DE GRUYTER OPEN

JAISCR, 2017, Vol. 7, No. 2, pp. 87 – 101
10.1515/jaiscr-2017-0006

SWARM INTELLIGENCE ALGORITHM BASED ON COMPETITIVE PREDATORS WITH DYNAMIC VIRTUAL TEAMS

Shiqin Yang, Yuji Sato

Graduate School of Computer and Information Sciences,Hosei University
3-7-2 Kajino-cho Koganei-shi, Tokyo, Japan

Abstract

In our previous work, Fitness Predator Optimizer (FPO) is proposed to avoid premature convergence for multimodal problems. In FPO, all of the particles are seen as predators. Only the competitive, powerful predator that are selected as an elite could achieve the limited opportunity to update. The elite generation with roulette wheel selection could increase individual independence and reduce rapid social collaboration. Experimental results show that FPO is able to provide excellent performance of global exploration and local minima avoidance simultaneously. However, to the higher dimensionality of multimodal problem, the slow convergence speed becomes the bottleneck of FPO. A dynamic team model is utilized in FPO, named DFPO to accelerate the early convergence rate. In this paper, DFPO is more precisely described and its variant, DFPO-r is proposed to improve the performance of DFPO. A method of team size selection is proposed in DFPO-r to increase population diversity. The population diversity is one of the most important factors that determines the performance of the optimization algorithm. A higher degree of population diversity is able to help DFPO-r alleviate a premature convergence. The strategy of selection is to choose team size according to the higher degree of population diversity. Ten well-known multimodal benchmark functions are used to evaluate the solution capability of DFPO and DFPO-r. Six benchmark functions are extensively set to 100 dimensions to investigate the performance of DFPO and DFPO-r compared with LBest PSO, Dolphin Partner Optimization and FPO. Experimental results show that both DFPO and DFPO-r could demonstrate the desirable performance. Furthermore, DFPO-r shows better robustness performance compared with DFPO in experimental study.

Keywords: swarm intelligence, sitness predator optimizer, dynamic virtual team, population diversity

1 Introduction

Swarm Intelligence (SI) is a relatively new interdisciplinary field of research, which has gained huge popularity these days. It is study of computational systems, which draw inspiration from the collective intelligence emerging from the behavior of groups of simple agents (like bees, ants and birds). The most well-known paradigms in the area of swarm intelligence are Ant Colony Optimization (ACO) [9], Particle Swarm Optimization (PSO) [10], Artificial Bee Colony (ABC) [12], Stochastic Diffusion Search (SDS) [2], [25] and bacterial foraging algorithm [20]. Recently, there are many other nature-inspired meta-heuristic algorithms that proposed as extensions of SI, such as firefly algorithm [34], fireworks algorithm [32], wasp swarm algorithm [26], Glowworm Swarm Optimization

(GSO) [21], [22], Gravitational Search Algorithm (GSA) [27] and so on.

A major problem with most swarm intelligence algorithms in multimodal optimization is premature convergence, which results in great performance loss and sub-optimal solutions [1]. Generally, the fast social collaboration between particles seems to be the reason for loss of population diversity. Diversity declines rapidly in the later iteration period, leaving the optimization algorithm with great difficulties of escaping the local optima. Consequently, the clustering particles with fitness stagnation further exacerbates the premature convergence situation. An accepted hypothesis is that maintenance of high diversity is crucial for preventing premature convergence in multimodal optimization.

Many kinds of optimization algorithms are proposed to improve the diversity of the population for preventing premature convergence. Some of them are inspired by the social behavior of swarms and herds in nature. Moreover, hunting and search behavior of predators are implemented by more and more researchers and proved to be an effective method. For instance, the basic idea of Artificial Fish-Swarm Algorithm (AFSA) [23] is to imitate fish behavior such as preying, swarming, following with local search of individual fish for reaching the global optimum. The Grey Wolf Optimizer (GWO) [24] algorithm mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. The Dolphin Partner Optimization (DPO) [30] mimics the hunting mechanism of dolphins in nature. Bat-inspired Algorithm (BA) [35] based on the echolocation behavior of bats. Bats use a type of sonar, called, echolocation to detect prey, avoid obstacles and locate their roosting crevices in the dark. Krill Herd (KH) [11] is based on the simulation of the herding of the krill swarms in response to specific biological and environmental processes. The most closely associated algorithm with our proposed method is the Predator-Prey Optimizer (PPO) [31]. It is a form of particle swarm optimization where new particles called predators are introduced. The objective of predator is to pursue the best individual in the swarm. The fear probability is the probability of a particle changing its velocity due to the presence of the predator. When all of the particles tend to move toward the best global particle, a predator particle nearby will disturb these particles' movement to avoid premature convergence. The experimental results show that the PPO has a better performance than the traditional PSO [13], [16] in regard to multimodal functions. However, the parameters of the PPO were empirically defined, which greatly influence the robust performance of the algorithm.

Generally, most of swarm intelligence algorithms, such as cuckoo search [36], brain storm optimization algorithm [28], bacterial foraging optimization algorithm [8], focus on social collaboration of population while they seldom concentrate on the individual competition and independent self awareness. The skills of individual competition are effective methods for inspiration to develop intelligent systems and provide solutions to multimodal problems. We assume that the individual competition is more efficient for reducing the rapid social collaboration and increase the ability of being out of the local optimum for the swarm. This motivated our attempt to present a new swarm intelligence optimization, the Fitness Predator Optimizer (FPO) [33] that implements hunting and search behavior of predators, but also emphasizes on the individual competition and independent self awareness.

In an FPO system, all of the individuals are seen as predators. Only the competitive, powerful individuals selected as elites can achieve the limited opportunity to update. The elite team reduces the risk of all of the individuals moving towards the same place. However, in regards to the higher dimensionality of multimodal problem, the slow convergence speed in the early iteration becomes the bottleneck to restrict the improvement of the Fitness Predator Optimizer. In order to resolve this problem, we try to design a superior topology structure of FPO in this paper.

Many investigations about the swarm paradigm [14], [17] have found that the *gbest* type converges quickly on problem solutions but has a weakness for becoming trapped in the local optima, while *lbest* populations are able to escape from local optima, as subpopulations explore different regions. In [15], [18], Kennedy theorized that heterogeneous population structures, with some subsets of the population tightly connected and others relatively isolated, could provide the benefits of both *lbest* and *gbest* sociometries. Motivated by the heterogeneous population structure, a modified dynamic virtual team

is presented in this paper with the aim of accelerating the early convergence rate and improving the global searching capability for FPO. Dynamic virtual team, which was first presented by the Dolphin Partner Optimization (DPO) [30] mimics the hunting mechanism of dolphins in nature. The performance of DPO with the virtual team model is evaluated on several benchmark functions. Experiments show that it could demonstrate the desirable performance. However, the original structure of the dynamic team is not beneficial for the FPO system. Furthermore, the implementation of dynamic virtual team in FPO caused a slight performance degradation. Actually, the individual independent consciousness is weakened by the team leader. Our next work is to propose an appropriate wheel topology instead of the original topology structure in DPO. In this paper, the modified structure of the dynamic virtual team as an appropriate wheel topology is applied to FPO, which is a Dynamic Fitness Predator Optimizer (DFPO).

The rest of the paper is organized as follows: Section 2 provides a brief introduction of related work about FPO, dynamic virtual team and the algorithm for the team model. In Section 3, DFPO-r is proposed. Experiments on ten well-known multimodal functions are conducted in Section 4. The analysis and discussion of the performance of DFPO and DFPO-r are also shown in Section 4. Finally, some concluding remarks and suggestions for future research are provided in Section 5.

2 Related Work

In this Section, the brief introduction of FPO, the basic concepts of dynamic virtual team and the algorithm of a dynamic virtual team model are apprehensively presented.

2.1 Fitness Predator Optimizer

Fitness Predator Optimizer (FPO) [33] is a new bionic-inspired algorithm proposed in our previous work to avoid premature convergence for multimodal problems. In this Section, we mainly focus on the basic principles of FPO and the main function of FPO as described in Algorithm 1.

Principle 1: The limited opportunities are only for the stronger competitors.

To most swarm intelligence algorithms, all particles in a swarm get an equal opportunity to update during each iteration. Unlike most swarm intelligence algorithms, FPO practices a limited number of updates for each iteration. This greatly stimulates the competition among the particles in a swarm. Principle 1 means that a particle with a better fitness function value, that had a prior possibility to update is considered a stronger competitor. The stronger competitor is called the elite. It is worth noting that the elite is not the group's best one, many particles could be the elites when they receive an update opportunity in each iteration.

Principle 2: The elite team is composed of competitors.

Principle 2 defines the components of an elite team. The elite team topology of FPO differs from the previous studies. For example, the global *gbest* topology structure in canonical PSO. In its global *gbest* topology, only one particle is denoted as a global best position (so called *gbest*) for the entire swarm. Some publications [19], [7] show that the *gbest* model converges quickly on problem solutions, but has a weakness for becoming trapped in the local optima. The fast social collaboration between particles will lead all the particles to move toward the global best position. While the elite team topology proposed in FPO is composed of a number of personal best positions instead of single global best position, which reduces the possibility of all the particles moving toward the same position.

Principle 3: The territory intrusion is to take advantage of a companion's position information.

In principle 3, the new territory for elite is the next alternative updated position. If the elite i does not know what is the next best place, a sensible way is to dynamically adjust it according to its own experience and its companions' experience. The definition of updated position is:

$$x^*_{(i,j)} = x_{(i,j)} + \theta * c * w * (x_{(g,j)} - x_{(i,j)}). \quad (1)$$

To the position i, $x_i = (x_{i1}, x_{i2}, ..., x_{id})$ is a vector with d dimensions. $x^*_{(i,j)}$ is the next alternative position and $x_{(i,j)}$ is the current position. $\theta = (rand - 0.5)$ is a random decimal. Generally, $c \in [0,2]$ and w is defined as inertia weight, first proposed by Y.Shi and Eberhart [29] for PSO. $g(g \neq i)$ is a companion of position i.

Having presented three basic principles, the main function of FPO is described in Algorithm 1.

Algorithm 1 Main Function of FPO

```
Initialize population: popsize = n
Initialize x_i: rand(x_i) ∈ (x_min, x_max)
Initialize opportunities: ρ
Repeat
for all particles do
    if x_i get an opportunity to update then
        Generate a new position by equation (1)
        if the new position is better than x_i then
            Update x_i and record it as an elite
        end if
    end if
end for
Until maximum opportunities are attained
```

From the main function of FPO shown in Algorithm 1, we note that the FPO is an iterative, synchronous algorithm. Each individual particle in a swarm moving to a new location carries implicit synchronization on each iteration. As previously mentioned, the slow convergence speed in the early iteration becomes the performance bottleneck for the FPO. In order to improve the optimum searching capability of FPO, an improved FPO with dynamic virtual team is proposed to accelerate the rate of convergence. We hypothesized that such a dynamic team network topology could provide higher quality solutions for FPO. Consequently the mechanism of a dynamic virtual team is described in the next subsection.

2.2 Dynamic Virtual Team

In this Section, the original concept of dynamic virtual team in [30] is introduced at first. Then a simple diagram of virtual team is shown in Figure 1.

The dynamic virtual team is first proposed by the Dolphin Partner Optimization (DPO) [30]. It mimics the hunting mechanism of dolphins in nature. During the hunting process, a dolphin will look for his neighbors and select some of them as his partners. All of his partners and himself form a self-organizing team, which is defined as a dynamic virtual team. In a swarm, each dolphin will do the same clustering behavior at the same time. Here are the related definitions about the dynamic team.

Team: According to the nearest neighbor principle, one dolphin and some of his neighbors are composed of a dynamic virtual team. It should be noted that one dolphin could belong to multiple teams when teams are connected.

Role recognition: A dolphin evaluates himself by comparing the best fitness value with other partners in his virtual team. Normally, the one that has the best fitness value will be selected as the team leader. Each member of the team is recognized as either a leader or an ordinary member.

Exchange: A dolphin provides his best experience information to the team. The team's best experience could be concluded by comparing the team members' experience. The information expansion of individuals and team experience is carried out by the dolphin that belongs to multiple teams.

Leader: The function of team leader dolphin is to analyze all communication information and to predict the most applicable position for the next step.

Member: The function of the ordinary member is to abide with the team leader.

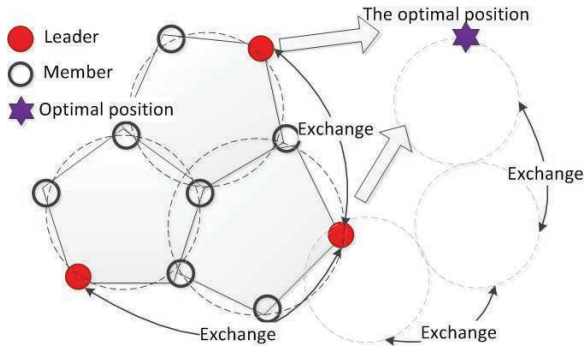


Figure 1. Diagram of dynamic virtual teams

The simple diagram of a virtual team is shown in Figure 1. The solid red circles are team leader and the blank rings present ordinary team members. According to the nearest neighbor rule, each individual dynamically select its 4 immediate neighbors as a team on each iteration. The structure of this team is seemed as an appropriate circle topology. In the circle topology, if two teams have the same members, they are interrelated and affect each other. Otherwise, they are distant from one another and also independent of one another.

Generally, the leader marked with red color has the best fitness value in the team. The main function of the team leader is to analyze all communication information and to predict the most accessible posi-

tion for the further search. The star in purple is the future optimal position that is obtained by one of team leaders from the exchanged information with interrelated teams. The expansion of personal experience and the team's experience is carried out by some interrelated members between the teams. Another function of the ordinary team member is to abide with the team leader which belongs to.

Having accomplished the description of original concepts of dynamic virtual teams, the main organization algorithm of a dynamic virtual team model will be demonstrated in the next subsection.

2.3 Algorithm of Dynamic Virtual Team

First of all, the implementation of the dynamic virtual team model is shown in Algorithm 2. Two important key points shown in lines 20 and 23 in Algorithm 2 are worthy of note. One of them is to predict the next best team position by the team leader, and the other one is to get the next position for the team's ordinary members.

Algorithm 2 Team Organization Function

1: Initialization: $popsize$, $teamsize$, each dolphin x_i
2: **for all** $i = 1 : popsize$ **do**
3: **for all** $g = 1 : popsize$ **do**
4: Calculate distance between x_i and x_g
5: $disM(i,g) = Edistance(x_i, x_g)$
6: **end for**
7: Sort $disM(i,g)$ in ascending
8: $Team_i = sort(disM(i,g))$
9: Select the x_i's dynamic partners from $Team_i$
10: **for all** $h = 2 : teamsize$ **do**
11: $partners(i) = h$
12: **end for**
13: Organize dynamic virtual team for x_i
14: $x_{team(i)} = partners(i)$
15: **end for**
16: **for all** $x_{team(i)}$ **do**
17: Specify the best one as a team leader
18: **if** x_i is the team leader **then**
19: Exchange its best position within team
20: **Predict the next best position**
21: **end if**
22: **if** x_i is an ordinary member **then**
23: **Update its position by equation (2)**
24: **end if**
25: **end for**

It is an open issue as to how to predict and calculate the next team's best position and ordinary member's position. In DPO, the definitions about these two points are not fit for all of the optimal problems. The dynamic team organization mimics the dolphin's teamwork during the process of searching for prey, and attacking prey. It should be noted that the structure of teamwork is a really complicated system, the proposed dynamic virtual team model is only a simplified prototype.

In order to come out the next optimal position, a so-called "Nucleus" is presented in DPO to predict the best position according to the information of individual experience and the team's best experience. As an ordinary member, its next position is updated by equation (2).

$$newx(i) = x(i) + c_1 r_1 (T_{best} - x(i)) + c_2 r_2 (N_{best} - x(i)), \quad (2)$$

where, T_{best} is the best fitness solution among the team, namely the solution of the team leader. N_{best} is the best fitness solution coming from his neighboring teams, which could be acquired from the neighboring team leader.

Under the guidance of the team leader and the influence of neighboring team leaders, members are pulled into the potential global optimum. c_1 and c_2, called cognitive factor and social factor, are positive numbers defined by their upper limit (usually equals to 2.0). Two values r_1 and r_2 are random numbers generated in the interval [0,1].

The definitions of "next best team position" and "ordinary member's position" could not be fit for all kinds of the optimal problems. In addition, the multivariate team size selection of DPO is not discussed in the original paper. In the next Section, we will provide new definitions for the functions of team members and new equations to get the next optimal position. Consequently, the modified dynamic virtual team model would be combined with FPO to form a new dynamic algorithm named DFPO as our main work in this paper.

3 Proposal of DFPO-r

In this Section, our main work contains three parts. First of all, we modified the previous defini-

tions of the virtual team and proposed a new diagram. Secondly, we provided a tentative team size selection for DFPO-r. Finally, the basic pseudocode of DFPO-r is presented in this Section.

3.1 Modified Topology of Team Model

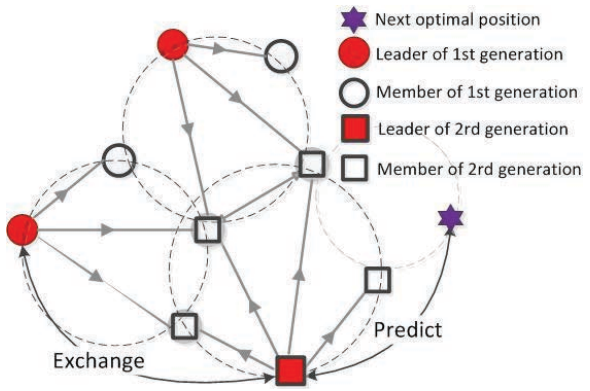


Figure 2. The modified diagram of virtual team

The modified diagram of the virtual team is shown in Figure 2. In the first iteration, each individual is connected to its 3 immediate neighbors. The solid red circles shown in Figure 2 are the team leaders. The neighbors of this team are denoted with black circles, and are not guaranteed to connect with each other. The structure of this team is an appropriate wheel topology. The wheel topology effectively isolates individuals from one another. In the next iteration, the team size is increased to 5, the solid red box shown in Figure 2 is the new team leader. Which is connected to its 4 new immediate neighbours shown with black square. Obviously shown in Figure 2, the team size is dynamic changed during the search process. In addition, team members would not follow up with the team leader and keep their independent consciousness at all time. The function of the team leader is to provide team's best position for the team members. The information of individual best experience and the team's best experience is spread by weak ties of these black squared members. Each of the individuals independently depicts its next position only according to its own experience, the random elite's experience and the neighboring teams' experience. The modified definition of updated position is:

$$\begin{aligned} newx_{(i,j)} = x_{(i,j)} + \theta * c * w * (x_{(g,j)} - x_{(i,j)}) \\ + \theta * c * w * (N_{best} - x_{(i,j)}) \end{aligned}, \quad (3)$$

where $\theta = (rand - 0.5)$ is a random decimal, c is a positive constant defined by its upper limit (usually equals to 2.0). w is defined as inertia weight. x_g is the random elite, N_{best} is the best fitness solution coming from the neighboring teams, which could be acquired from the neighboring team leader.

3.2 Method of Team Size Selection

As shown in Figure 2 the team size could dynamically change during the search process. In addition, the different teams could have a various number of team members in the process of optimization. The method of team size selection is tentatively proposed as an experimental study in this paper.

Generally, we believe that increasing the population size enables the optimization algorithm to search more points and thereby obtain a better solution. However, a large swarm population is not guaranteed to get the better optimized performance. A similar discussion could be found in [37] and [4]. In other words, the team size should be diversified for dynamic environments and various optimization problems.

The diversity of the population is one of the most important factors that determines the performance of the optimal algorithm. We consider that the population diversity is useful for team size Section. A diversity measure $S_{(p)}$ introduced in [5] and [6] is adopted here to indicate the changed of diversity with various populations.

$$\begin{cases} \bar{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij} \\ S_j = \frac{1}{n} \sum_{i=1}^{n} |x_{ij} - \bar{x}_j| \\ S_{(p)} = \frac{1}{d} \sum_{j=1}^{d} S_j \end{cases}, \quad (4)$$

where $S_{(p)}$ denotes the diversity of the swarm p, n is the swarm size, d is the dimensionality of the function, $\bar{x}_j$ represents the pivot of solutions in dimension j, and S_j measures solution diversity based on L_1 norm for dimension j.

An experiment is carried out to confirm if the proper population size is fit for various dimensional functions. It is shown in Figure 3 that the swarm population is not always a case of 'the more, the better'. When the dimension reaches 500, a swarm size of 50 has higher initial population diversity

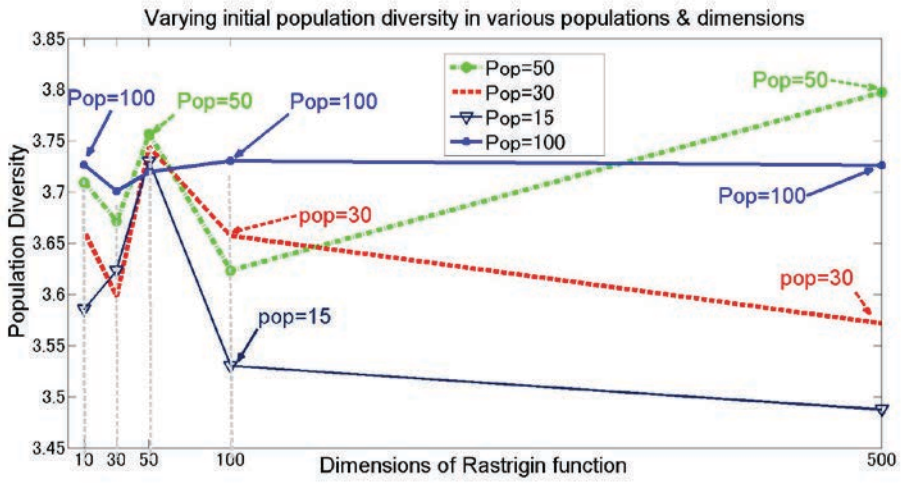


Figure 3. Comparison of varying initial population diversity in various sizes and dimensions

than that of 100. Similarly, the population diversity of the swarm size of 30 is better than that of 50 on 100 dimensions. Inspired by the Competitive Swarm Optimizer [4] and based on our experiment, the method of team size selection is presented in this paper.

Firstly, two kinds of team size are randomly selected from the size pool. The size pool is a boundary of team size, which could be designed according to the real situation. Secondly, both of these team sizes are compared by their population diversities. Only the better one is kept as the team size. Once the dynamic team size is researched, it will be shared with different teams during the whole process of optimization.

Algorithm 3 Main Function of DFPO-r

```
Initialize population: popsize
Initialize x_i: rand(x_i) ∈ (x_min, x_max)
Confirm team size by population diversity
for all Each position x_i do
    Create its team by Dynamic Team Organization Function
end for
for all particles do
    if x_i gets a chance to update its position then
        Generate a new position x_i^* by equation (3)
        if x_i^* is better than x_i then
            Record x_i^* as a new elite
        end if
    end if
end for
For all population use the elitism strategy
```

3.3 Pseudocode of DFPO-r

Algorithm 3 demonstrates the main function of DFPO-r. It should be noted that DFPO-r is a variant of DFPO. The difference between them is the method of team size selection. If the third sentence in Algorithm 3 is deleted, it will be the the main function of DFPO. The team size of DFPO-r is dynamic confirmed under the guidance of population diversity while the DFPO's team size is specified to one third of the population.

4 Experiments

In this Section, ten well-known benchmark functions are used for comparison with GBest PSO [3], LBest PSO [3], DPO, FPO, DFPO and DFPO-r. The aim of the experiments is to compare the global convergence and optimization ability of DFPO and DFPO-r with other four swarm intelligence algorithms on various dimensions of multmodal problems. The analysis and discussion of the experiments are also shown later.

4.1 Evaluation Method

The experimental design is separated by two parts. In part A, four fixed-dimension multimodal problems are used to make a comparison of the convergence rate on Constricted GBest PSO, DPO, FPO and DFPO. In part B, six flexible dimension functions are used to test the capability of global

Table 1. Bounds and global optimums of benchmark functions

Function	Bound	Optimum
Michalewicz: $f_1(x) = -\sum_{j=1}^{d} sin(x_j)[sin(\frac{i.x_i^2}{\pi})]^{20}$	$[0,\pi]^2$	-1.8013
Branin: $f_2(x) = a(x_2 - bx_1^2 + cx_1 - 6)^2$ $+g(1-h)\cos(x_1) + 10,\ a = 1$ $b = 1.25\pi^{-2}, c = 5\pi^{-1}, g = 10, h = 0.125\pi^{-1}$	$x_1 \in [-5,10]$ $x_2 \in [0,15]$	0.397887
Shekel: $f_3(x) = -\sum_{k=1}^{m}(\sum_{j=1}^{4}(x_j - C_{jk})^2 + \beta_k)^{-1}$ $m = 10, \beta = \frac{1}{10}(1,2,2,4,4,6,3,7,5,5)^T$ $C = \begin{pmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3 \\ 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3 \end{pmatrix}$	$[0,10]^4$	-10.5364
Shaffers N.2: $f_4(x,y) = 0.5 + \frac{sin^2(x^2-y^2) - 0.5}{(1+0.001*(x^2+y^2))^2}$	$[-100,100]^2$	0
Rosenbrock: $f_5(x) = \sum_{j=1}^{D}(100*(x_{j+1} - x_j^2)^2 + (x_j - 1)^2)$	$[-5,10]^d$	0
Rastrigin: $f_6(x) = \sum_{j=1}^{d}(x_j^2 - 10*\cos(2\pi x_j)) + 10*d$	$[-5.12,5.12]^d$	0
Griewank: $f_7(x) = \frac{1}{4000}\sum_{j=1}^{d} x_j^2 - \prod_{j=1}^{d}\cos(\frac{x_j}{\sqrt{j}}) + 1$	$[-5,10]^d$	0
Ackley: $f_8(x) = -a*\exp(-0.02*(d^{-1}\sum_{j=1}^{d-1} x_j^2)\frac{1}{2})$ $-\exp(d^{-1}\sum_{j=1}^{d}\cos(2\pi x_j)) + a + \exp, a = 20$	$[-15,30]^d$	0
Levy: $f_9(x) = \sum_{j=1}^{d-1}(\varpi_j - 1)^2[1 + 10sin^2(\pi\varpi_j + 1)]$ $+sin^2(\pi\varpi_1) + (\varpi_d - 1)^2[1 + sin^2(2\pi\varpi_d)]$ $\varpi_j = 1 + \frac{x_j - 1}{4}$	$[-10,10]^d$	0
Schwefel: $f_{10}(x) = 418.9829*d - \sum_{j=1}^{d} x_j sin(\sqrt{\lvert x_j \rvert})$	$[-500,500]^d$	0

optimal solution on Constricted LBest PSO, DPO, FPO, DFPO and DFPO-r.

The experiments are implemented on a PC with an Intel Core i7 860, 2.8 GHz CPU and Microsoft Windows 7 Professional SP1 64-bit operating system, and all algorithms are written in Matlab language. The bounds and global minimums of ten functions are shown in Table 1. The parameters setting are illustrated in Table 2 and Table 3.

4.2 Experimental Results and Discussion

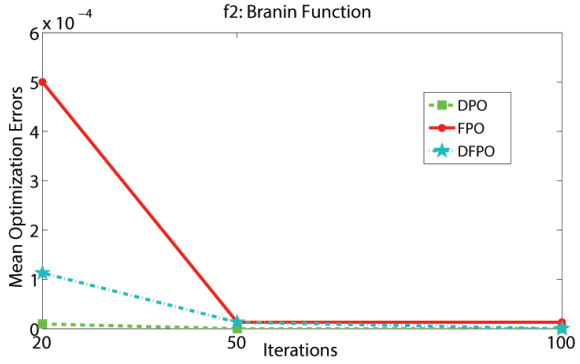


Figure 4. Mean optimization errors of the Branin function with three algorithms in 20, 50 and 100 times iteration respectively

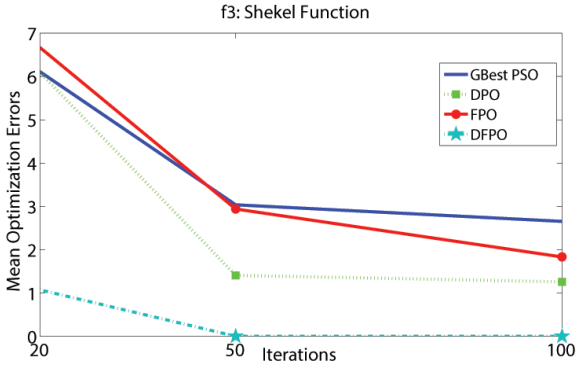


Figure 5. Mean optimization errors of the Shekel function with four algorithms in 20, 50 and 100 times iteration respectively

A systematic discussion are presented here based on experimental results shown in Tables 4, 5 and 6. Table 4 shows the statistical results of optimization errors on four fixed-dimension multimodal functions among GBest PSO, DPO, FPO and DFPO. Table 5 and Table 6 show the global optimal solution capability and average computation time of five algorithms respectively. To all statistical results, the best global optimization errors and mean of best global optimization errors are marked with bold type and down arrow.

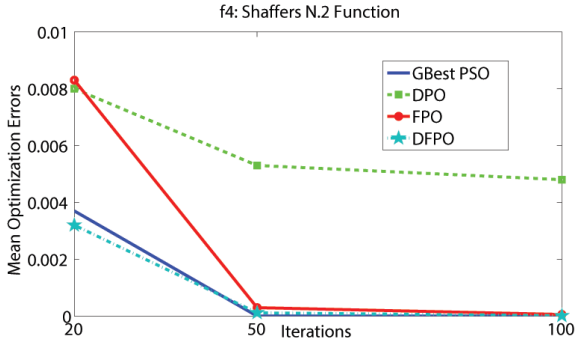


Figure 6. Mean optimization errors of the Shaffers N.2 function with four algorithms in 20, 50 and 100 times iteration respectively

Figures 4, 5 and 6 are the most important processing curves of mean optimization errors obtained by Table 4. From Figure 4, we found that three algorithms (DPO, FPO and DFPO) are able to accurately find the global optimum after 50 iterations. GBest PSO is trapped into the local solution and could not find the accurate global optimum for Branin function. In Figure 5, DFPO is the best ranked and DPO is the second ranked of four algorithms (GBest PSO, DPO, FPO and DFPO). In Figure 6, DFPO still keeps the best global searching capability, but DPO shows the worst performance compared with the others. According to the results of Table 4 and figures, DFPO algorithm could find the satisfied global optimum within 50 iterations. GBest PSO and DPO are trapped into local optimal solutions on Branin function and Shaffers N.2 function respectively. One of the possible reasons is that competitions of predators in DFPO increase individual independence and reduce rapid social collaboration on multimodal functions. These characters in a large degree avoid DFPO dropping into local optimum.

Table 7 shows the ranking of the solution quality based on the statistical results from Table 5 and Table 6. The results of ranking are grouped by functions, dimensions, the best global solutions and mean of optimum solutions, respectively. From the summary statistic data at the bottom of the table 7, it is demonstrated that DFPO-r algorithm is able to provide very competitive results both on the best solution quality and on the average of best optimum

Table 2. Experimental parameters setting of fixed-dimension multimodal functions

Experiment A	Algorithm	Parameters
	GBest PSO	$c1 = c2 = 2.05, \chi \approx 0.72984$
	DPO	$c = 2$, $w \in [0.4, 0.9]$, $teamsize = 10$
Population=100	FPO	$c = 2$, $w \in [0.4, 0.9]$
Run Number=50	DFPO	$c = 2$, $w \in [0.4, 0.9]$, $teamsize = 30$

Table 3. Experimental parameters setting of six flexible dimensional functions

Experiment B	Algorithm	Parameters
	LBest PSO	$c1 = c2 = 2.05$, $\chi \approx 0.72984$
	DPO	$c = 2$, $w \in [0.4, 0.9]$, $teamsize = 5$
Population=50	FPO	$c = 2$, $w \in [0.4, 0.9]$
Run Number=20	DFPO	$c = 2$, $w \in [0.4, 0.9]$, $teamsize = 15$
	DFPO-r	$c = 2$, $w \in [0.4, 0.9]$, $teamsize \in [10, 30]$

Table 4. Statistical results of optimization errors on fixed-dimension multimodal functions

Algorithm	Iter.		f_1	f_2	f_3	f_4
GBestPSO	20	Best	3.00e-04	1.30e-05	1.09e+00	6.17e-06
		Mean	2.90e-03	1.77e-02	6.11e+00	3.70e-03
DPO	20	Best	**0.00e+00**	**0.00e+00**↓	5.33e-02	2.31e-06
		Mean	**0.00e+00**	**9.55e-06**↓	4.09e+00	8.00e-03
FPO	20	Best	**0.00e+00**	1.30e-05	2.09e-02	3.59e-06
		Mean	1.00e-04	5.00e-04	6.67e+00	8.30e-03
DFPO	20	Best	**0.00e+00**	1.30e-05	**4.00e-03**↓	**1.28e-06**↓
		Mean	**0.00e+00**	1.13e-04	**1.07e+00**↓	**3.20e-03**↓
GBestPSO	50	Best	0.00e+00	1.30e-05	5.00e-04	4.04e-10
		Mean	0.00e+00	1.30e-05	3.04e+00	**1.09e-06**↓
DPO	50	Best	0.00e+00	**0.00e+00**↓	2.00e-04	**7.35e-13**↓
		Mean	0.00e+00	**0.00e+00**↓	1.41e+00	5.30e-03
FPO	50	Best	0.00e+00	1.30e-05	7.50e-03	1.78e-07
		Mean	0.00e+00	1.30e-05	2.94e+00	2.98e-04
DFPO	50	Best	0.00e+00	1.30e-05	**0.00e+00**↓	1.82e-08
		Mean	0.00e+00	1.30e-05	**1.40e-03**↓	1.17e-04
GBestPSO	100	Best	0.00e+00	1.30e-05	0.00e+00	1.33e-14
		Mean	0.00e+00	1.30e-05	2.65e+00	**6.42e-12**↓
DPO	100	Best	0.00e+00	**0.00e+00**↓	0.00e+00	**0.00e+00**↓
		Mean	0.00e+00	**0.00e+00**↓	1.26e+00	4.80e-03
FPO	100	Best	0.00e+00	1.30e-05	0.00e+00	5.85e-09
		Mean	0.00e+00	1.30e-05	1.83e+00	5.76e-05
DFPO	100	Best	0.00e+00	0.00e+00	0.00e+00	1.83e-09
		Mean	0.00e+00	0.00e+00	**1.00e-04**↓	1.36e-05

Table 5. Statistical Results of Optimization Errors on 10-Dimension, 50-Dimension Multimodal Benchmark Functions after 20 Trials of 10^3, 2×10^3 function Evaluations, respectively

Algorithm	10-D	f_5	f_6	f_7	f_8	f_9	f_{10}
LBestPSO	Best	3.15e-02	2.00e+00	7.40e-03	4.71e-14	**0.00e+00**	2.37e+02
	Worst	4.41e+00	9.97e+00	4.43e-02	1.01e-12	0.00e+00	9.51e+02
	Mean	**1.83e+00**↓	6.02e+00	1.64e-02	3.98e-13	**0.00e+00**	5.64e+02
DPO	Best	2.48e-02	0.00e+00	2.46e-02	**4.44e-15**	**0.00e+00**	1.18e+02
	Worst	3.97e+00	2.69e+00	6.60e-01	8.00e-15	0.00e+00	1.19e+03
	Mean	2.97e+00	4.98e-01	2.28e-01	**6.04e-15**↓	**0.00e+00**	6.60e+02
FPO	Best	2.20e-03	0.00e+00	1.11e-16	2.60e-12	6.42e-17	1.27e-04
	Worst	1.84e+01	0.00e+00	2.21e-02	2.55e-10	3.08e-16	3.55e+02
	Mean	2.81e+00	0.00e+00	5.90e-03	4.09e-11	2.33e-16	2.11e+02
	Time(s)	4.28e+00	4.31e+00	4.61e+00	4.64e+00	5.08e+00	4.56e+00
DFPO	Best	**4.91e-04**↓	0.00e+00	**0.00e+00**	8.00e-15	5.16e-17	1.27e-04
	Worst	1.72e+01	0.00e+00	1.23e-02	2.00e-14	2.28e-16	1.27e-04
	Mean	2.14e+00	0.00e+00	1.20e-03	1.41e-14	1.03e-16	1.27e-04
	Time(s)	5.26e+00	6.99e+00	5.61e+00	5.69e+00	5.99e+00	1.06e+01
DFPO-r	Best	8.70e-03	0.00e+00	0.00e+00	**4.44e-15**	6.16e-17	1.27e-04
	Worst	7.04e+00	0.00e+00	1.23e-02	8.00e-15	1.09e-16	1.27e-04
	Mean	2.49e+00	**0.00e+00**↓	**2.73e-03**↓	7.64e-15	8.41e-17	1.27e-04
Algorithm	50-D	f_5	f_6	f_7	f_8	f_9	f_{10}
LBestPSO	Best	3.40e+01	1.50e+02	9.85e-12	2.00e-04	2.56e-05	5.68e+03
	Worst	1.36e+02	2.78e+02	7.40e-03	1.95e+00	3.99e+01	8.71e+03
	Mean	6.74e+01	2.29e+02	4.00e-04	3.93e-01	1.05e+01	7.49e+03
DPO	Best	9.64e+01	5.12e+01	3.66e-05	7.70e-03	1.20e-03	4.65e+03
	Worst	2.69e+03	2.13e+02	5.81e-01	1.06e+01	3.45e-01	1.14e+04
	Mean	3.83e+02	1.10e+02	2.48e-02	2.34e+00	7.08e-02	7.20e+03
FPO	Best	**1.60e-02**↓	4.34e-09	7.57e-11	1.75e-05	3.21e-12	1.78e+03
	Worst	1.43e+02	5.97e+00	2.83e-09	1.32e-04	1.66e-09	2.49e+03
	Mean	1.70e+01	1.95e+00	7.62e-10	6.27e-05	1.09e-10	2.23e+03
	Time(s)	8.67e+00	8.96e+00	9.62e+00	9.60e+00	1.22e+01	9.44e+00
DFPO	Best	4.25e-02	8.64e-11	1.98e-14	1.74e-06	4.11e-07	1.18e+03
	Worst	9.84e+01	3.99e+00	5.18e-12	8.63e-06	2.27e-06	1.90e+03
	Mean	**1.49e+01**↓	1.41e+00	3.25e-13	4.08e-06	9.54e-07	1.61e+03
	Time(s)	1.12e+01	1.15e+01	1.21e+01	1.21e+01	1.21e+01	1.19e+01
DFPO-r	Best	1.15e-01	2.26e-12	**1.11e-16**↓	**3.80e-08**↓	**2.55e-15**↓	6.36e-04
	Worst	7.97e+01	3.80e-09	2.03e-12	1.59e-07	1.16e-11	3.55e+02
	Mean	2.50e+01	**2.49e-10**↓	**1.13e-13**↓	**1.00e-07**↓	**1.63e-12**↓	**9.51e+01**↓

Table 6. Statistical Results of Optimization Errors on 100-Dimension Multimodal Benchmark Functions after 20 Trials of 2×10^3 function Evaluations

Algorithm	100-D	f_5	f_6	f_7	f_8	f_9	f_{10}
LBestPSO	Best	3.81e+02	4.97e+02	1.65e-05	1.95e+00	6.41e+01	1.48e+04
	Worst	9.23e+02	7.28e+02	9.00e-04	6.97e+00	1.44e+02	2.17e+04
	Mean	5.65e+02	6.43e+02	8.77e-05	2.76e+00	9.11e+01	1.90e+04
DPO	Best	9.50e+02	2.09e+02	1.53e-02	2.16e+00	1.71e+00	1.22e+04
	Worst	3.34e+03	3.88e+02	9.99e-01	1.34e+01	1.07e+01	2.67e+04
	Mean	1.86e+03	2.88e+02	2.30e-01	5.78e+00	4.01e+00	2.04e+04
FPO	Best	2.89e-01	6.13e+00	2.79e-05	3.10e-03	**1.06e-08**↓	4.91e+03
	Worst	1.40e+02	4.16e+01	1.30e-03	2.17e+00	2.29e+00	6.98e+03
	Mean	**3.13e+01**↓	2.26e+01	5.42e-04	9.23e-01	2.17e-01	5.87e+03
	Time(s)	8.85e+00	9.17e+00	1.02e+01	1.00e+01	1.47e+01	9.77e+00
DFPO	Best	**1.23e-01**↓	1.54e+01	1.60e-06	6.40e-03	1.40e-02	4.51e+03
	Worst	1.39e+02	2.52e+01	3.45e-04	1.25e+00	1.91e-01	6.10e+03
	Mean	3.50e+01	2.05e+01	7.88e-06	5.94e-01	6.15e-02	5.20e+03
	Time(s)	1.22e+01	1.31e+01	1.32e+01	1.31e+01	1.32e+01	1.32e+01
DFPO-r	Best	2.02e+00	**3.07e-08**↓	**1.70e-08**↓	**1.01e-03**↓	5.70e-08	**1.94e+03**↓
	Worst	2.12e+02	9.04e+00	2.03e-07	4.96e-03	3.95e-06	3.46e+03
	Mean	7.68e+01	**3.42e+00**↓	**1.03e-07**↓	**2.36e-03**↓	**6.03e-07**↓	**2.82e+03**↓

Table 7. Ranking of the best solution quality and mean of best solution quality obtained by the Table 5 and Table 6 with 5 kinds of algorithms on 6 multimodal benchmark functions

Fun.	Dim.	Best					Mean				
		PSO	DPO	FPO	DFPO	DFPO-r	PSO	DPO	FPO	DFPO	DFPO-r
f_5	10	5	4	2	1	3	1	5	4	2	3
	50	4	5	1	2	3	4	5	2	1	3
	100	4	5	2	1	3	4	5	1	2	3
f_6	10	5	2.5	2.5	2.5	2.5	5	4	2	2	2
	50	5	4	3	2	1	5	4	3	2	1
	100	5	4	2	3	1	5	4	3	2	1
f_7	10	4	5	3	1.5	1.5	4	5	3	2	1
	50	3	5	4	2	1	4	5	3	2	1
	100	3	5	4	2	1	3	5	4	2	1
f_8	10	4	1.5	5	3	1.5	4	1	5	3	2
	50	4	5	3	2	1	4	5	3	2	1
	100	4	5	2	3	1	4	5	3	2	1
f_9	10	1.5	1.5	5	3	4	1.5	1.5	5	4	3
	50	4	5	2	3	1	5	4	2	3	1
	100	5	4	1	3	2	5	4	3	2	1
f_{10}	10	5	4	2	2	2	4	5	3	1.5	1.5
	50	5	4	3	2	1	5	4	3	2	1
	100	5	4	3	2	1	4	5	3	2	1
Total	Median	4	4	2.75	2	1.25	4	5	3	2	1

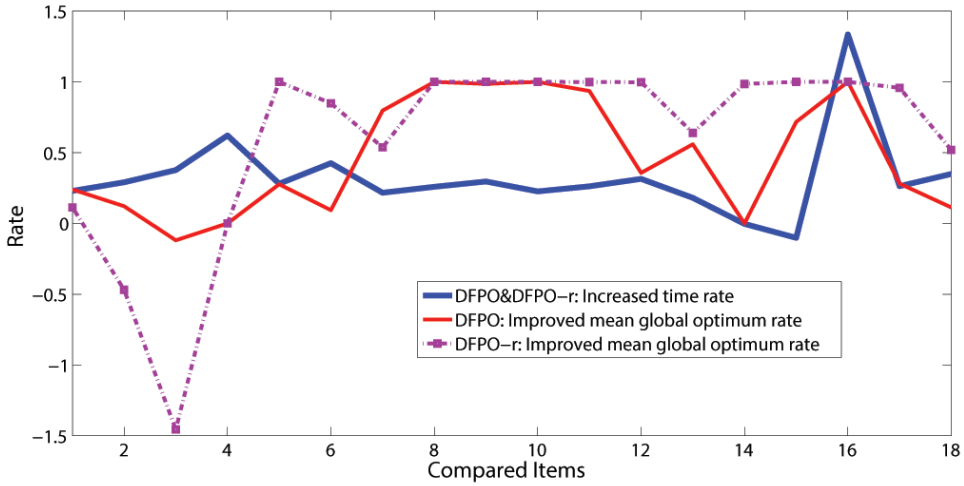


Figure 7. The improved rate of mean global optimum and increased time rate

solution quality even on the flexible dimensional functions. It also suggests that the DFPO-r is more robust for higher dimensional optimization because DFPO-r has the superior performance of escaping local optimums on 100 dimensions.

Figure 7 shows the increased rates of mean global optimum of DFPO and DFPO-r compared with FPO. For the computation time of DFPO-r is very close to DFPO, the run time ratio is only focusd on DFPO and FPO. In Figure 7, the red curve denotes the optimal rates of mean global solution of DFPO compared with FPO in 18 kinds of conditional tests. These 18 kinds of comparable tests come from six multimodal functions using 10, 50 and 100 dimensions, respectively. Similarly, the purple dotted curve represents the same comparisons between DFPO-r and FPO. The blue curve shows the corresponding increased computation time of DFPO compared with FPO. As shown in Figure 7, the improved global solution ratio of DFPO-r is better than DFPO. Consequently, the dynamic team size selection method for DFPO-r is valid for multimodal problems in most cases.

To summarize, the excellent global optimizing abilities of DFPO and DFPO-r show that the paralleled exchanging information system by the dynamic virtual teams could help accelerate the early convergence rate and improve the global searching capability. We note that during the process of improving the performance of DFPO and DFPO-r, the average computation times are slightly increased. However, the increasing space is limited within the reasonable range.

5 Conclusion

To avoid premature problem and enhance the capability of global exploration, a new algorithm, DFPO is proposed to build a paralleled exchanging information system based on dynamic virtual team. However, the fixed team size is not suitable for all of the various real situations. In this paper, the strategy of team size selection is presented based the idea that a team size with higher population diversity is able to prevent solutions from clustering too tightly in the local search space. Then DFPO with a dynamic team size selection strategy is provided as DFPO-r. In DFPO-r, two kinds of team sizes are randomly selected from the size pool, which is designed by a real situation. Comparing their population diversities, the one with higher population diversity will be kept as the current team size. A paralleled exchanging information system is created by these dynamic virtual teams. Which is also able to enhance the global optimal capability and speed up convergence rate during the process of searching. To evaluate the performance of DFPO-r and DFPO, ten well-known benchmark functions are used to compare with GBest PSO, LBest PSO, DPO and FPO. Experimental results demonstrate that both DFPO-r and DFPO have desirable performances for multimodal functions. In

addition, DFPO-r shows better robustness performance in most of cases compared with DFPO. The implementation of DFPO-r on Graphics Processing Units leads to a more efficient algorithm and our future work is focusing on it.

References

[1] Gerardo Beni and Jing Wang, Swarm intelligence in cellular robotic systems, In Robots and Biological Systems: Towards a New Bionics? Springer, 1993, pp. 703–712

[2] J.M. Bishop, Stochastic searching networks, In IEEE Conf. on Artificial Neural Networks, 1989, IEEE, pp. 329–331

[3] Daniel Bratton and James Kennedy, Defining a standard for particle swarm optimization, In Swarm Intelligence Symposium, 2007, IEEE, pp. 120–127

[4] Ran Cheng and Yaochu Jin, A competitive swarm optimizer for large scale optimization, Cybernetics, IEEE Transactions on, vol. 45, 2015, pp. 191–204

[5] Shi Cheng, Yuhui Shi, Quande Qin, TO Ting, and Ruibin Bai, Maintaining population diversity in brain storm optimization algorithm, In Evolutionary Computation, 2014, IEEE, pp. 3230–3237

[6] Shi Cheng, Yuhui Shi, Quande Qin, Qingyu Zhang, and Ruibin Bai, Population diversity maintenance in brain storm optimization algorithm, Journal of Artificial Intelligence and Soft Computing Research, vol. 4, 2014, pp. 83–97

[7] Maurice Clerc and James Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, Evolutionary Computation, IEEE Tansaction on Evolutionary Computation, vol. 6, 2002, pp. 58–73

[8] Swagatam Das, Arijit Biswas, Sambarta Dasgupta, and Ajith Abraham, Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications, In Foundations of Computational Intelligence Volume 3, Springer, 2009, pp. 23–55

[9] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26, 1996, pp. 29–41

[10] R C Eberhart and J Kennedy, A new optimizer using particle swarm theory, Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43

[11] Amir Hossein Gandomi and Amir Hossein Alavi, Krill herd: a new bio-inspired optimization algorithm, Communications in Nonlinear Science and Numerical Simulation, vol. 17, 2012, pp. 4831–4845

[12] Dervis Karaboga and Bahriye Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, Journal of global optimization, vol. 39, 2007, pp. 459–471

[13] James Kenndy and R C Eberhart, Particle swarm optimization, In IEEE International Conference on Neural Networks, 1995, IEEE, pp. 1942–1948

[14] James Kennedy, The behavior of particles, In Evolutionary Programming VII, 1998, Springer, pp. 579–589

[15] James Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, In Proceedings of the 1999 Congress on Evolutionary Computation, 1999, IEEE, pp. 1931–1938

[16] James Kennedy, Particle swarm optimization, In Encyclopedia of Machine Learning, Springer, 2010, pp. 760–766

[17] James Kennedy, James F Kennedy, and Russell C Eberhart, Swarm intelligence, Morgan Kaufmann, 2001

[18] James Kennedy and Rui Mendes, Population structure and particle swarm performance, In Congress on Evolutionary Computation, 2002, IEEE computer Society

[19] James Kennedy and Rui Mendes, Neighborhood topologies in fully informed and best-of-neighborhood particle swarms, IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews, vol. 36, 2006, p. 515

[20] Dong Hwa Kim, Ajith Abraham, and Jae Hoon Cho, A hybrid genetic algorithm and bacterial foraging approach for global optimization, Information Sciences, vol. 177, 2007, pp. 3918–3937

[21] KN Krishnanand and D Ghose, Glowworm swarm optimisation: a new method for optimising multimodal functions, International Journal of Computational Intelligence Studies, vol. 1, 2009, pp. 93–119

[22] KN Krishnanand and Debasish Ghose, Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions, Swarm intelligence, vol. 3, 2009, pp. 87–124

[23] Xiaolei Li, A new intelligent optimization-artificial fish swarm algorithm, Doctor thesis, 2003

[24] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis, Grey wolf optimizer, Adv. Eng. Softw., vol. 69, March 2014, pp. 46–61

[25] S.J. Nasuto and J.M. Bishop, Convergence analysis of stochastic diffusion search, Journal of Parallel Algorithms and Applications, vol. 14, 1999, pp. 89–107

[26] Pedro C Pinto, Thomas A Runkler, and Joao MC Sousa, Wasp swarm algorithm for dynamic max-sat problems, In Adaptive and Natural Computing Algorithms, Springer, 2007, pp. 350–357

[27] Esmat Rashedi, Hossein Nezamabadi-Pour, and Saeid Saryazdi, Gsa: a gravitational search algorithm, Information sciences, vol. 179, 2009, pp. 2232–2248

[28] Yuhui Shi, Brain storm optimization algorithm, In Advances in Swarm Intelligence, Springer, 2011, pp. 303–309

[29] Yuhui Shi and Russell Eberhart, A modified particle swarm optimizer, In Evolutionary Computation, 1998, IEEE, pp. 69–73

[30] Yang Shiqin, Jiang Jianjun, and Yan Guangxing, A dolphin partner optimization, In Proceedings of the 2009 WRI Global Congress on Intelligent Systems - Volume 01, GCIS '09, 2009, pp. 124–128

[31] Arlindo Silva, Ana Neves, and Ernesto Costa, Chasing the swarm: a predator prey approach to function optimisation, In Proceedings of the MENDEL2002—-8th International Conference on Soft Computing, Brno, Czech Republic, 2002

[32] Ying Tan and Yuanchun Zhu, Fireworks algorithm for optimization, In Advances in Swarm Intelligence, Springer, 2010, pp. 355–364

[33] Shiqin Yang and Yuji Sato, Fitness predator optimizer to avoid premature convergence for multimodal problems, In Systems, Man and Cybernetics, 2014 IEEE International Conference on, 2014, IEEE, pp. 258–263

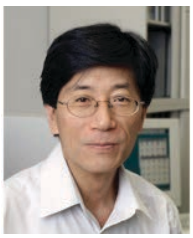
[34] Xin-She Yang, Nature-inspired metaheuristic algorithms, Luniver press, 2010

[35] Xin-She Yang, A new metaheuristic bat-inspired algorithm, In Nature inspired cooperative strategies for optimization (NICSO 2010), Springer, 2010, pp. 65–74

[36] Xin-She Yang and Suash Deb, Cuckoo search via lévy flights, In World Congress on Nature & Biologically Inspired Computing, NaBIC, 2009, IEEE, pp. 210–214

[37] You Zhou and Ying Tan, Gpu-based parallel particle swarm optimization, In Evolutionary Computation, 2009, CEC'09, IEEE Congress on, 2009, IEEE, pp. 1493–1500

Shiqin Yang received the M.Eng degree from JiangNan University, Wuxi, China, in 2008. She is currently pursuing the Ph.D. degree with the Graduate School of Computer and Information Sciences, Hosei University, Tokyo, Japan. Her research interests include swarm intelligence, continuous dynamic constrained optimization and fuzzy clustering algorithms.



Yuji Sato received the B.Eng. and Ph.D. degrees in engineering from the University of Tokyo, Japan in 1981 and 1997, respectively. He is a professor of Faculty of Computer and Information Sciences at Hosei University, Japan. From 2007 to 2008, he was a visiting scholar at Illinois Genetic Algorithms Laboratory (IlliGAL). His research interests include evolutionary computation on many-core architecture, swarm intelligence for multimodal problems and evolution of machine learning techniques in design. He has won the 2015 highly commended paper award in International Journal of Intelligent Computing and Cybernetics. He co-organized a special session "EC on Many-core Architecture to Solve Large-scale Problems" in CEC2011, "Computational Intelligence on Consumer Games and Graphics Hardware CIGPU" in WCCI2012, "Parallel and Distributed Evolutionary Computation in the Cloud Era" in CEC2013 and CEC2015, and he is a member of program committee of GECCO since 1999.