www.s2is.org

International Journal on Smart Sensing and Intelligent Systems

ISSN 1178-5608

A NEW KIND OF PSO: PREDATOR PARTICLE SWARM OPTIMIZATION

[1]Mehdi Neshat, [2]Mehdi Sargolzaei, [3]Azra Masoumi, [4]Adel Najaran

[1]Department of computer science, Shirvan branch, Islamic Azad University, Shirvan, Iran

Emails: neshat_mehdi@ieee.org

[2]Department of software engineering, Shirvan branch, Islamic Azad University, Shirvan, Iran

Emails: sargolzae@iau-shirvan.ac.ir

[3]Department of hardware engineering, Shirvan branch, Islamic Azad University, Shirvan, Iran

Emails: masoumi@iau-shirvan.ac.ir

[4]Department of computer science, Shirvan branch, Islamic Azad University, Shirvan, Iran

Emails: najaran@iau-shirvan.ac.ir

Submitted: Mar. 17, 2012 ***Accepted: May 11, 2012*** ***Published: June 1, 2012***

Abstract- Today, swarm intelligence is widely used in optimization problems. PSO is one the best swarm intelligence methods. In the method, each particle moves toward the direction in which the best individual and group experience has happened. The most important disadvantage of this method is that it falls in local optima. To fix the problem, a metaheuristic method is proposed in this paper. There has always been a competition between prey and predator in the nature. Little birds often fly in a colony form to run away from birds of prey. Being inspired by the phenomenon, a new particle is added to PSO algorithm known as predator, also a new behavior called "Take flight from predator" is defined.

This particle is responsible for attacking the colony of particles so as to prevent the premature convergence. With the predator attack to the colony, particles run away and again the chance rises for a Global optimum to be gained. The attack just caused particles dispersion and no particle dies. It can be repeated for m times and the optimal point is saved each time. To test the method, 12 benchmark functions were employed and the results were compared to OPSO, VPSO, LPSO, and GPSO methods. Regarding the results, the proposed method had a better performance.

Index terms*:* Predator; particle swarm optimization; local optimum; premature convergence.

I. INTRODUCTION

All science is inspired from nature, especially the swarm intelligence. Studying social behaviors of entities is very interesting and useful; ants, birds, fish, and various animal species living socially or in colonies. The relationship between members, migration, searching for food, and running away from hunters are the main behaviors of every colony. Among the behaviors, birds' have been drawn further attention, because most birds live in huge communities and have complicated behaviors. Running away from the predator is one of these behaviors addressed in this paper. Competition between birds and predator is inevitable and the colony must always be prepared for birds of prey to attack. Presence of a bird of prey results in a better awareness and cooperation within the group members.

Particle swarm optimization (PSO), which was introduced by Kennedy and Eberhart in 1995 [1], [2], is one of the most important swarm intelligence paradigms [3]. The PSO uses a simple mechanism that mimics swarm behavior in birds flocking and fish schooling to guide the particles to search for globally optimal solutions. As PSO is easy to implement, it has rapidly progressed in recent years and with many successful applications seen in solving real-world optimization problems [4][10].

However, similar to other evolutionary computation algorithms, The PSO is also a population-based iterative algorithm. Hence, the algorithm can computationally be inefficient as measured by the number of function evaluations (FEs) required [11]. Further, the standard PSO algorithm can easily get trapped in the local optima when solving complex multimodal problems [10]. These weaknesses have restricted wider applications of the PSO [5].

Therefore, accelerating convergence speed and avoiding the local optima have become the two most important and appealing goals in PSO research. A number of variant PSO algorithms have, hence, been proposed to achieve these two goals [6], [7], [9], [10]. In this development, control of algorithm parameters and combination with auxiliary search operators have become two of the three most salient and promising approaches (the other being improving the topological structure) [8]. However, so far, it is seen to be difficult to simultaneously achieve both goals. For example, the comprehensive-learning PSO (CLPSO) in [10] focuses on avoiding the local optima, but brings in a slower convergence as a result.

To achieve both goals, adaptive PSO (APSO) is formulated by developing a systematic parameter adaptation scheme and an elitist learning strategy (ELS) [23]. To enable adaptation, an evolutionary state estimation (ESE) technique is first devised. Hence, adaptive parameter control strategies can be developed based on the identified evolutionary state and by making use of existing research results on inertia weight [11]–[12] and acceleration coefficients [13]–[14].

To avoid possible local optima in the convergence state ,combinations with auxiliary techniques have been developed elsewhere by introducing operators such as selection [15],crossover [16], mutation [17], local search [18], reset [19], [20], reinitialization [21], [22], etc., into PSO. These hybrid operations are usually implemented in every generation [15]–[17] or at a prefixed interval [18] or are controlled by adaptive strategies using stagnated generations as a trigger [19]–[22]. While these methods have brought improvements in PSO, the performance may further be enhanced if the auxiliary operations are adaptively performed with a systematic treatment according to the evolutionary state. For example, the mutation, reset, and reinitialization operations can be more pertinent when the algorithm has converged to a local optimum rather than when it is exploring.

In Section II, the PSO and its developments are briefly reviewed. Section III presents the Predator Particle Swarm Optimization (PPSO) approach in detail. Section IV experimentally compares the PPSO with various existing PSO algorithms using a set of benchmark functions. Finally, conclusions are drawn in Section V.

II. PSO AND ITS DEVELOPMENTS

The fundament to the development of PSO is a hypothesis [24] that social sharing of information among conspeciates offers an evolutionary advantage. PSO is similar to the other evolutionary algorithms in that the system is initialized with a population of random solutions. However, each potential solution is also assigned a randomized velocity, and the potential solutions, call *particles*, corresponding to individuals. Each particle in PSO flies in the D-dimensional problem space with a velocity which is dynamically adjusted according to the flying experiences of its own and its colleagues. The location of the *i*th particle is represented as $X_i = x_{i_1}, ..., x_{i_d}, ..., x_{i_D})$, where $x_{i_d} \in {}'_d, u_d], d \in \lfloor, D], l_d, u_d$ are the lower and upper bounds for the *d*th dimension, respectively. The best previous position (which giving the best fitness value) of the *i*th particle is recorded and represented as $P_i = p_{i_1}, ..., p_{i_d}, ..., p_{i_D})$ which is also called pbest. The index of the best particle among all the particles in the population is represented by the symbol *g*. The location *Pg* is also called *gbest*. The velocity for the *i*th particle is represented as $V_i = v_{i_1}, ..., v_{i_d}, ..., v_{i_D})$, is clamped to a maximum velocity $V_{\max} = v_{\max_1}, ..., v_{\max_d}, ..., v_{\max_D})$, which is specified by the user. The particle swarm optimization concept consists of, at each time step, changing the velocity and location of each particle toward its *pbest* and *gbest* locations according to the equations (1a) and (1b), respectively:

$$v_{i_d} = v * v_{i_d} + {}'_1 * rand() * (p_{i_d} - {};_{i_d}) + {}'_2 * rand() * (p_{g_d} - {};_{i_d}) \qquad (1$$

$$x_{i_d} = {}^{\varsigma}_{i_d} + {}'_{i_d} \qquad (2$$

Where *w* is inertia weight, *c*1 and *c*2 are acceleration constants and *rand* () is a random function in the range [0, 1]. For equation (1), the first part represents the inertia of pervious velocity; the second part is the "cognition" part, which represents the private thinking by itself; the third part is the "social" part, which represents the cooperation among the particles . If the sum of accelerations would cause the velocity v_{i_d} on that dimension to exceed $v_{\max_d}$, then is v_{i_d} limited to $v_{\max_d}$. $V_{\max}$ determines the resolution with which regions between the present position and the target position are searched .

The process for implementing PSO is as follows:

a).Initialize a population (array) which including *m* particles, For the *i*th particle, it has random location *Xi* in the problem space and for the *d*th dimension of velocity v_{i_d}, $v_{i_d} = Rand2() * v_{\max_d}$, where *Rand*2() is in the range [-1, 1];

b). Evaluate the desired optimization fitness function for each particle;

c).Compare the evaluated fitness value of each particle with its *pbest*. If current value is better than *pbest*, then set the current location as the *pbest* location. Furthermore, if current value is better than *gbest*, then reset *gbest* to the current index in particle array;

d). Change the velocity and location of the particle according to the equations (1) and (2), respectively;

e).Loop to step b) until a stop criterion is met, usually a sufficiently good fitness value or a predefined maximum number of generations *Gmax*.

The parameters of standard PSO includes: number of particles *m*, inertia weight *w*, acceleration constants *c*1 and *c*2, maximum velocity $v_{\max_d}$.

Given its simple concept and effectiveness, the PSO has become a popular optimizer and has widely been applied in practical problem solving. Thus, theoretical studies and performance improvements of the algorithm have become important and attractive. Convergence analysis and stability studies have been reported by Clerc and Kennedy [26], Trelea [27], Yasuda *et al.* [28], Kadirkamanathan *et al.* [29], and van den Bergh and Engelbrecht [30].

The inertia weight *w* in (1) was introduced by Shi and Eberhart [25]. They proposed a *w* linearly decreasing with the iterative generations as:

$$w = v_{\max} - w_{\max} - v_{\min})g/G \qquad (3)$$

Where *g* is the generation index representing the current number of evolutionary generations, and *G* is a predefined maximum number of generations. Here, the maximal and minimal weights $w_{\max}$ and $w_{\min}$ are usually set to 0.9 and 0.4, respectively [25], [31]. In addition, a fuzzy adaptive w was proposed in [32], and a random version setting *w* to 0.5 + *random* (0*,* 1)*/*2 was experimented in [33] for dynamic system optimization. As this random *w* has an expectation of 0.75, it has a similar idea as Clerc's constriction factor [34], [35]. The constriction factor has been introduced into PSO for analyzing the convergence behavior, i.e., by modifying (1) to

$$v_i^d = \chi\ v_i^d + {}_1 rand_1^d (pBest_i^d - {}_i^d) + {}_2 rand_2^d (nBest^d - {}_i^d)] \qquad (4)$$

Where the constriction factor

$$\chi = \frac{2}{\left|2 -) - \sqrt{\varphi^2 - \varphi}\right|} \qquad (5)$$

is set to 0.729 with

$$\varphi = c_1 + c_2 = .1 \qquad (6)$$

Where $c1$ and $c2$ are both set to 2.05 [35] mathematically, the constriction factor is equivalent to the inertia weight, as Eberhart and Shi pointed out in [36]. The PSO with an inertia weight and use a global version of PSO (GPSO) [25] to denote the traditional global-version PSO with an inertia weight as given by (3). In addition to the inertia weight and the constriction factor, the acceleration coefficients $c1$ and $c2$ are also important parameters in PSO. In Kennedy's two extreme cases [37], i.e., the "social-only" model and the "cognitive-only" model, experiments have shown that both acceleration coefficients are essential to the success of PSO. Kennedy and Eberhart suggested a fixed value of 2.0, and this configuration has been adopted by many other researchers. Suganthan [38] showed that using ad hoc values of $c1$ and $c2$ rather than a fixed value of 2.0 for different problems could yield better performance. Ratnaweera *et al.* [39] proposed a PSO algorithm with linearly time-varying acceleration coefficients (HPSO-TVAC), where a larger $c1$ and a smaller $c2$ were set at the beginning and were gradually reversed during the search. Among these three methods, the HPSO-TVAC shows the best overall performance [39]. This may be owing to the time-varying $c1$ and $c2$ that can balance the global and local search abilities, which implies that adaptation of $c1$ and $c2$ can be promising in enhancing the PSO performance. Hence, this paper will further investigate the effects of $c1$ and $c2$ and develop an optimal adaptation strategy according to ESE.

Another active research trend in PSO is hybrid PSO, which combines PSO with other evolutionary paradigms. Angeline [40] first introduced into PSO a selection operation similar to that in a genetic algorithm (GA). Hybridization of GA and PSO has been used in [41] for recurrent artificial neural network design. In addition to the normal GA operators, e.g., selection [40], crossover [42], and mutation [43], other techniques such as local search [44] and differential evolution [45] have been used to combine with PSO. Cooperative approach [46], self organizing hierarchical technique [47], deflection, stretching, and repulsion techniques [48] have also been hybridized with traditional PSO to enhance performance. Inspired by biology, some researchers introduced niche [49], [50] and speciation [51] techniques into PSO to prevent the swarm from crowding too closely and to locate as many optimal solutions as possible and adaptive particle swarm optimization (APSO) that features better search efficiency than classical particle swarm optimization [58].

In addition to research on parameter control and auxiliary techniques, PSO topological structures are also widely studied. The LPSO with a ring topological structure and the von Neumann topological structure PSO (VPSO) have been proposed by Kennedy and Mendes [52], [53] to enhance the performance in solving multimodal problems. Further, dynamically changing neighborhood structures have been proposed by Suganthan [38], Hu and Eberhart [54], and Liang and Suganthan [55] to avoid the deficiencies of fixed neighborhoods. Moreover, in the "fully informed particle swarm" (FIPS) algorithm [56], the information of the entire neighborhood is used to guide the particles. The CLPSO in [57] lets the particle use different *pBest*'s to update its flying on different dimensions for improved performance in multimodal applications.

III. PREDATOR PARTICLE SWARM OPTIMIZATION (PPSO)

In PSO algorithm, particles move based on the resultant of three vectors: the first vector, toward the global best position; second, toward the best past personal experience; and third, toward the previous particle path. Next position will be determined based on the three vectors. Falling in local optima points is one of the main disadvantages of PSO method, namely, if a particle is entrapped by local optima, other particles will be converged to that particle and a premature convergence will occur. PPSO algorithm was developed to fix the problem. It is inspired by PSO algorithm with partial modifications.

a. Predator Particle (PP)

There is a new particle in the algorithm called "predator". It behaves differently comparing to other particles and never seeks the environment for optimum points. It is a predator and intends to attack other particles per se. Indeed, the time of attack is predetermined and occurs when the colony is totally converged. Despite the nature, our predator no longer kills particles but aims at scaring them to save them from local optima trap based on their instinct to run away from danger. The particle prevents premature convergence to happen. When the convergence occurs, PP particle is located at a random position of an environment and attacks the global best.

```
a.i..Algorithm Predator Particle
If ( $\sum_{i=1}^{n} gbest_i - x_i(t)| \le \varkappa$ ) then {
 a . Random Create PP
 b . PP attack to global best particle
```

$V_{PP}(t+\ )=\delta * V_{PP}(t)+\ rand*(gbest_{i}-X_{PP})$

$X_{PP}(t+\ )=X_{PP}(t)+V_{PP}(t+\ )$

}

In PPSO algorithm, if sum of the difference between all particles and global best is less than or equal to , PP particle performs the attack behavior. Otherwise, particles keep on their natural behaviors.   is a changing value varying based on each function. The value of   depends on number of the particles in the environment and dimensions of the environment. V_{PP} velocity vector must be calculated, When the attack behavior gets started. There is a constant index   in velocity vector formula with a value between (1-2).

After PP particle attacked to colony and as soon as it approaches the particles, they show a new behavior called "Take Flight" based on their runaway-from-predator instinct. Figure 1 illustrates the estate of particles runaway behavior.

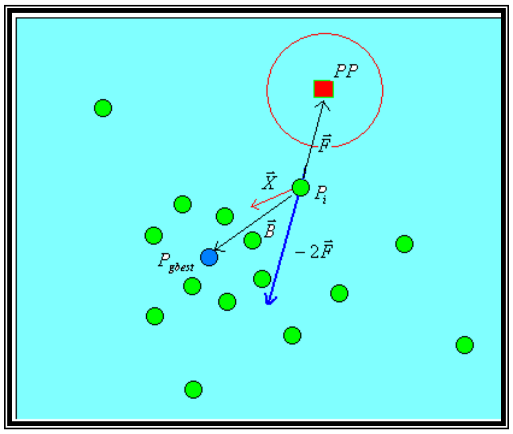


Figure 1. Particles Take Flight behavior of predator particle

a.ii. Take Flight behavior

Upon observing PP particle, each of the other particles performs the runaway behavior. It is out of instinct and entities practice it without previous thought and planning. They only intend to run away from the predator and to go as far as possible. However, in the proposed method, each particle tries to move far away from the predator based on the resultant of three vectors: $\vec{X},\vec{B},-\ \vec{F}$. The vectors are described in the following section.

$$\begin{aligned}
&\vec{X} = x_i(t+\ ) \\
&\vec{B} = rand * (gbest_i - x_i(t)) \\
&\vec{F} = rand * (X_{PP} - x_i(t)) \\
&x_i(t+\ ) = \vec{X} + \vec{B} - 2\vec{F}
\end{aligned} \tag{7}$$

Vector $\vec{X}$ is the particle's previous path, vector $\vec{B}$ is the difference between particle position and global best multiplied by a random number. Rand function generates a random number between 0 and 1 and gives the particle a movement freedom. Vector $\vec{F}$ is the difference between particle position and the predator. Runaway factor of the predator is employed in equation (7) with a factor of 2 indicating the significance and effect of the vector in the next move. Runaway factor has a twice magnitude of $\vec{F}$ vector but in the reverse direction. Figure (2) shows PPSO algorithm flowchart.

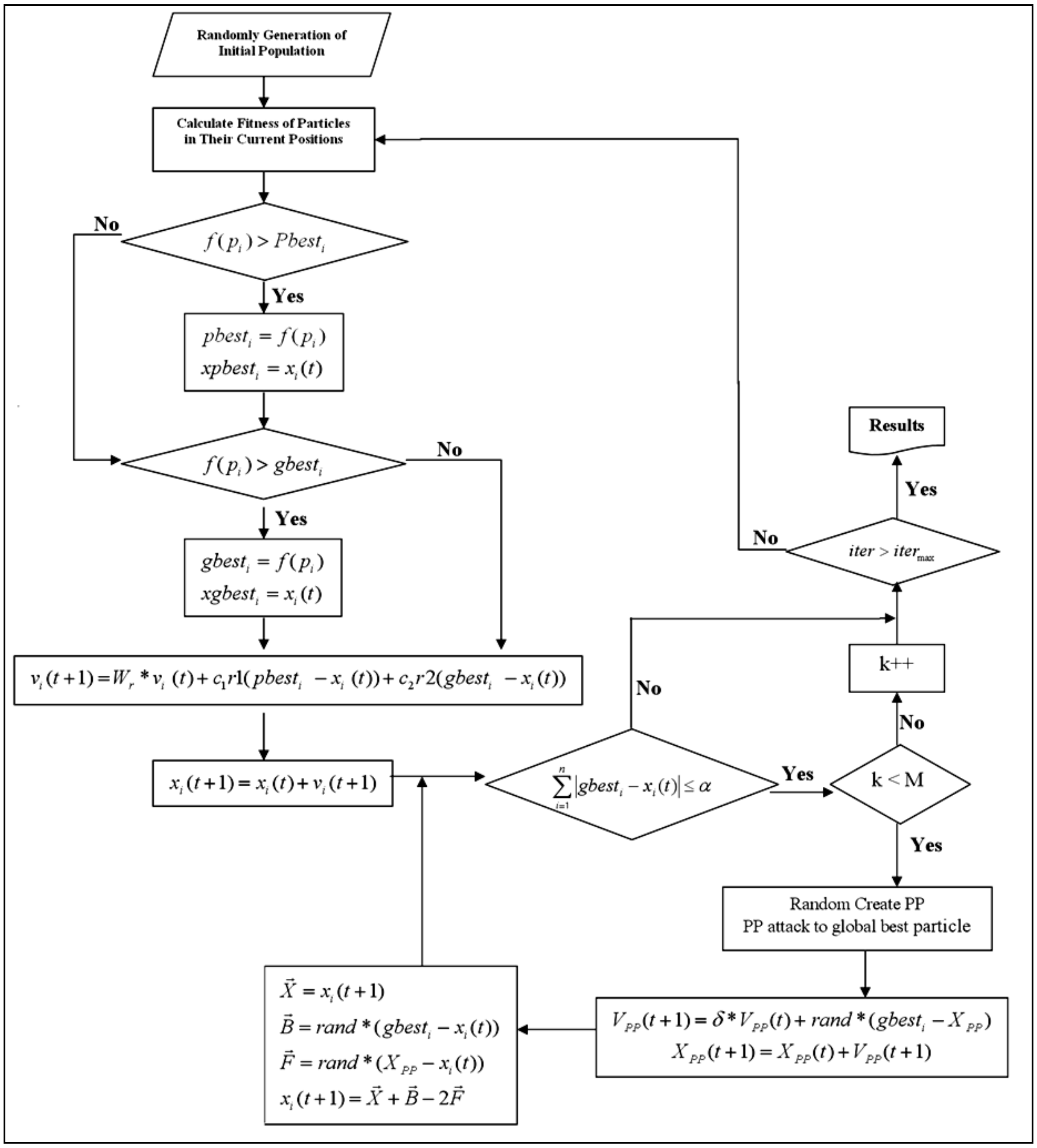

Randomly Generation of Initial Population
Calculate Fitness of Particles in Their Current Positions
$f(p_i) > Pbest_i$
No
Yes
$pbest_i = f(p_i)$
$xpbest_i = x_i(t)$
$f(p_i) > gbest_i$
No
Yes
$gbest_i = f(p_i)$
$xgbest_i = x_i(t)$
$v_i(t+1) = W_r * v_i(t) + c_1 r1(pbest_i - x_i(t)) + c_2 r2(gbest_i - x_i(t))$
$x_i(t+1) = x_i(t) + v_i(t+1)$
$\sum_{i=1}^{n} |gbest_i - x_i(t)| \le \alpha$
No
Yes
$k < M$
No
Yes
k++
$iter > iter_{max}$
No
Yes
Results
Random Create PP
PP attack to global best particle
$V_{PP}(t+1) = \delta * V_{PP}(t) + rand * (gbest_i - X_{PP})$
$X_{PP}(t+1) = X_{PP}(t) + V_{PP}(t+1)$
$\vec{X} = x_i(t+1)$
$\vec{B} = rand * (gbest_i - x_i(t))$
$\vec{F} = rand * (X_{PP} - x_i(t))$
$x_i(t+1) = \vec{X} + \vec{B} - 2\vec{F}$

Figure 2. PPSO algorithm flowchart

IV. EXPERIMENT RESULTS

12 benchmark functions are applied to test the proposed method. Since the functions are widely used and a variety of methods are tested using them, they can be considered as suitable measurement criteria. Table (1) represents benchmark functions and their properties.

Table 1. represents benchmark functions and their properties.

#	Function	Equation	Domain	F_{min}	
f_1	Sphere	$\sum_{i=1}^{D} x_i^2$	$\pm$ 5.12	0	
f_2	Step	$\sum_{i=}^{D} \lfloor x_i + .5 \rfloor^2$	$\pm$ 00	0	
f_3	Rosenbrock	$\sum_{i=}^{D-} (100(x_{i+} - x_i^2)^2 + x_i -)^2)$	$\pm$ 50	0	**Unimodal**
f_4	Schwefel's P2.22	$\sum_{i=}^{D} x_i \mid + \prod_{i=}^{D} x_i \mid$	$\pm$ 0	0	
f_5	Quadric Noise	$\sum_{=} x_i^4 + and[0,1)$	$\pm$ .28	0	
f_6	Ackley	$20 + - 0e^{- .2\sqrt{\frac{1}{D}\sum_{i=}}} - e^{\frac{1}{D}\cos(2\pi)}$	$\pm$ 32	0	
f_7	Griewank	$\sum_{i=}^{D} (\frac{x_i^2}{4000}) - \prod_{i=}^{D} os(\frac{x_i}{\sqrt{i}}) +$	$\pm$ 600	0	
f_8	Rastrigin	$\sum_{i=}^{D} x_i^2 - 0\cos(2\pi _i) + 0)$	$\pm$ 5.12	0	
f_9	Schwefel	$\sum_{i=}^{D} - x\sin(\sqrt{x_i})$	$\pm$ 00	- 12569.5	
f_{10}	Noncontinuous Rastrigin	$\sum_{i=1}^{D}(y_i^2 - 10\cos(2\pi _i) + 10)$ where $y_i = \begin{cases} x_i & \mid x_i \mid < 0.5 \\ \frac{round(2x_i)}{2} & \mid x_i \mid \geq 0.5 \end{cases}$	$\pm$ 5.12	0	
f_{11}	Perm #1	$\sum_{x=} \left[\sum_{i=} i^k + \beta (x_i / i)^k -) \right]$	$\pm$ 4 β = 0	0	**Multimodal**
f_{12}	Generalized Penalized	$\pi D\{10\sin^2(\pi _i) + \sum_{i=}^{D-} (y_i -)^2[1 + 0\sin^2(\pi _{i+})]$ $+ y_D -)^2\} + \sum_{i=}^{D} u(x_i,10,100,4)$ Where $y_i = 1 + \frac{1}{4}(x_i + 1)$, $u(x_i,a,k,m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \leq x_i \leq a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$\pm$ 50	0	

To test the proposed method, two series of benchmark functions are used – Unimodal function and multimodal function to determine its performance with both types. The 12 benchmark functions are displayed in Figure (3).

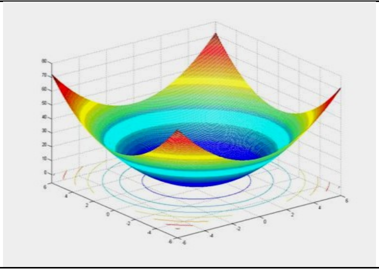
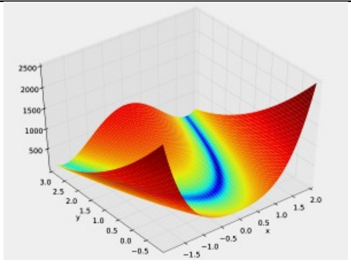
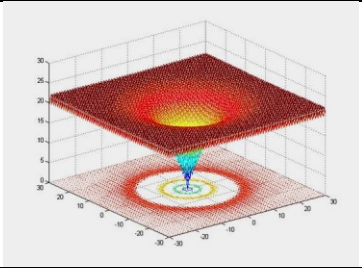
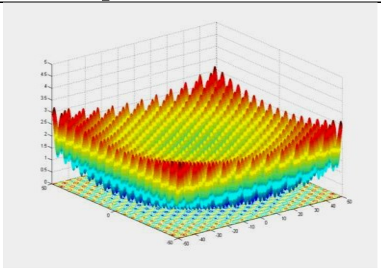
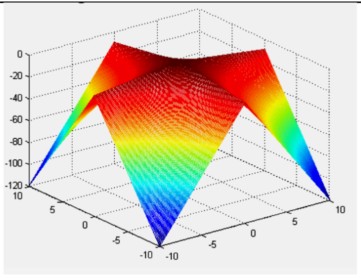
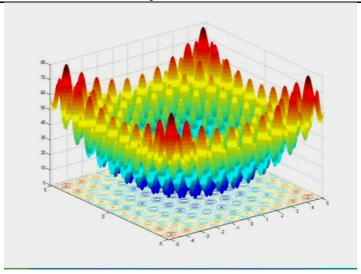
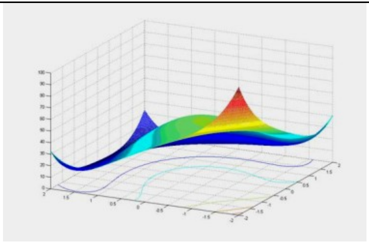
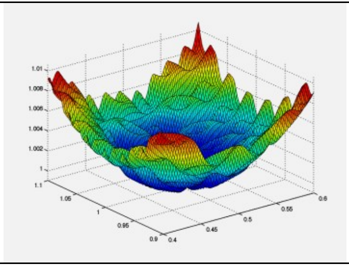
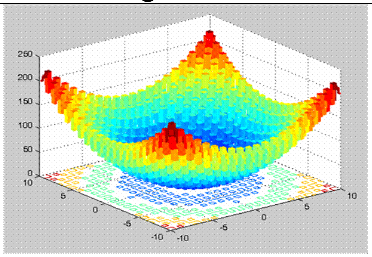
		
Sphere Function	Rosenbrock Function	Ackley Function
		
Griewank Function	Schwefel 2.22	Rastrigin Function
		
Perm Function	Quadric Noise	Non continuous Rastrigin

Figure 3. The 12 benchmark functions are used in this study

Other four PSO methods were employed to compare the proposed method results. Each of the four has its own specific characteristics and all are among suitable PSO methods, there are also other modern methods, however. Whole features of the four PSO methods and their parameters are listed in Table (2).

Table 2. Exhibits different methods of PSO.

Algorithm	Year	Topology	Parameters setting	Reference
GPSO	1998	Global star	$w:0.9-$).4$,c_1,c_2 =$!	[61]
LPSO	2002	Local ring	$w:0.9-$).4$,c_1,c_2 =$!	[60]
VPSO	2002	Local von Neumann	$w:0.9-$).4$,c_1,c_2 =$!	[59]
OPSO	2008	Orthogonal particle swarm	$w:0.9-$).4$,c_1 =$ $'_2 =$).2$,V_{\max} =$).5$*rang$	[62]

Each of the above mentioned methods is tested on the six benchmark functions and then the results are compared to the proposed method. The results of the 4 PSO methods and the proposed method comparison are listed in Table (3). The number of PP particle attacks is 10 times in the algorithm suggested.

Table.3. Comparison between PPSO and optimization four methods for 30-dimension problems

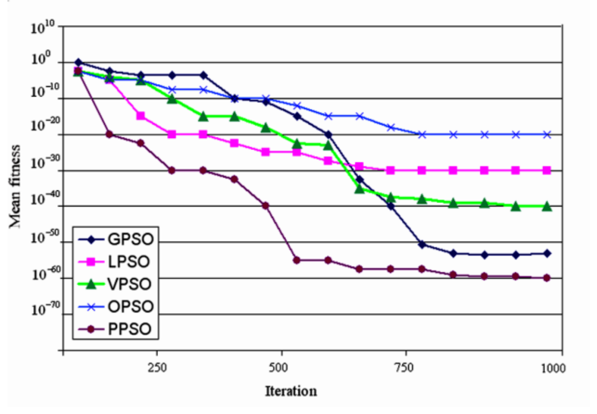
#	Sphere	Rosenbrock	Quadric Noise	Schwefel's P2.22	step	Griewank
GPSO	1.98e-053	28.1	6.45e-002	2.51e-034	0	2.37e-002
LPSO	4.77e-029	21.8627	18.6	2.03e-020	0	1.10e-002
VPSO	5.11e-038	37.6469	1.44	6.29e-027	0	1.31e-002
OPSO	6.45e-018	49.61	2.44e-002	1.26e-010	0	2.29e-003
PPSO	3.45e-066	19.72	3.54e-002	3.49e-033	0	2.49e-005
#	Schwefel	Noncontinuous Rastrigin	Ackley	Rastrigin	Generalized Penalized	Perm #1
GPSO	-10090.16	15.5	1.15e-014	6.97	1.04e-002	1.02e-001
LPSO	-9628.35	30.4	1.85e-014	8.68	2.18e-030	1.41e-002
VPSO	-9845.27	21.33	1.4e-014	7.25	3.46e-003	12.5
OPSO	-8402.53	2.49e-006	6.23e-009	8.07	1.56e-019	2.33e-002
PPSO	-110451	3.08e-007	5.12e-016	2.94	2.91e-021	2.84e-003

As seen in Table (4), the proposed method gains the best result with all functions except schwefelsP2.22. GPSO method gains the best result with the schwefelsP2.22 function and the proposed method is ranked the second with a little difference. There have been better results with the proposed method comparing to four PSO methods. Results in Table (4) are the average of 10 repetitions of each algorithm on the benchmark function.

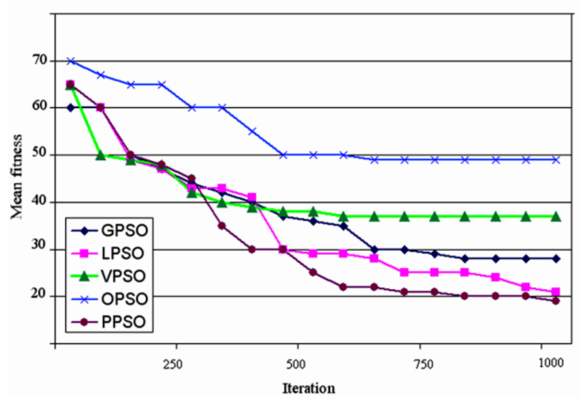
As seen in Table (4), the proposed method has not good results in the $f_{4,}$ f_5 and f_{12} . According to Tables 3 and 4 whenever the function's dimension was increased, then computational complexity become more. PPSO is reached to best results in other functions.

Table 4. Comparison between PPSO and optimization four methods for 10-dimension problems

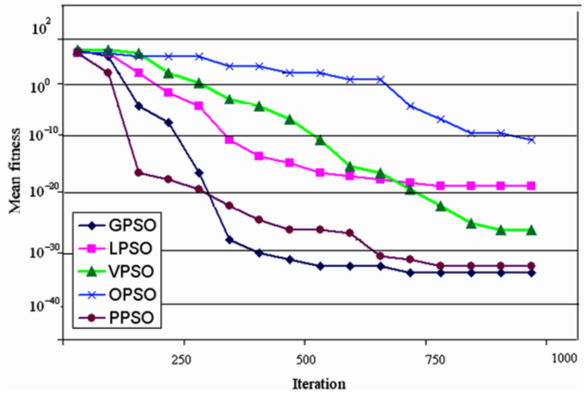
#	Sphere	Rosenbrock	Quadric Noise	Schwefel's P2.22	step	Griewank
GPSO	2.28e-063	9.12	3.74e-003	4.21e-040	0	1.07e-004
LPSO	1.21e-038	1.472	1.61e-001	1.62e-028	0	8.22e-004
VPSO	4.38e-049	17.224	1.44e-001	5.48e-036	0	4.91e-004
OPSO	9.04e-028	29.79	2.44e-004	1.36e-017	0	2.69e-006
PPSO	2.66e-076	0.49	3.54e-004	7.41e-039	0	1.47e-008
#	Schwefel	Noncontinuous Rastrigin	Ackley	Rastrigin	Generalized Penalized	Perm #1
GPSO	-11090.16	1.712	4.05e-018	1.48	1.84e-004	2.74e-004
LPSO	-10618.25	3.449	1.22e-017	2.59	1.74e-035	1.83e-006
VPSO	-10775.87	1.624	3.18e-017	1.38	2.13e-007	1.5
OPSO	-9002.83	1.79e-007	5.25e-011	2.07	1.07e-025	1.33e-004
PPSO	-12569.5	2.18e-009	4.73e-020	1.72e-001	1.81e-028	1.84e-007

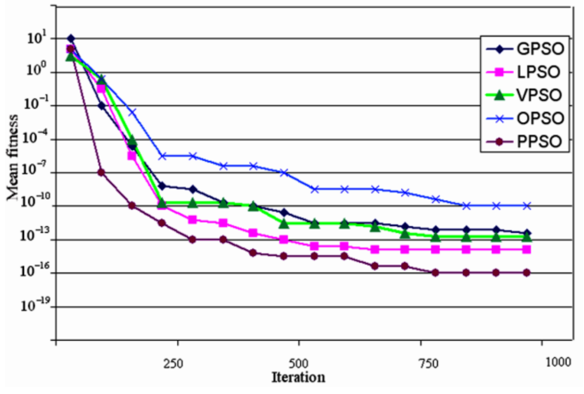


(a)



(b)





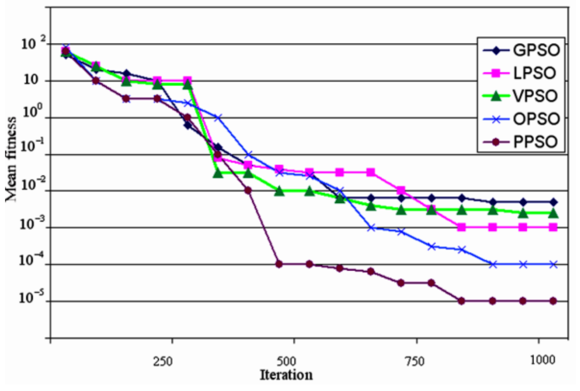
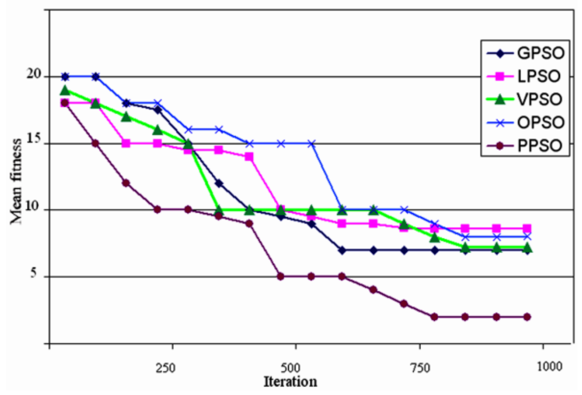
(c)	(d)
	
(e)	(f)

Figure 4)Proposed method in comparison with other methods, functions :a) Sphere ;b) Rosenbrock ; c) Schwefel's P2.22 ; d) Ackley ; e) Griewank ; f) Rastrigin

The proposed method has the best result as well as a desirable convergence rate with sphere function. GPSO and VPSO methods have also gained good results. In Rosenbrock function, VPSO has the best performance, when there are 100 iterations, but in the next iterations, PPSO has performed the best. LPSO also gains partially good result. In schwefelsP2.22 function, PPSO performs well at iterations less than 250 with a considerable convergence rate, but in the next iterations, PPSO has gained better results. Besides, in other functions, the proposed method performs better than the other methods.

V. Conclusion

PSO is one the best swarm intelligence methods. In the method, each particle moves toward the direction in which the best individual and group experience has happened. The most important disadvantage of this method is that it falls in local optima. To fix the problem, a metaheuristic method is proposed in this paper. There has always been a competition between prey and predator in the nature. Little birds often fly in a colony form to run away from birds of prey. Being inspired by the phenomenon, a new particle is added to PSO algorithm known as *predator*, also a new behavior called "*Take flight from predator*" is defined. This particle is responsible for attacking the colony of particles so as to prevent the premature convergence. With the predator attack to the colony, particles run away and again the chance rises for a Global optimum to be gained. The attack just caused particles dispersion and no particle dies. It can be repeated for *m* times and the optimal point is saved each time. To test the method, 12 benchmark functions were employed and the results were compared to OPSO, VPSO, LPSO, and GPSO methods. Regarding the results, the proposed method had a better performance.

REFERENCES

[1] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, Australia, 1995, vol. 4, pp. 1942–1948.
[2] R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in *Proc. 6th Int. Symp. Micromachine Human Sci.*, Nagoya,Japan, 1995, pp. 39–43.
[3] J. Kennedy, R. C. Eberhart, and Y. H. Shi, *Swarm Intelligence*. San Mateo, CA: Morgan Kaufmann, 2001.
[4] R. C. Eberhart and Y. H. Shi, “Particle swarm optimization: Developments, applications and resources,” in *Proc. IEEE Congr. Evol. Comput.*,Seoul, Korea, 2001, pp. 81–86.
[5] X. D. Li and A. P. Engelbrecht, “Particle swarm optimization: An introduction and its recent developments,” in *Proc. Genetic Evol. Comput.Conf.*, 2007, pp. 3391–3414.
[6] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, “OPSO: Orthogonal particle swarm optimization and its application to task assignment problems,” *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 2, pp. 288–298, Mar. 2008.
[7] B. Liu, L. Wang, and Y. H. Jin, “An effective PSO-based memetic algorithm for flow shop scheduling,” *IEEE Trans. Syst., Man, Cybern. B,Cybern.*, vol. 37, no. 1, pp. 18–27, Feb. 2007.
[8] R. C. Eberhart and Y. Shi, “Guest editorial,” *IEEE Trans. Evol. Comput.—Special Issue Particle Swarm Optimization*, vol. 8, no. 3,pp. 201–203, Jun. 2004.
[9] G. Ciuprina, D. Ioan, and I. Munteanu, “Use of intelligent-particle swarm optimization in electromagnetics,” *IEEE Trans. Magn.*, vol. 38, no. 2,pp. 1037–1040, Mar. 2002.
[10] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295,Jun. 2006.
[11] Y. Shi and R. C. Eberhart, “A modified particle swarm optimizer,” in *Proc. IEEE World Congr. Comput. Intell.*, 1998, pp. 69–73.
[12] Y. Shi and R. C. Eberhart, “Fuzzy adaptive particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, 2001, vol. 1, pp. 101–106.
[13] A. Ratnaweera, S. Halgamuge, and H. Watson, “Particle swarm optimization with self-adaptive acceleration coefficients,” in *Proc. 1st Int. Conf.Fuzzy Syst. Knowl. Discovery*, 2003, pp. 264–268.
[14] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, “Adaptive multi-objective particle swarm optimization algorithm,” in *Proc. IEEE Congr. Evol.Comput.*, Singapore, 2007, pp. 2281–2288.
[15] P. J. Angeline, “Using selection to improve particle swarm optimization,”in *Proc. IEEE Congr. Evol. Comput.*, Anchorage, AK, 1998, pp. 84–89.
[16] Y. P. Chen,W. C. Peng, andM. C. Jian, “Particle swarm optimization with recombination and dynamic linkage discovery,” *IEEE Trans. Syst., Man,Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1460–1470, Dec. 2007.
[17] P. S. Andrews, “An investigation into mutation operators for particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, Vancouver,BC, Canada, 2006, pp. 1044–1051.

[18] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer with local search," in *Proc. IEEE Congr. Evol. Comput.*, 2005,pp. 522–528.
[19] A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in *Proc. Int. Conf. Artif. Intell.*, Las Vegas, NV,2000, pp. 429–434.
[20] X. Hu and R. C. Eberhart, "Adaptive particle swarm optimization: Detection and response to dynamic systems," in *Proc. IEEE Congr. Evol.Comput.*, Honolulu, HI, 2002, pp. 1666–1670.
[21] X. Xie, W. Zhang, and Z. Yang, "Adaptive particle swarm optimization on individual level," in *Proc. Int. Conf. Signal Process.*, 2002,pp. 1215–1218.
[22] M. Clerc, "The swarm and the queen: Toward a deterministic and adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 1999,pp. 1951–1957.
[23] Zhi-Hui Zhan, Jun Zhang, Yun Li,Henry Shu-Hung Chung, "Adaptive Particle Swarm Optimization", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, 39 Issue:6 , 1362 – 1381,2009.
[24] E. O. Wilson, *Sociobiology: the new synthesis*, Belknap Press, Cambridge, MA, 1975.
[25] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, 1998, pp. 69–73.
[26] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.
[27] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, Mar. 2003.
[28] K. Yasuda, A. Ide, and N. Iwasaki, "Stability analysis of particle swarm optimization," in *Proc. 5th Metaheuristics Int. Conf.*, 2003, pp. 341–346.
[29] V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 245–255, Jun. 2006.
[30] F. van den Bergh and A. P. Engelbrecht, "A study of particle optimization particle trajectories," *Inf. Sci.*, vol. 176, no. 8, pp. 937–971, Apr. 2006.
[31] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 1999, pp. 1945–1950.
[32] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2001, vol. 1, pp. 101–106.
[33] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, Korea, 2001, pp. 94–97.
[34] M. Clerc, "The swarm and the queen: Toward a deterministic and adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 1999, pp. 1951–1957.
[35] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.
[36] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2000, pp. 84–88.
[37] J. Kennedy, "The particle swarm social adaptation of knowledge," in *Proc. IEEE Int. Conf. Evol. Comput.*, Indianapolis, IN, Apr. 1997, pp. 303–308.
[38] P. N. Suganthan, "Particle swarm optimizer with neighborhood operator," in *Proc. IEEE Congr. Evol. Comput.*, Washington DC, 1999, pp. 1958–1962.

[39] A. Ratnaweera, S. Halgamuge, and H. Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.
[40] P. J. Angeline, “Using selection to improve particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, Anchorage, AK, 1998, pp. 84–89.
[41] C. F. Juang, “A hybrid of genetic algorithm and particle swarm optimization for recurrent network design,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 997–1006, Apr. 2004.
[42] Y. P. Chen,W. C. Peng, andM. C. Jian, “Particle swarm optimization with recombination and dynamic linkage discovery,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1460–1470, Dec. 2007.
[43] P. S. Andrews, “An investigation into mutation operators for particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1044–1051.
[44] J. J. Liang and P. N. Suganthan, “Dynamic multi-swarm particle swarm optimizer with local search,” in *Proc. IEEE Congr. Evol. Comput.*, 2005, pp. 522–528.
[45] W. J. Zhang and X. F. Xie, “DEPSO: Hybrid particle swarm with differential evolution operator,” in *Proc. IEEE Conf. Syst., Man, Cybern.*, Oct. 2003, pp. 3816–3821.
[46] F. van den Bergh and A. P. Engelbrecht, “A cooperative approach to particle swarm optimization,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.
[47] A. Ratnaweera, S. Halgamuge, and H. Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.
[48] K. E. Parsopoulos and M. N. Vrahatis, “On the computation of all global minimizers through particle swarm optimization,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 211–224, Jun. 2004.
[49] R. Brits, A. P. Engelbrecht, and F. van den Bergh, “A niching particle swarm optimizer,” in *Proc. 4th Asia-Pacific Conf. Simul. Evol. Learn.*, 2002, pp. 692–696.
[50] R. Brits, A. P. Engelbrecht, and F. van den Bergh, “Locating multiple optima using particle swarm optimization,” *Appl. Math. Comput.*, vol. 189, no. 2, pp. 1859–1883, Jun. 2007.
[51] D. Parrott and X. D. Li, “Locating and tracking multiple dynamic optima by a particle swarm model using speciation,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 440–458, Aug. 2006.
[52] J. Kennedy and R. Mendes, “Population structure and particle swarm performance,” in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1671–1676.
[53] J. Kennedy and R. Mendes, “Neighborhood topologies in fully informed and best-of-neighborhood particle swarms,” *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 36, no. 4, pp. 515–519, Jul. 2006.
[54] X. Hu and R. C. Eberhart, “Multiobjective optimization using dynamic neighborhood particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1677–1681.
[55] J. J. Liang and P. N. Suganthan, “Dynamic multi-swarm particle swarm optimizer,” in *Proc. Swarm Intell. Symp.*, Jun. 2005, pp. 124–129.
[56] R. Mendes, J. Kennedy, and J. Neves, “The fully informed particle swarm: Simpler, maybe better,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, Jun. 2004.
[57] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[58] Zhi-Hui Zhan, , Jun Zhang, , Yun Li, Henry Shu-Hung Chung, "Adaptive Particle Swarm Optimization", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, 2009.

[59] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 36, no. 4, pp. 515–519, Jul. 2006.

[60] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1671–1676.

[61] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, 1998, pp. 69–73.

[62] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, "OPSO: Orthogonal particle swarm optimization and its application to task assignment problems,"*IEEE Trans. Syst., Man, Cybern. A*, vol. 38, no. 2, pp. 288–298,Mar. 2008.