01
	Mean Best	8.2268e-01	6.1272e-03	1.4716e-04	9.8631e-03	1.1676e-03	**2.2902e-09**
f_{11}	Std.Dev	1.3137e+00	2.7228e-03	1.1561e-04	3.9462e-03	7.2527e-04	1.5875e-09
	P-value	1.1409e-02	4.7495e-09	1.7335e-05	8.5117e-10	7.7269e-07	**3.4874e-06**
	Mean Best	4.4742e+01	8.0697e+01	3.1756e+01	7.8214e+01	3.7760e+01	**6.6165e+00**
f_{12}	Std.Dev	1.2293e+01	1.6660e+01	8.3386e+00	2.5239e+01	2.0518e+01	1.1104e+01
	P-value	1.2971e-12	7.4237e-15	5.7791e-13	2.1947e-11	1.0970e-07	**1.5308e-02**

Table 3.Comparison results of GPSO, LPSO, DMS_PSO, PSO_DCIW, PSO-DAP, and DDEPSO for 50 particles of 50 dimensions in 10000 iterations

Test Functions		GPSO	LPSO	DMS_PSO	PSO_DCIW	PSO_DAP	DDEPSO
	Mean Best	4.6654e+02	1.2605e-06	2.4968e-13	8.9220e-05	9.7664e-11	**1.4953e-28**
f_1	Std.Dev	1.4037e+03	1.5399e-06	3.5123e-13	1.2986e-04	1.5701e-10	2.4918e-28
	P-value	1.5359e-01	1.6619e-03	4.9402e-03	6.2662e-03	1.1886e-02	**1.4695e-02**
f_2	Mean Best	9.2716e+02	3.1334e+02	3.4233e+02	3.3668e+02	1.5804e+02	**1.6427e-09**

	Std.Dev	1.1952e+02	6.9740e+01	7.6679e+01	8.5808e+01	1.1148e+02	2.1230e-09
	P-value	1.1998e-18	2.9262e-14	3.2863e-14	3.3879e-13	4.3933e-06	**2.6201e-03**
	Mean Best	1.1943e+03	1.3066e+03	7.3504e+02	8.4019e+02	1.8833e+02	**1.8079e-09**
$f3$	Std.Dev	1.3210e+02	2.4991e+02	1.3078e+02	1.6610e+02	1.5827e+02	2.1169e-09
	P-value	6.7897e-20	1.8301e-15	4.8243e-16	3.3544e-15	3.8942e-05	**1.1575e-03**
	Mean Best	1.3874e-01	9.0294e-02	2.9539e-02	7.0970e-02	3.0653e-02	**5.6180e-04**
f_4	Std.Dev	1.7213e-02	1.3980e-02	9.0828e-03	1.0274e-02	6.6863e-03	2.3991e-04
	P-value	5.8588e-19	3.6615e-17	9.4565e-12	1.0477e-17	2.0282e-14	**2.4897e-09**
	Mean Best	2.8016e+05	9.8139e+01	8.2831e+01	1.8373e+02	7.3112e+01	**9.8728e+00**
f_5	Std.Dev	2.4902e+05	3.7019e+01	5.7348e+01	1.4864e+02	1.0706e+02	1.6159e+01
	P-value	7.4118e-05	3.1772e-10	3.4330e-06	2.4783e-05	6.5286e-03	**1.3230e-02**
	Mean Best	8.2761e+03	6.6019e+03	5.5454e+03	7.9751e+03	**3.7025e+03**	4.2235e+03
f_6	Std.Dev	2.6300e+03	4.6987e+02	8.3730e+02	4.9911e+02	5.1421e+02	3.3760e+02
	P-value	1.6818e-11	1.6602e-23	2.2962e-17	1.4571e-24	**4.8309e-18**	1.4910e-22
	Mean Best	2.8898e+01	6.0988e+01	2.0496e+01	3.8953e+01	6.2579e+00	**0.0000e+00**
f_7	Std.Dev	8.6475e+00	8.7955e+00	5.4630e+00	8.6817e+00	5.3820e+00	0.0000e+00
	P-value	5.8724e-12	9.7648e-18	7.5501e-13	3.0008e-14	5.0961e-05	**NaN**
	Mean Best	3.1897e+01	6.5129e+01	2.3800e+01	4.7663e+01	1.1892e+01	**0.0000e+00**
f_8	Std.Dev	7.2006e+00	8.5145e+00	6.8411e+00	1.1851e+01	8.5193e+00	0.0000e+00
	P-value	3.7857e-14	1.5602e-18	2.8883e-12	2.1709e-13	5.3859e-06	**NaN**
	Mean Best	3.0279e+00	1.6296e+00	1.3320e+00	4.8615e+00	1.0069e+00	**1.2300e-01**
f_9	Std.Dev	1.6107e+00	9.7312e-02	9.1206e-02	2.7865e-01	2.4564e-01	2.1735e-01
	P-value	7.9572e-08	5.9954e-25	7.9918e-24	2.7570e-25	1.5430e-13	**2.0373e-02**
	Mean Best	1.0203e+00	8.7172e-04	1.2729e-02	3.6399e-02	1.2941e-03	**0.0000e+00**
f_{10}	Std.Dev	1.2977e+00	2.6875e-03	2.0802e-02	4.5395e-02	3.0314e-03	0.0000e+00
	P-value	2.3079e-03	1.6320e-01	1.3110e-02	1.9709e-03	7.1466e-02	**NaN**
	Mean Best	1.4754e+00	5.6186e-05	6.2226e-08	6.0750e-05	1.2249e-06	**2.1849e-14**
f_{11}	Std.Dev	1.9505e+00	2.4692e-05	9.3639e-08	1.0532e-04	2.0337e-06	2.6473e-15
	P-value	3.1232e-03	3.9690e-09	7.8341e-03	1.8363e-02	1.4388e-02	**3.7576e-19**
	Mean Best	4.9520e+01	1.1892e+02	2.7461e+01	8.1892e+01	**2.2934e+01**	3.6714e+01
f_{12}	Std.Dev	1.2545e+01	2.1770e+01	5.3082e+00	2.7493e+01	2.7188e+01	3.7388e+01
	P-value	3.0414e-13	8.1603e-16	2.2211e-15	4.3559e-11	**1.2884e-03**	3.1375e-04