International Journal of
Intelligent Systems and Applications in Engineering

ISSN:2147-6799 www.ijisae.org Original Research Paper

Particle Swarm Optimization with Flexible Swarm for Unconstrained Optimization

Humar Kahramanlı,*[a] and N. Allahverdi [b]

Received 21th February 2013, Accepted 11th March 2013

Abstract: *Particle Swarm Optimization (PSO) algorithm inspired from behaviour of bird flocking and fish schooling. It is well-known algorithm which has been used in many areas successfully. However it sometimes suffers from premature convergence. In resent year's researches have been introduced a various approaches to avoid of this problem. This paper presents the particle swarm optimization algorithm with flexible swarm (PSO-FS). The new algorithm was evaluated on 14 functions often used to benchmark the performance of optimization algorithms. PSO-FS algorithm was compared to some other modifications of PSO. The results show that PSO-FS always performed one of the better results.*

Keywords: *Particle Swarm Optimization Algorithm, Particle Swarm Optimization Algorithm with Flexible Swarm, Unconstrained Optimization.*

1. Introduction

I Particle Swarm Optimization (PSO) algorithm has been developed by Eberhart and Dr. Kennedy, inspired by behaviour of bird flocking and fish schooling (Kennedy and Eberhart, 1995). PSO is a stochastic population-based global optimization technique. In PSO context the population is called as swarm, while the members of population are called as particle. The algorithm is made of two essential steps: particle movement (dynamics) through a search space of solutions and an indirect particle interaction, which is mediated by information sharing within a social network (Blackwell and Bratton, 2008). PSO uses the physical movements of the individuals in the swarm and has a flexible and well-balanced mechanism to enhance and adapt to the global and local exploration abilities. Because of its easy implementation and inexpensive computation, its simplicity in coding and consistency in performance, the PSO has proved to be an effective and competitive algorithm for the optimization problem in continuous space (Marinakis and Marinak, 2008). PSO has many key advantages over other optimization techniques like (Alrashidi and El-Hawary, 2006):

• It is a derivative-free algorithm unlike many conventional techniques.
• It has the flexibility to be integrated with other optimization techniques to form a hybrid tool.
• It is less sensitive to the nature of the objective function, i.e., convexity or continuity.
• It has less parameter to adjust unlike many other competing evolutionary techniques.
• It has the ability to escape local minima.
• It is easy to implement and program with basic mathematical and logic operations.
• It can handle objective functions with stochastic nature.
• It does not require a good initial solution to start its iteration process.

PSO has been successfully applied in many different application areas due to its robustness and simplicity. In comparison with other stochastic optimization techniques, PSO have fewer complicated operations and fewer defining parameters, and can be coded in just a few lines (Wanget al., 2007). But this algorithm sometimes may suffer from premature convergence problem. There have been many researchs to improve the performance of the original PSO algorithm.

Bergh and Engelbrecht (2004) proposed a cooperative particle swarm frame (CPSO), the cooperative particle swarm optimizer. The authors used multiple swarms to optimize different components of the solution vector cooperatively. They tested CPSO over five benchmark optimization problems. The tests showed that CPSO reaches significantly better solution than PSO. Pena, Upegui and Sanchez (2006) presented a hybrid bio-inspired optimization technique that introduces the concept of discrete recombination in a particle swarm optimizer (PSODR), obtaining a simple and powerful algorithm, well suited for embedded applications. Proposed algorithm validated over four optimization problems and noted that PSODR shows a better performance than the standard PSO algorithm. Baskar and.Suganthan (2004) present PSO (CONPS0) algorithm to alleviate the premature convergence problem of PSO algorithm. CONPSO is a type of parallel algorithm in which modified PSO and FDR-PSO algorithms are simulated concurrently with frequent message passing between them. To demonstrate the effectiveness of the proposed algorithm experiments were conducted on six benchmark optimization problems. Results clearly demonstrate the superior performance of the proposed algorithm. Kok and Snyman (2008) proposed dynamic interacting particle swarm optimization algorithm (DYN-PSO). In this method, the

[a] *Faculty of Technology, Selcuk University Campus,Konya,Turkey.*
Tel: +90 332 2233330; E-mail:hkahramanli@selcuk.edu.tr
[b] *Faculty of Technology, Selcuk University Campus,Konya,Turkey.*
Tel: +90 332 2233329; E-mail:noval@selcuk.edu.tr
** Corresponding Author*

minimization of a function is achieved through the dynamic motion of a strongly interacting particle swarm, where each particle in the swarm is simultaneously attracted by all other particles located at positions of lower function value. The force of attraction experienced by a particle at higher function value due to a particle at a lower function value is equal to the difference between the respective function values divided by their stochastically perturbed position difference. The resultant motion of the particles under the influence of the attracting forces is computed by solving the associated equations of motion numerically. An energy dissipation strategy is applied to each particle. The specific chosen force law and the dissipation strategy result in the rapid collapse (convergence) of the swarm to a stationary point. Obtained results of seven benchmark optimization problem show that, in comparison to the standard particle swarm algorithm, the proposed DYN-PSO algorithm is promising. Yan Jiang, Tiesong Hu, ChongChao Huang, Xianing Wu (2007) proposed an improved particle swarm optimization (IPSO). In IPSO, a particle sampled randomly from the feasible space. Then the population is partitioned into several sub-swarms, each of which is made to evolve based on particle swarm optimization (PSO) algorithm. At periodic stages in the evolution, the entire population is shuffled, and then points are reassigned to sub-swarms to ensure information sharing. Simulations for three benchmark test functions show that IPSO possesses better ability to find the global optimum than that of the standard PSO algorithm. Ali and Kaelo (2008) proposed some modifications in the position update rule of particle swarm optimization algorithm in order to make the convergence faster. These modifications result in two new versions of the particle swarm optimization algorithm. A numerical study is carried out using a set of 54 test problems some of which are inspired by practical applications. Results show that the new algorithms are much more robust and efficient than some existing particle swarm optimization algorithms. Yuxin Zhao, Wei Zu, Haitao Zeng (2009) proposed a new modified algorithm to ensure the rational flight of every particle's dimensional component. Meanwhile, two parameters of particle-distribution-degree and particle-dimension-distance are introduced into the proposed algorithm in order to avoid premature convergence. Simulation results of the new PSO algorithm show that it has a better ability of finding the global optimum, and still keeps a rapid convergence as with the standard PSO. S. He, Q.H. Wu, J.Y. Wen, J.R. Saunders, R.C. Paton (2004) presented a particle swarm optimizer with passive congregation to improve the performance of standard PSO. By introducing passive congregation to PSO, information can be transferred among individuals of the swarm. A particle swarm optimizer with passive congregation (PSOPC) is tested with a set of 10 benchmark functions with 30. Experimental results indicate that the PSO with passive congregation improves the search performance on the benchmark functions significantly. Qi Kang, Lei Wang, Qi-di Wu (2008) presented a particle swarm optimization algorithm from the angle of ecological population evolution, called the ecological particle swarm optimization (EPSO). From the basis of the ecological population competition model, the EPSO algorithm and its general framework are proposed; in which particle swarm system with ecological hierarchy and competition model is defined and two collocating strategies of inertia weight factor are considered. The convergence performance and population dynamics including population aggregation and population diversity of the proposed approach are discussed separately through empirical simulations with four benchmarks optimization problems. Chen (2011) proposed a two-layer particle swarm optimization (TLPSO) to increase the diversity of the particles so that the drawback of trapping in a local optimum is avoided. In order to design the TLPSO, a structure with two layers (top layer and bottom layer) is proposed so that M swarms of particles and one swarm of particles are generated in the bottom layer and the top layer, respectively. Each global best position in each swarm of the bottom layer is set to be the position of the particle in the swarm of the top layer. Therefore, the global best position in the swarm of the top layer influences indirectly the particles of each swarm in the bottom layer so that the diversity of the particles increases to avoid trapping into a local optimum. Besides, a mutation operation is added into the particles of each swarm in the bottom layer so that the particles leap the local optimum to find the global optimum. Nine optimization problems were used to illustrate the efficiency of the proposed method. The author noted that proposed TLPSO approach is superior to the other evolutionary algorithms in the ability to finding the global optimum solution. Bratton and Blackwell (2008) introduced PSO with discrete recombination model (PSO-DR). They have noted that the PSO-DR variant is important not only because of its improved performance on several benchmark functions, but also because its simplified state allows us to examine what happens to the standard algorithm when pieces are modified or removed. PSO-DR has been tested over 14 benchmark problem. Bratton and Kennedy (2007) have compared three PSO algorithms in the interest of demonstrating the performance gains granted by improvements to the technique: the original algorithm, a constricted swarm using a global topology, and a constricted swarm using a local topology. The algorithms have been tested over 14 benchmark problem. Chen and Chi (2010) have tuned some of the parameters and added mechanisms to the PSO algorithm in order to improve its robustness in finding the global solution. This approach has been tested over 10 problems. Abd-El-Wahed,Mousa, and El-Shorbagy (2011) introduced a hybrid approach combining two heuristic optimization techniques, particle swarm optimization (PSO) and genetic algorithms (GA). The method integrates the merits of both GA and PSO and it has two characteristic features. Firstly, the algorithm is initialized by a set of random particles which travel through the search space. During this travel an evolution of these particles is performed by integrating PSO and GA. Secondly, to restrict velocity of the particles and control it, the authors introduce a modified constriction factor. Finally, the results of various experimental studies using a suite of multimodal test functions taken from the literature have demonstrated the superiority of the proposed approach to finding the global optimal solution. Akbari and Ziarati (2011) presented a variation on the standard PSO algorithm called the rank based particle swarm optimizer (PSOrank). In this method, in order to efficiently control the local search and convergence to global optimum solution, the γ best particles are taken to contribute to the updating of the position of a candidate particle. The contribution of each particle is proportional to its strength. The strength is a function of three parameters: Strivness, immediacy and number of contributed particles. All particles are sorted according to their fitness values, and only the γ best particles will be selected. The value of γ decreases linearly as the iteration increases. A time-varying inertia weight decreasing non-linearly is introduced to improve the performance. PSO rank is tested on five commonly used optimization problems.

In this study PSO-FS model has been proposed. The work is organized as follows: In Section 2 and Section 3 classical PSO

algorithm and new PSO-FS algorithms are presented respectively. In Section 4, Experimental Results and Comparison has been explained. In Section 5 this paper is concluded.

2. Overview of the standard PSO

Since PSO is a population based optimization method, first a population of random particles must be generated. Suppose that the search space is D-dimensional and the swarm consists of N particles. The ith particle is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, i=1,2,…N. A current velocity of i^{th} particle is represented as v), i=1,2,…N. The personal best position achieved by i^{th} particle is represented by
and the best position achieved by swarm is represented by). The velocity of particle and its next position is updated according to the following equations (Bratton and Kennedy, 2007):

$$[v \quad (p \quad) + \quad (p \quad)] \tag{1}$$

$$\tag{2}$$

where i=1,2,…N, j=1, 2,…D. χ is a parameter called as constriction factor which is used to limit the maximum velocity; c_1 and c_2 are two positive constants called cognitive and social parameter, respectively; and r_1, r_2, are random numbers in the range [0, 1].
The execution of the PSO algorithm is as follows:

1. Initialization: randomly initialize a population of particles, set parameters c_1, c_2, r_1, r_2, χ .

2. Population loop: for each particle x_i do:

2.1. Goodness evaluation and update: evaluate the goodness x) using corresponding fitness function of the particle. If $f(x_i) > Pbest_i$, namely its goodness is greater than its best goodness so far, then this particle becomes the best particle found so far.

2.2. Neighbourhood evaluation: if $f(x_i) > Gbest_i$, namely the goodness of this particle is the best among all its neighbors, then this particle becomes the best particle of the whole neighbourhood.

2.3. Determine v_i: Update the velocity v_i using Equation 1.

2.4. Particle update: Update the position x_i using Equation 2.

3. Cycle: repeat Step 2 until a given termination criterion is met.

3. PSO with flexible swarm

In this method, natural selection has been applied to swarm. For this after each iteration particles were scored between *1*-*N*, where successful particle has *N*, unsuccessful particle has *1* point. After *m* iteration higher score maxs and lower score mins has been found. Using Equation 3 valid score vs has been calculated.

$$\sqrt{\text{m}} \tag{3}$$

The particles, which score less than vs were removed and a new particles has been generated instead it.
The execution of the algorithm is as follows:

1. Initialization: randomly initialize a population of particles, set parameters c_1, c_2, r_1, r_2, χ , m. Set parameter *iter*=0.

2. Iteration loop: Set *iter*=*iter*+1.

3. Population loop: for each particle x_i , do:

3.1. Goodness evaluation and update: evaluate the goodness x) using corresponding fitness function of the particle. If $f(x_i) > Pbest_i$, namely its goodness is greater than its best goodness so far, then this particle becomes the best particle found so far.

3.2. Neighbourhood evaluation: if $f(x_i) > Gbest_i$, namely the goodness of this particle is the best among all its neighbours, then this particle becomes the best particle of the whole neighbourhood.

3.3. Determine v_i: Update the velocity v_i: using Equation 1.

3.4. Particle update: Update the position x using Equation 2.

3.5. Determine s_i: Determine the scor s_i .

3.6. Revision of swarm: If *iter*=*m*, then set *iter*=0, determine *vs* using Equation 3, wipe out particles, which score less than *vs* and compose new particle instead of they.
4. Cycle: Go to step 2 until a given termination criterion is met.

4. Experimental Results and Comparison

Algorithms have been tested over of 14 benchmark functions for 300000 iterations for each function. Testing functions, feasible bounds, optimum points and dimension of functions are shown in Table 1 and Table 2. Functions f_1-f_3 are functions with a single minimum, f_4-f_9 are complex problems, with a number of local minimas and a single global minimum. Functions f_1-f_9 are high-dimensional problems. Function f_{10} is symmetric around the origin, so it has two global minimum and a several local minimas. f_{11}–f_{14} are problems with a several local minimas and a single global minimum point.
Some parameters of the algorithm in this paper are set as follow: the number of particles in the population space is set as *ps*=50; The maximum iteration number is set as 300000 in each running. Each algorithm for each function has been run 30 times. Dimension is set as d=30 for functions f_1-f_9, *d*=2 for functions $f_{10\text{-}11}$ and d=4 for functions $f_{12\text{-}14}$. These dimensions are widely used in literature, so for objective comparison in this study they have been preferred. During the optimization process, the particles are limited to move in the region defined by $[x_{min}, x_{max}]$. An inertia weight of 0.5, cognitive and social parameters of 2 were used. A fully connected topology is used in all cases. Performance was measured as the minimum error $|f(x) - f(x^*)|$ found over the trial, where $f(x^*)$ is the optimum fitness for the problem.

Table 2. Results of the biopsy, FPSA/PSA, risk ratios of online calculator and FES

Equation	Name
$f_1 = \sum_{i=1}^{D} x_i^2$	Sphere / Parabola
$f_2 = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	Schwefel 1.2
$f_3 = \sum_{i=1}^{D-1} \left\{ 100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2 \right\}$	Generalized Rosenbrock
$f_4 = -\sum_{i=1}^{D} x_i \sin\left(\sqrt{x_i}\right)$	Generalized Schwefel 2.6
$f_5 = \sum_{i=1}^{D} \left\{ x_i^2 - 10\cos(2\pi x_i) + 10 \right\}$	Generalized Rastrigin
$f_6 = -20\exp\left\{ -0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2} \right\} - \exp\left\{ \frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i) \right\} + 20 + e$	Ackley
$f_7 = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Generalized Griewank
$f_8 = \frac{\pi}{D}\left\{ 10\sin^2(\pi y_i)\left\{ 1 + 10\sin^2(\pi y_{i+1}) \right\} + \left(y_D - 1\right)^2 \right\} +$ $+ \sum_{i=1}^{D} \mu\left(x_i, 10, 100, 4\right)$ $y_i = 1 + \frac{1}{4}\left(x_i + 1\right)$ $\mu\left(x_i, a, k, m\right) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	Penalized Function P8
$f_9 = 0.1\left\{ \sin^2\left(3\pi x_i\right) + \sum_{i=1}^{D-1} (x_i - 1)^2 \left\{ 1 + \sin^2\left(3\pi x_{i+1}\right) \right\} + \left(x_D - 1\right)^2 \times \left\{ 1 + \sin^2\left(2\pi x_D\right) \right\} \right\} + \sum_{i=1}^{D} \mu(x_i, 5, 100, 4)$	Penalized Function P16
$f_{10} = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	Six-hump Camel-back
$f_{11} = \left\{ 1 + \left(x_1 + x_2 + 1\right)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2\right) \right\} \times \left\{ 30 + \left(2x_1 - 3x_2\right)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2\right) \right\}$	Goldstein-Price
$f_{12} = -\sum_{i=1}^{5} \left\{ \sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i \right\}^{-1}$	Shekel 5
$f_{13} = -\sum_{i=1}^{7} \left\{ \sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i \right\}^{-1}$	Shekel 7
$f_{14} = -\sum_{i=1}^{10} \left\{ \sum_{j=1}^{4} (x_j - a_{ij})^2 + c_i \right\}^{-1}$	Shekel 10

Table 2. The feasible bounds, optimas and dimensions of benchmark function

Equation	Feasible Bounds	Optimum	Dimension
f_1	$(-100,100)^D$	0.0^D	30
f_2	$(-100,100)^D$	0.0^D	30
f_3	$(-30,30)^D$	1.0^D	30
f_4	$(-500,500)^D$	420.9687^D	30
f_5	$(-5.12,5.12)^D$	0.0^D	30
f_6	$(-32,32)^D$	0.0^D	30
f_7	$(-600,600)^D$	0.0^D	30
f_8	$(-50,50)^D$	-1.0^D	30
f_9	$(-50,50)^D$	1.0^D	30
f_{10}	$(-5,5)^D$	$(-0.0898 \quad ,0.7126), (0.0898 \quad ,-0.7126)$	2
f_{11}	$(-2,2)^D$	$(0 \quad ,-1)$	2
f_{12}	$(0,10)^D$	4.0^D	4
f_{13}	$(0,10)^D$	4.0^D	4
f_{14}	$(0,10)^D$	4.0^D	4

Table 3. Mean error after 30 trials

Eq.	PSO	PSO-FS	Const. Gbest	Const. Lbest	PSO-DR M1 Ring	PSO-DR M1 Global	PSO-DR M2	PSO-DR M3
f_1	0	0	0	0	0	0	0	0
f_2	173.5927	0	0	0.1259	0.01	0	3.7E-7	5.14
f_3	36.4739	0	8.1579	12.6648	16.79	0.8	34.57	18.64
f_4	2112	0.0134	3508	3360	2697	3754	2418	1830
f_5	62.7733	0.1990	140.4876	144.8155	44.64	115.51	35.21	9.88
f_6	$1.8608e^{-009}$	$2.1574e^{-011}$	17.6628	17.5891	0.68	18.51	0	0
f_7	0.2269	0.0245	0.0308	0.0009	0	0.008	0	0
f_8	3	0	0.1627	0	0	0.005	0	0
f_9	0,0179	0	0.0040	0	0	0.002	0	0
f_{10}	0	0	0	0	0	0	0	0
f_{11}	0	0	0	0	0	0	0	0
f_{12}	5.1753	1.1789	4.5882	2.5342	0.17	4.34	0	0
f_{13}	2.6457	0.7046	4.4747	1.0630	0	2.55	$8.1e^{-11}$	0
f_{14}	4.7298	0.3574	3.8286	0.5409	0	3.13	$6.6e^{-11}$	0

In many studies researches showed the results on the few functions. Hence we couldn't compare all of our result with a number of others. Table 3 shows the averaged results of 30 independent trials relative to the works Bratton & Blackwell (2008) and Bratton & Kennedy (2007).

5. Conclusions

The PSO algorithm was inspired by the social behaviour of birds and fishes flocking to a food source. There are numerous researches in modifications of original PSO algorithm. In this study the particle swarm optimization algorithm with flexible swarm has been presented. Proposed algorithm was evaluated on fourteen functions, which often used to benchmark the performance of optimization algorithms. PSO-FS algorithm was compared to some other modifications and classical PSO. The results show that PSO-FS always performed one of the better results than the others.

References

Abd-El-Waheda WF., Mousab AA., El-Shorbagy MA (2011).

Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems. Journal of Computational and Applied Mathematics 235:1446–1453.

Akbari R. Ziarati K (2011). A rank based particle swarm optimization algorithm with dynamic adaptation, Journal of Computational and Applied Mathematics, 235(8):2694–2714.

Ali MM, Kaelo P (2008). Improved particle swarm algorithms for global optimization. Applied Mathematics and Computation 196:578–593.

Alrashidi MR., El-Hawary ME (2006). A Survey of Particle Swarm Optimization Applications in Power System Operations, Electric Power Components and Systems, 34/12:1349 — 1357.

Baskar S., Suganthan PN (2004). A Novel Concurrent Particle Swarm Optimization. Proceedings of the Congress on Evolutionary Computation, 792-796.

Blackwell T., Bratton D (2008). Examination of Particle Tails, Journal of Artificial Evolution and Applications, 8:1-10.

Bratton D., Kennedy J (2007). Defining a Standard for Particle Swarm Optimization, Proceedings of the 2007 IEEE Swarm Intelligence Symposium.

Bratton D. and Blackwell T (2008). A Simplified Recombinant PSO. Journal of Artificial Evolution and Applications, 8:1-10.

Chen CC (2011). Two-layer particle swarm optimization for unconstrained optimization problems. Applied Soft Computing, 11(1): 295-304

Chen TY., Chi TM (2010). On the improvements of the particle swarm optimization algorithm. Advances in Engineering Software 41:229–239.

He S., Wu QH, Wen JY, Saunders JR, Paton RC (2004). A particle swarm optimizer with passive congregation. BioSystems 78:135–147.

Jiang Y., Hu T., Huang CC, Wu X (2007). An improved particle swarm optimization algorithm. Applied Mathematics and Computation 193:231–239.

Kang Q., Wang L., Wu Q (2008). A novel ecological particle swarm optimization algorithm and its population dynamics analysis. Applied Mathematics and Computation 205:61–72.

Kennedy J., Eberhart R. (1995). Particle Swarm Optimization, IEEE International Conference on Neural Networks.

Kok S., Snyman JA (2008). A Strongly Interacting Dynamic Particle Swarm Optimization Method. Journal of Artificial Evolution and Applications. 28:1-9.

Marinakis Y., Marinaki M., Dounias G (2008). Particle swarm optimization for pap-smear diagnosis, Expert Systems with Applications, 35:1645–1656.

Pena J., Upegui A., Sanchez E (2006). Particle Swarm Optimization with Discrete Recombination: An Online Optimizer for Evolvable Hardware, Proceedings of the First NASA/ESA Conference on Adaptive Hardware and Systems.

Van den Bergh F., Engelbrecht AP. (2004). A cooperative approach to particle swarm optimization, IEEE Trans Evolut Comput, 8(3) 225–39.

Wang Z., Sun X., Zhang. D (2007). A PSO-Based Classification Rule Mining Algorithm, ICIC 2007, LNAI 4682: 377–384.

Zhao Y., Zu W., Zeng H (2009). A modified particle swarm optimization via particle visual modelling analysis, Computers and Mathematics with Applications, 57(11-12):2022-2029.