Dynamic Multi-Swarm Particle Swarm Optimizer

J. J. Liang and P. N. Suganthan
School of Electrical and Electronic Engineering,
Nanyang Technological University, Singapore 639798
liangjing@pmail.ntu.edu.sg, epnsugan@ntu.edu.sg

ABSTRACT

In this paper, a novel dynamic multi-swarm particle swarm optimizer (PSO) is introduced. Different from the existing multi-swarm PSOs and the local version of PSO, the swarms are dynamic and the swarms' size is small. The 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. Experiments are conducted on a set of shifted rotated benchmark functions and results show its better performance when compared with some recent PSO variants.

1. INTRODCTION

Particle swarm optimizer (PSO), introduced by Kennedy and Eberhart in 1995 [1][2], emulates flocking behavior of birds to solve the optimization problems. In PSO, each solution is regarded as a particle. All particles have fitness values and velocities. The particles fly through the *D* dimensional problem space by learning from the historical information of all the particles. Therefore, the particles have a tendency to fly towards better search area over the course of search process. The velocity Vi^d and position Xi^d updates of dth dimension of the ith particle are presented below:

$$v_i^d = \omega * v_i^d + c_1 * rand1_i^d * (pbest_i^d - x_i^d)$$
$$+c_2 * rand2_i^d * (gbest^d - x_i^d)$$
$$x_i^d = x_i^d + v_i^d$$

where c_1 and c_2 are the acceleration constants, $rand1_i^d$ and $rand2_i^d$ are two uniform distributed random numbers in the range [0,1]. $\mathbf{x_i} = (x_i^1, x_i^2, ..., x_i^D)$ is the position of the ith particle; $\mathbf{pbest}_i = (pbest_i^1, pbest_i^2, ..., pbest_i^D)$ is the best previous position yielding the best fitness value $pbest_i$ for the ith particle; $\mathbf{gbest} = (gbest^1, gbest^2, ..., gbest^D)$ is the best position discovered by the whole population; $\mathbf{v}_i = (v_i^1, v_i^2, ..., v_i^D)$ represents the rate of the position change (velocity) for particle i. ω is the inertia weight used to balance between the global and local search abilities.

In the PSO world, there exist global and local PSO versions. Instead of learning from the personal best and the best position achieved so far by the whole population, in the local version of PSO, each particle's velocity is adjusted according to its personal best and the best performance achieved so far within its neighborhood. Focusing on improving the local version of PSO, different neighborhood structures are proposed and discussed.

Kennedy claimed that PSO with large neighborhood would perform better for simple problems and PSO with small neighborhoods might perform better on complex problems [3]. Kennedy and Medes discussed the effects of different neighborhood topological structures on the local version PSO [4]. Suganthan applied a combined version of PSO where a local version PSO is run first followed by a global version of PSO at the end [5]. Hu and Eberhart proposed a dynamically adjusted neighborhood when they solve the multi-objective optimization problems using PSO [6]. In their dynamically adjusted neighborhood, for each particle, the *m* closest particles are selected to be its new neighborhood. Veeramachaneni and his group developed a new version of PSO, Fitness-Distance-Ratio based PSO (FDR-PSO), with near neighbor interactions [7]. When updating each velocity dimension, the FDR_PSO algorithm selects one other particle, *nbest*, which has higher fitness value and near the particle being updated, in the velocity updating equation. In Mendes and Kennedy's fully informed particle swarm optimization algorithm, all the neighbors of a particle are weighted and used to calculate the velocity [8].

Except these local versions of PSO, some variants which use multi-swarm [9], subpopulation [10] can also be included in the local version PSOs if we treat the sub-groups as a special neighborhood structure. In the existing local versions of PSO with different neighborhood structures and the multi-swarm PSOs, the swarms are predefined or dynamically adjusted according to the distance. Hence, the freedom of the swarms is limited. In this paper, we propose a dynamic multi-swarm particle swarm optimizer (DMS-PSO) whose neighborhood topology is dynamic and randomly assigned.

The paper is organized as follows. Section 2 describes dynamic multi-swarm particle swarm optimizer. Section 3 defines the benchmark continuous optimization problems used for experimental comparison of the algorithms, the experimental setting for each algorithm, and the discussions of the results. The conclusions are given Section 4

2. DYNAMIC MULTI-SWARM PARTICLE SWARM OPTIMIZER

The dynamic multi-swarm particle swarm optimizer is constructed based on the local version of PSO and a new neighborhood topology is used. This new neighborhood structure has two important characters:

1) Small Sized Swarms

Not as other evolutionary algorithms that prefer larger population, PSO needs a comparatively smaller population size. Especially for simple problems, a population with three to five particles can achieve satisfactory results. PSO with small neighborhoods performs better on complex problems. Hence, in the new version, small neighborhoods are used. In order to slow down the population's convergence velocity and increase diversity, in the DMS-PSO, we divide the population into small sized swarms. Each swarm uses its own members to search for better area in the search space.

2) Randomly Regrouping Schedule

Since the small sized swarms are searching using their own best historical information, they are easy to converge to a local optimum because of PSO's convergence property. In this case, if we keep the neighborhood structures unchanged, then there will be no information exchange among the swarms, amd it will be a co-evolutionary PSO with these swarms searching in parallel. In order to avoid this situation, a randomized regrouping schedule is introduced. Every R generations, the population is regrouped randomly and starts searching using a new configuration of small swarms. Here R is called regrouping period. In this way, the good information obtained by each swarm is exchanged among the swarms. Simultaneously the diversity of the population is increased. The new neighborhood structure has more freedom when compared with the classical neighborhood structure. It is not surprising that it performs better on complex multimodal problems.

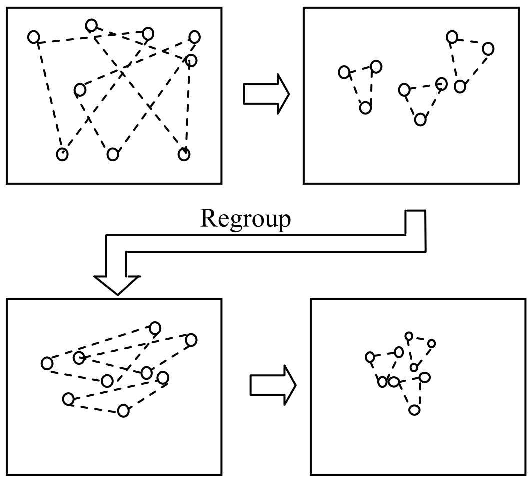

Figure 1. DMS-PSO's Search

In Figure 1, we use three swarms with three particles in each swarm to show the regrouping schedule. First, the nine particles are divided into three swarms randomly. Then the three swarms use their own particles to search for better solutions. In this period, they may converge to near a local optimum. Then the whole population is regrouped into new swarms. The new swarms begin their search. This process is continued until a stop criterion is satisfied. With the randomly regrouping schedule, particles from different swarms are grouped in a new configuration so that each small swarms search space is enlarged and better solutions are possible to be found by the new small swarms.

In the end of the search, in order to perform a better local search, all particles form a single swarm to become a global PSO version. The flowchart of DMS-PSO is given in Figure 2.

```
m: Each swarm's population size
n: Swarms' number
R: Regrouping period
Max_gen: Max generations, stop criterion
Initialize m*n particles (position and velocity)
Divide the population into n swarms randomly,
with m particles in each swarm.
For i=1:0.9*Max_gen
    Update each swarm using local version PSO
    If mod(i, R)==0,
        Regroup the swarms randomly,
    End
For i=0.9*Max_gen:Max_gen
    Update all particles using global version PSO
End
```

Figure 2. DMS-PSO

3. EXPERIMENTS

Part I. Introduction to the Benchmark Functions

In the experiments, six 10 dimensional benchmark functions with different properties are chosen to test DMS-PSO's performance. The equations are listed below:

1) Sphere Function

$$f(x)=\sum_{i=1}^{D}x_i^2 \qquad \text{where} \quad x\in[-5.12,5.12]^D$$

2) Rosenbrock's Function

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

$$\text{where} \quad x\in[-2.048,2.048]^D$$

3) Ackley's Function

$$f(x)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})$$

$$-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$$

$$\text{where} \quad x\in[-32.768,32.768]^D$$

4) Griewank's Function

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

$$\text{where} \quad x\in[-600,600]^D$$

5) Rastrigin's Function

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

$$\text{where} \quad x\in[-5.12,5.12]^D$$

6) Weierstrass Function

$$f(x)=\sum_{i=1}^{D}(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i+0.5))])$$

$$-D\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k\cdot 0.5)]$$

where a=0.5, b=3, k_{max}=20, $x\in[-0.5,0.5]^D$

Among the six benchmark functions, sphere function is a simple unimodal function and it is the easiest one among these six functions. Rosenbrock's function may be regarded as either unimodal or multimodal function. Though it seems to have local optima, the algorithms which solve unimodal problems well, yield better performance on Rosenbrock's function too. Functions 3-6 are all multimodal problems. Ackley's Function has a narrow global basin. Griewank's Function has linkage among dimensions. Rastrigin's Function has a huge number of local optima. Weierstrass Function is continuous but differentiable only on a set of points.

In order to avoid some problems existing in the benchmark functions as we have analyzed in [11], we shifted the global optimum points and rotated the test functions according to Salomon's method [12] except sphere function since sphere is still a sphere after rotation.

Part II. Parameters Setting for DMS-PSO

To solving these test functions, population size is set at 30 and *Max_Gen* is set at 2000. *w* is reducing with increasing generations from 0.9 to 0.2. Particles' velocities are restricted by *Vmax*, where *Vmax* is equal to 20% of the search range.

Except these common parameters used in PSO, there are two additional parameters in DMS-PSO need to be specified. One is each swarm's population size *m* and another is regrouping period *R*.

First we change *m* and fix regrouping period *R* at 10. DMS-PSO is run 20 times on each test function. The Results obtained are presented in Table 1. From the results, we could observe when m=2, the performance is the worst, and when m=5, it gives better performance on the Unimodal function Rosenbrock's function and comparable simple multimodal function Ackley's Functions, while when m=3, it gives better performance for more complex multimodal problems. Two particles are not sufficient to construct a good swarm, while five particles give better local search ability and reduced global search ability. Three particles achieve the balance between them and the swarms show better global search ability.

Table 1. Results achieved under different *m*

m / *Func*	2	3	5
1	0	0	0
2	3.0127e+000	1.3612e+000	**9.9970e-001**
3	3.1974e-015	2.1316e-015	**1.4211e-015**
4	3.2935e-002	**3.2496e-002**	3.7349e-002
5	4.9325e+000	**3.6068e+000**	5.3728e+000
6	8.6547e-002	**1.2269e-004**	1.0408e-005

After fixing sub-swarm size *m* at 3, DMS-PSO is tested with different regrouping period *R* and results are recorded in Table 2. Except Rosenbrock's Function for which the best result is achieved when *R=20,* for most test functions the better results are achieved when *R=5*.

Table 2. Results achieved under different *R*

R / *Func*	*2*	*3*	*5*
1	**0**	**0**	**0**
2	1.7154e+000	1.5024e+000	1.0910e+000
3	**0**	**0**	**0**
4	5.3793e-002	4.7304e-002	**2.3389e-002**
5	5.9381e+000	3.2946e+000	**3.2767e+000**
6	7.1016e-003	3.8061e-003	**0**

Table 2 (Cond.). Results achieved under different *R*

n / *Func*	*10*	*20*	*50*
1	**0**	**0**	**0**
2	1.3612e+000	**8.1698e-001**	1.7155e+000
3	2.1316e-015	3.1974e-015	4.6185e-015
4	3.2496e-002	5.2159e-002	3.6415e-002
5	3.6068e+000	4.7446e+000	5.5718e+000
6	1.2269e-004	6.9781e-002	1.9784e-001

Part III. Comparison with Other PSO Algorithms

In this part, we tested the same six 10 dimensional benchmark test functions using the following eight PSO variants:

- The modified PSO with inertia weight (PSO_w) [13]
- Global Version of PSO with constriction factors (PSO_cf) [14]
- Local Version of PSO with inertia weight (PSO_w_local)
- Local Version of PSO with constriction factors (PSO_cf_local) [4]
- Unified Particle Swarm Optimization (UPSO) [15]
- Fitness-distance-ratio based particle swarm optimization (FDR–PSO) [7]
- Cooperative PSO (CPSO-H) [16]
- Dynamic multi-swarm PSO (DMS-PSO)

Each algorithm is run 20 times. Except CPSO, population size is set at 30 and *Max_Gen* is set at 2,000 for all the other algorithms. Considering high FES requirement of CPSO-H in each generation, its population size is set at 10.

and *Max_Gen* is set at 550. Hence, the maximum fitness Evaluations is approximately 60,000 for all algorithms. The size of sub-swarms, *m* was set at 3 and the regrouping period *R* was set at 5 for DMS-PSO. In PSO local versions PSO_w_local and PSO_cf_local, the best reported neighborhood topology von Neumann neighborhoods where neighbors above, below, and one each side on a two-dimensional lattice were connected was used. The results are given in Table 3.

Table 3. Results achieved by different PSOs

Func / *Algorithm*	*1*	*2*	*3*
PSO_w	**0**	3.5613e+000	1.6462e-001
PSO_cf	**0**	9.4853e-001	2.8014e-001
PSO_w_local	**0**	4.7613e+000	2.7001e-014
PSO_cf_local	**0**	2.0034e+000	**0**
UPSO	**0**	1.2945e+000	**0**
FDR-PSO	**0**	**2.6422e-001**	3.1974e-015
CPSO	**0**	4.4890e+000	3.2048e+000
DMS_PSO	**0**	1.0910e+000	**0**

Table 3 (Cond.). Results achieved by different PSOs

Func / *Algorithm*	*4*	*5*	*6*
PSO	1.5702e-001	1.0945e+001	1.6495e-001
PSO_cf	1.1320e-001	1.5919e+001	1.5106e+000
PSO_local	5.0928e-002	7.3698e+000	7.0933e-003
PSO_cf_local	4.9718e-002	9.7694e+000	1.8117e-001
UPSO	3.4530e-002	1.4231e+001	2.2264e+000
FDR-PSO	1.1596e-001	8.0592e+000	3.3068e-001
CPSO	1.5276e-001	2.1392e+001	4.2082e+000
DMS_PSO	**2.3389e-002**	**3.2767e+000**	**0**

From the results, we can observe that among the eight algorithms, all algorithms attain the same good results on the sphere function, FDR-PSO surpasses others algorithms on Rosenbrock's function because of its better local search ability, while DMS-PSO performs the best on the four multimodal problems. Especially on the more difficult multimodal problem Rastrigin's Function, DMS-PSO performs much better than other seven algorithms. For the multi-modal problems, the local PSO variant always yield better performance when compared with the global PSO variant.

4. CONCLUSIONS

A novel dynamic multi-swarm PSO (DMS-PSO) is introduced and discussed in this paper. In order to achieve a better diversity, more freedom is given to the DMS-PSO. The population is divided into many small swarms, and these swarms are regrouped frequently to exchange the information among the swarms. Through combining the exploitation and exploration together, this neighborhood structure gives better performance on complex multi-modal problems when compared with some other PSO variants.

5. REFERENCES

[1] R. C. Eberhart, and J. Kennedy, "A new optimizer using particle swarm theory", *Proc. of the Sixth Int. Symposium on Micromachine and Human Science*, Nagoya, Japan. pp. 39-43, 1995

[2] J. Kennedy, and R. C. Eberhart, "Particle swarm optimization ". *Proceedings of IEEE International Conference on Neural Networks*, Piscataway, NJ. pp. 1942-1948, 1995

[3] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance ". *Proc. of IEEE Congress on Evolutionary Computation (CEC 1999),* Piscataway, NJ. pp. 1931-1938, 1999

[4] J. Kennedy and R. Mendes, "Population structure and particle swarm performance ". *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2002),* Honolulu, Hawaii USA. 2002

[5] P. N. Suganthan, "Particle swarm optimizer with neighborhood operator," *Proc. of the IEEE Congress on Evolutionary Computation (CEC 1999)*, Piscataway, NJ, pp. 1958-1962, 1999.

[6] X. Hu and R. C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle

swarm optimization," Proc. IEEE Congress on Evolutionary Computation (CEC 2002), Hawaii, pp. 1677-1681, 2002.

[7] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," *Proc. IEEE Swarm Intelligence System*, Indianapolis, Indiana, USA, pp. 174-181, 2003.

[8] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better". IEEE Transactions on Evolutionary Computation, 8(3):204 - 210, June 2004

[9] T. M. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments". LNCS No. 3005: *Proceedings of Applications of Evolutionary Computing: EvoWorkshops* 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoISAP, EvoMUSART, and EvoSTOC, Coimbra, Portugal. pp. 489-500, 2004

[10] M. Løvbjerg, T. K. Rasmussen and T. Krink, "Hybrid particle swarm optimiser with breeding and subpopulations ". *Proc. of the Genetic and Evolutionary Computation Conference*, (GECCO 2001), 2001

[11] J. J. Liang, P. N. Suganthan and K. Deb, "Novel Composition Test Functions for Numerical Global Optimization", submitted to *IEEE International Swarm Intelligence Symposium*, 2005.

[12] R. Salomon, "Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions," *BioSystems*, vol. 39, pp. 263-278, 1996.

[13] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer", *Proc. of the IEEE Congress on Evolutionary Computation (CEC 1998)*, Piscataway, NJ, pp. 69-73, 1998.

[14] Clerc, M. and Kennedy, J., "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE T. on Evol. Computation*, vol. 6, no. 1, pp. 58-73, 2002.

[15] K. E. Parsopoulos and M. N. Vrahatis, "UPSO- A Unified Particle Swarm Optimization Scheme", *Lecture Series on Computational Sciences*, 2004, pp. 868-873

[16] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, 8(3):225 - 239, June 2004.