ORIGINAL RESEARCH

Fuzzy adaptive catfish particle swarm optimization

Li-Yeh Chuang[1], Sheng-Wei Tsai[2], Cheng-Hong Yang[2]

1. Institute of Biotechnology and Chemical Engineering, I-Shou University, Kaohsiung, Taiwan 80041. 2. Department of Electronic Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan 80778.

Correspondence: Cheng-Hong Yang. Address: Department of Electronic Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan 80778. Telephone: 88-673-814-526 ext 5639. E-mail: chyang@cc.kuas.edu

Received: March 29, 2012 **Accepted:** May 11, 2012 **Published:** December 1, 2012
DOI: 10.5430/air.v1n2p149 **URL:** http://dx.doi.org/10.5430/air.v1n2p149

Abstract

The catfish particle swarm optimization (CatfishPSO) algorithm is a novel swarm intelligence optimization technique. This algorithm was inspired by the interactive behavior of sardines and catfish. The observed catfish effect is applied to improve the performance of particle swarm optimization (PSO). In this paper, we propose fuzzy CatfishPSO (F-CatfishPSO), which uses fuzzy to dynamically change the inertia weight of CatfishPSO. Ten benchmark functions with 10, 20, and 30 different dimensions were selected as the test functions. Statistical analysis of the experimental results indicates that F-CatfishPSO outperformed PSO, F-PSO and CatfishPSO.

Key words

Particle swarm optimization, CatfishPSO, Fuzzy logic, Fuzzy System

1 Introduction

Particle swarm optimization (PSO) is motivated by natural evolution and simulates the social behavior of organisms to describe an automatically evolving system. The PSO algorithm is a stochastic, population-based evolutionary computer algorithm developed by Kennedy and Eberhart in 1995 [1]. Over the past decade, PSO has been successfully employed to many application areas, and generally obtains better results in a faster and cheaper way than other methods [2].

A most important consideration in PSO is to effectively balance the global and local search abilities of the swarm since the balance between global and local search throughout the process is critical to the success of PSO [3]. PSO shows promising performance on nonlinear function optimization and has thus received much attention [4]. However, the local search ability of PSO is rather poor [5] and often results in premature convergence, especially under circumstances where PSO is applied to complex multi-peak search problems [6]. For this reason, a strategic parameter, the well-known inertia weight factor, was introduced by Shi and Eberhart [7] in an effort to strike a better balance between global exploration and local exploitation. The inertia weight factor was introduced into the velocity update equation of the original PSO. A large inertia weight with a value greater than 1.0 facilitates global exploration, while a small inertia weight with a value smaller than 1.0 facilitates local exploitation. An initially large inertia weight value that progressively decreases throughout the process allows particles to move around a broader search space and to move on to more promising regions of the search space. The original authors suggested using an inertia weight that linearly decreases from 0.9 to 0.4. The addition of the inertia weight greatly improved the performance of PSO [8].

After the inertia weight parameter was proposed, numerous methods have been proposed that focus on the modification of the inertia weight during the search process in order to improve the performance of PSO. Jiao et al. proposed dynamic inertia weight PSO, which uses a dynamic inertia weight that decreases based on iterative generations [9]. Yang et al. introduced a modified particle swarm optimization algorithm with dynamic adaptation, in which a modified velocity update formula with a dynamically changing inertia weight based on the run and evolution state for each particle is used [10]. An adaptive inertia weight factor (AIWF) was introduced by Liu et al. to efficiently balance the exploration and exploitation abilities [11]. Chuanwen and Bompard adopted an inertia weight parameter generated by chaotic map [12]. Shi and Eberhart used a fuzzy adaptive inertia weight, which improved the performance over PSO with a linearly decreasing inertia weight [3].

The fuzzy approach does not act blindly because fuzzy logic includes operations for interference by a user, namely user-defined rules governing the target control system. Such systems can be tailored to various problems [13]. The main source of fuzziness is the imprecision involved in defining and using symbols as a property of language. In most existing applications, the fuzzy rules encode expert reasoning into a program to make a decision or control the system [14]. Fuzzy systems are more suitable for complex system problems, and have been successfully applied in an increasing number of application areas.

In this paper, we propose fuzzy adaptive CatfishPSO (F-CatfishPSO), in which fuzzy systems are applied to adapt the inertia weight and thus improve the performance of the CatfishPSO algorithm. In CatfishPSO, the catfish effect introduces a competition function into a group of individuals. The introduced catfish particles improve the solution quality of PSO [15]. A fuzzy adaptive inertia weight for the CatfishPSO was introduced in our study to improve the search behavior and to prevent entrapment of particles in a locally optimal solution. The proposed method was then applied to ten benchmark functions from the literature. Experimental results and statistical analysis show that the performance of F-CatfishPSO is superior to PSO, F-PSO and CatfishPSO.

2 Method

2.1 Particle swarm optimization (PSO)

In original PSO [1], each particle is analogous to an individual "fish" in a school of fish. It is a population-based optimization technique, where a population is called a swarm. A swarm consists of N particles moving around a D-dimensional search space. The position of the ith particle can be represented by $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. The velocity for the ith particle can be written as $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The positions and velocities of the particles are confined within $[X_{\min}, X_{\max}]^D$ and $[V_{\min}, V_{\max}]^D$, respectively. Particles coexist and evolve simultaneously based on knowledge shared with neighboring particles; they make use of their own memory and knowledge gained by the swarm as a whole to find the best solution. The best previously visited position of the ith particle is noted as its individual best position $p_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, a value called $pbest_i$. The best value of the all individual $pbest_i$ values is denoted the global best position $g = (g_1, g_2, \ldots, g_D)$ and called *gbest*. The PSO process is initialized with a population of random particles, and the algorithm then executes a search for optimal solutions by continuously updating generations. At each generation, the position and velocity of the ith particle are updated by $pbest_i$ and *gbest* in the swarm. The update equations can be formulated as:

$$v_{id}^{new} = w \times v_{id}^{old} + c_1 \times r_1 \times \left(pbest_{id} - x_{id}^{old}\right) + c_2 \times r_2 \times \left(gbest_d - x_{id}^{old}\right) \tag{1}$$

$$x_{id}^{new} = x_{id}^{old} + v_{id}^{new} \tag{2}$$

 ISSN 1925-4040 E-ISSN 1925-4059

where r_1 and r_2 are random numbers between (0, 1), and c_1 and c_2 are acceleration constants that control how far a particle moves in a single generation. Velocities v_{id}^{new} and v_{id}^{old} are denote the velocities of the new and old particle, respectively. x_{id}^{old} is the current particle position, and x_{id}^{new} is the new, updated particle position. The inertia weight w controls the impact of the previous velocity of a particle on its current one [7]. In general, the inertia weight decreases linearly from 0.9 to 0.4 throughout the search process to effectively balance the global and local search abilities of the swarm [8]. The equation can be written as:

$$w = (w_{\max} - w_{\min}) \times \frac{Iteration_{\max} - Iteration_i}{Iteration_{\max}} + w_{\min} \tag{3}$$

In Eq. (3), $w_{\max}$ is 0.9, $w_{\min}$ is 0.4 and $Iteration_{\max}$ is the maximum number of allowed iterations. The pseudo-code of the PSO process is shown below.

```
01: begin
02:  Randomly initialize particles swarm
03:  while (number of iterations, or the stopping criterion is not met)
04:    Evaluate fitness of particle swarm
05:    for n = 1 to number of particles
06:      Find pbest
07:      Find gbest
08:      for d = 1 to number of dimension of particle
09:        update the position of particles by Eq. (1) and Eq. (2)
10:      next d
11:    next n
12:    update the inertia weight value by Eq. (3)
13:  next generation until stopping criterion
14: end
```

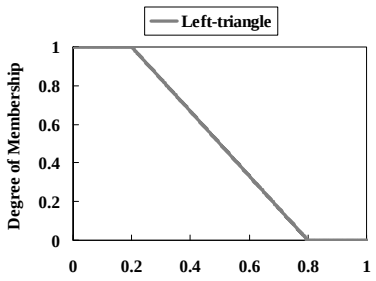
Figure 1. PSO pseudo-code

2.2 Fuzzy particle swarm optimization (F-PSO)

Unlike the above-mentioned PSO [8], the inertia weight in F-PSO does not linearly decrease from 0.9 to 0.4 throughout the search process but is dynamically adapted by a fuzzy system. Three fuzzy variables are adopted [3], i.e., the current best performance evaluation (CBPE) and the current inertia weight (Weight) are selected as two input variables to the fuzzy system; the change of the inertia weight (Weight-change) is the output variable. Each fuzzy variable has three fuzzy sets, namely High, Medium and Low, which correspond to the membership functions Right-triangle, Triangle and Left-triangle, respectively. These membership functions are:

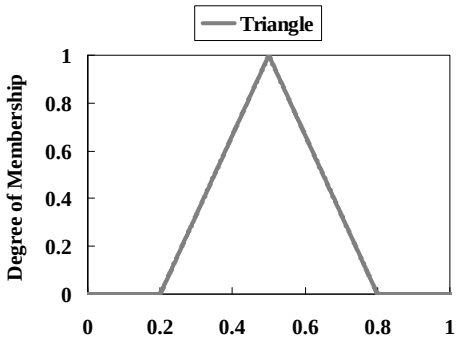
Left_Triangle membership function

$$f_{Left\_Triangle}(x) = \begin{cases} 1 & if \quad x < x_1 \\ \dfrac{x_2 - x}{x_2 - x_1} & if \quad x_1 \le x \le x_2 \\ 0 & if \quad x > x_2 \end{cases}$$



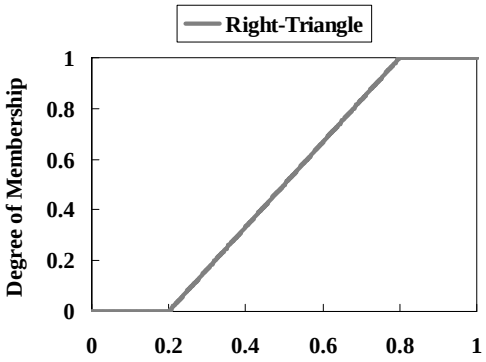
Triangle membership function

$$f_{Triangle}(x)=\begin{cases}0 & if \quad x<x_1\\ 2\dfrac{x-x_1}{x_2-x_1} & if \quad x_1\le x\le \dfrac{x_2+x_1}{2}\\ 2\dfrac{x_2-x}{x_2-x_1} & if \quad \dfrac{x_2+x_1}{2}<x\le x_2\\ 0 & if \quad x>x_2\end{cases}$$



Right_Triangle membership function

$$f_{Right\_Triangle}(x)=\begin{cases}0 & if \quad x<x_1\\ \dfrac{x-x_1}{x_2-x_1} & if \quad x_1\le x\le x_2\\ 1 & if \quad x>x_2\end{cases}$$



In the membership functions, x^1 and x^2 are critical parameters which determine the shape and location of the functions. The graph of each membership function is drawn from x^1=0.2 to x^1=0.8. Although other membership functions, e.g., Gaussian and Sigmoid functions, can also be employed in fuzzy system, the membership functions used here are useful for a variety of problems and can be easily implemented in microcontrollers and microprocessors [3]. CBPE has been proven to be an excellent measure for the performance of the best candidate solution in PSO [3]. However, since different problems have a different range of performance measurement values, CBPE has to be converted into a normalized format so that it can be applied to a wide range of optimization problems in fuzzy systems design [3]. For minimization problems, the normalized CBPE (NCBPE) can be formulated as:

$$NCBPE=\frac{CBPE-CBPE_{\min}}{CBPE_{\max}-CBPE_{\min}} \tag{4}$$

In Eq. (4), $CBPE_{\max}$ denotes the threshold that judges whether CBPE is an acceptable solution or not. For example, CBPE is not an acceptable solution if the solution is greater or equal to the $CBPE_{\max}$. $CBPE_{\min}$ denotes the estimated minimum. In the whole fuzzy system, we use the OR operator, that is we take the maximum value as the membership value of the fuzzy set for adapting the inertia weight of PSO. Table 1 defines the fuzzy system for adapting the inertia weight of PSO. Figs. 2-4 illustrate the membership function of three fuzzy variables NCBPE, Weight and Weight-change, respectively.

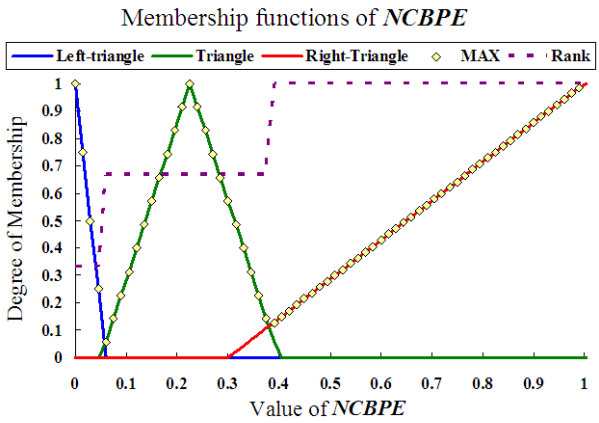


Figure 2. Membership function of variable *NCBPE*

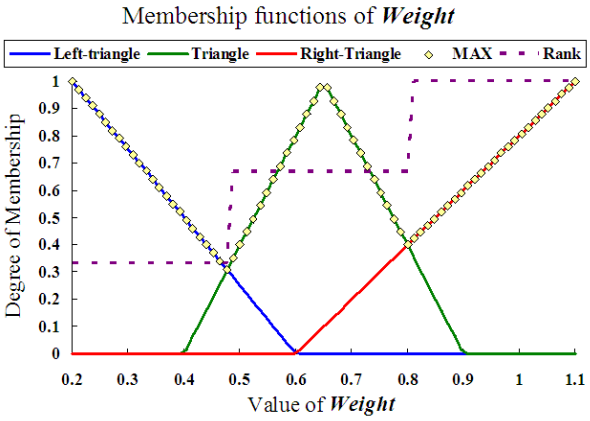


Figure 3. Membership function of variable *Weight*

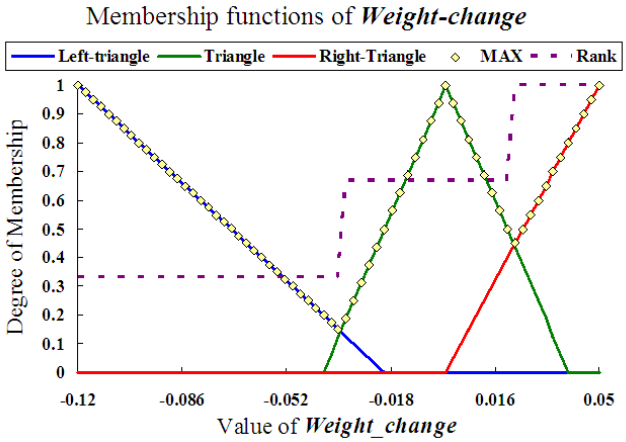


Figure 4. Membership function of variable *Weight-change*

Table 1. Definition of the fuzzy system for adapting the inertia weight of PSO.

Membership function	**NCBPE (input)**		**Weight (input)**		**Weight-change (output)**	
(Fuzzy set)	x^1	x^2	x^1	x^2	x^1	x^2
Left-triangle (Low, L)	0	0.06	0.2	0.6	-0.12	-0.02
Triangle (Medium, M)	0.05	0.4	0.4	0.9	-0.04	0.04
Right-triangle (High, H)	0.3	1	0.6	1.1	0	0.05
Dynamic range	(0, 1)		(0.2, 1.1)		(-0.12, 0.05)	
	L		L		M	
	L		M		L	
	L		H		L	
	M		L		H	
Nine rules in the fuzzy system	M		M		M	
	M		H		L	
	H		L		H	
	H		M		M	
	H		H		L	

The pseudo-code of F-PSO is shown below.

```
01: begin
02:  Randomly initialize particles swarm and inertia weight
03:  while (number of iterations, or the stopping criterion is not met)
04:    Evaluate fitness of particle swarm
05:    for n = 1 to number of particles
06:      Find pbest
07:      Find gbest
08:      for d = 1 to number of dimension of particle
09:        update the position of particles by Eq. (1) and Eq. (2)
10:      next d
11:    next n
12:    update the inertia weight value by fuzzy system
13:  next generation until stopping criterion
14: end
```

Figure 5. F-PSO pseudo-code

2.3 Catfish particle swarm optimization (CatfishPSO)

The underlying idea for the development of CatfishPSO was derived from the catfish effect observed when catfish are introduced into large holding tanks of sardines [15]. The catfish-sardine competition stimulates renewed movement amongst the sardines. Similarly, the introduced catfish particles stimulate a renewed search by the other sardine particles in CatfishPSO. In other words, the catfish particles can guide particles trapped in a local optimum on to a new region of the search space, and thus to potentially better particle solutions. The introduction of catfish particles in CatfishPSO is very simple and can be done without increasing the computational complexity of the process. Catfish particles overcome the inherent defects of PSO by initializing a new search over the entire search space from its extreme points.

In CatfishPSO, a population is randomly initialized in a first step and particles are distributed over a *D*-dimensional search space. The position and velocity of each particle are updated by Eqs. (1)-(3). If the distance between *gbest* and the surrounding particles is small, each particle is considered a part of the cluster around *gbest* and will only move a very small distance in the next generation. To avoid this premature convergence, catfish particles are introduced and replace the 10%

of the original particles with the worst fitness values of the swarm. These catfish particles are essential for the success of a given optimization task. The pseudo-code of CatfishPSO is shown below. Further details on CatfishPSO mechanisms can be found in Chuang et al. [15].

```
01: Begin
02:  Randomly initialize particles swarm
03:  while (number of iterations, or the stopping criterion is not met)
04:    Evaluate fitness of particle swarm
05:    for n = 1 to number of particles
06:      Find pbest
07:      Find gbest
08:      for d = 1 to number of dimension of particle
09:        Update the position of particles by Eq. (1) and Eq. (2)
10:      next d
11:    next n
12:    if fitness of gbest is the same seven times then
13:      Sort the particle swarm via fitness from best to worst
14:      for n = number of nine-tenths of particles to number of particles
15:        for d = 1 to number of dimension of particle
16:          Randomly select extreme points at Max or Min of the search space
17:          Reset the velocity to 0
18:        next d
19:      next n
20:    end if
21:    Update the inertia weight value by Eq. (3)
22:  next generation until stopping criterion
23: end
```

Figure 6. CatfishPSO pseudo-code

2.4 Fuzzy catfish particle swarm optimization (F-CatfishPSO)

In F-CatfishPSO, a fuzzy system designed by experts [3] is implemented to dynamically adapt the inertia weight of the CatfishPSO [15]. The fuzzy system improves the performance of CatfishPSO significantly. The pseudo-code of F-CatfishPSO is shown below.

```
01: Begin
02:  Randomly initialize particles swarm and inertia weight
03:  while (number of iterations, or the stopping criterion is not met)
04:    Evaluate fitness of particle swarm
05:    for n = 1 to number of particles
06:      Find pbest
07:      Find gbest
08:      for d = 1 to number of dimension of particle
09:        Update the position of particles by Eq. (1) and Eq. (2)
10:      next d
11:    next n
12:    if fitness of gbest is the same seven times then
13:      Sort the particle swarm via fitness from best to worst
14:      for n = number of nine-tenths of particles to number of particles
15:        for d = 1 to number of dimension of particle
```

```
16:        Randomly select extreme points at Max or Min of the search space
17:        Reset the velocity to 0
18:      next d
19:    next n
20:   end if
21:   Update the inertia weight value by fuzzy system
22: next generation until stopping criterion
23: end
```

Figure 7. F-CatfishPSO pseudo-code

3 Numerical simulation

3.1 Benchmark functions

In order to illustrate, compare, and analyze the effectiveness and performance of PSO, F-PSO, CatfishPSO, and F-CatfishPSO algorithms for optimization problems, ten representative benchmark functions were used to test the algorithms. These ten benchmark functions are shown below.

Sphere

$$f_1(x)=\sum_{i=1}^{D} x_i^2 \tag{6}$$

Ellipsoid

$$f_2(x)=\sum_{i=1}^{D} i x_i^2 \tag{7}$$

Sum of difference power

$$f_3(x)=\sum_{i=1}^{D} |x_i|^{i+1} \tag{8}$$

Cigar

$$f_4(x)=x_1^2+10\sum_{i=2}^{D} x_i^2 \tag{9}$$

Ridge

$$f_5(x)=\sum_{i=1}^{D}\sum_{j=1}^{i} x_j^2 \tag{10}$$

Step

$$f_6(x)=\sum_{i=1}^{D} \lfloor |x_i|+0.5 \rfloor^2 \tag{11}$$

Rosenbrock

$$f_7(x)=\sum_{i=1}^{D-1}\left(100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right) \tag{12}$$

Rastrigrin

$$f_8(x)=\sum_{i=1}^{D}\left(x_i^2-10\cos(2\pi x_i)+10\right) \tag{13}$$

Griewark

$$f_9(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1 \tag{14}$$

Ackley

$$f_{10}(x)=-20\exp^{\left(-0.2\times\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)}-\exp^{\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)}+20+e \tag{15}$$

These ten benchmark functions can be grouped into unimodal (Sphere, Ellipsoid, Sum of difference power, Cigar, Ridge, Step and Rosenbrock) and multimodal functions (Rastrigrin, Griewark and Ackley). In multimodal functions the number of local minima increases exponentially with the problem dimension.

3.2 Parameter settings

In our experiments, three different dimension sizes (Dim.) were tested for each function, namely 10, 20 and 30 dimensions, and the corresponding maximum number of generations (Gen.) was set to 1000, 1500 and 2000, respectively. In addition, four population sizes (Pop.) were used for each function with a different dimension, namely population sizes of 20, 40, 80 and 160, respectively. The same sets of parameters were assigned for PSO, F-PSO, CatfishPSO and F-CatfishPSO, i.e., $c_1=c_2=2$. For each experimental setting, we executed 50 independent runs for PSO, F-PSO, CatfishPSO and F-CatfishPSO. The parameter settings of the ten benchmark functions are summarized in Table 2.

Table 2. Parameter settings of the ten benchmark functions

Function	Modality	Search Space	Asymmetric Initialization Range	X_{min}	X_{max}	$CBPE_{max}$	Optimum
01. Sphere	Unimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0
02. Ellipsoid	Unimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0
03. Sum of difference power	Unimodal	$-3\le x_i\le 3$	$1.5\le x_i\le 3.0$	-3	3	BI	0
04. Cigar	Unimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0
05. Ridge	Unimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0
06. Step	Unimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0
07. Rosenbrock	Unimodal	$-100\le x_i\le 100$	$15\le x_i\le 30$	-100	100	500	0
08. Rastrigrin	Multimodal	$-10\le x_i\le 10$	$2.56\le x_i\le 5.12$	-10	10	70	0
09. Griewark	Multimodal	$-600\le x_i\le 600$	$300\le x_i\le 600$	-600	600	0.15	0
10. Ackley	Multimodal	$-100\le x_i\le 100$	$50\le x_i\le 100$	-100	100	BI	0

Legend: BI: best initial fitness value

3.3 Experimental results and discussion

The performances of PSO [8], F-PSO [3], CatfishPSO [15], and F-CatfishPSO methods were compared by means of the best fitness and the standard deviation among ten benchmark functions for different population, dimension and generation values. We adopted a z-test on pairs of group results to validate the results as statistical different for the methods. The *z*-test gives a *p*-value which is compared to a constant called α to determine whether a difference between alternative method is statistically significant or not. In our case, we adopted a 90%-confidence interval for the *z*-test in all benchmark functions, that is, if the *p*-value of a method is *p*-value $<\alpha = 0.1$, the method is significantly improved. The mean fitness values and standard deviations of PSO, F-PSO, CatfishPSO and F-CatfishPSO for the ten benchmark functions are listed in Tables 3

to 12. If any mean fitness values and standard deviations are<10-300, 0.000±0.000 is displayed in these Tables. The experimental results in Tables 3 to 12 are divided into three areas for analysis:

3.3.1 F-PSO compared to PSO

The results for PSO and F-PSO in tables 3-12 indicate that F-PSO outperformed PSO on all 10 test functions. However, statistical analysis of the results with a *z*-test at $\alpha = 0.1$ revealed that F-PSO is only significantly superior to PSO in the Sphere, Ellipsoid, Sum of difference power, Cigar, Ridge and Step functions.

3.3.2 CatfishPSO compared to PSO

Tables 3 to 12 indicate that CatfishPSO also outperformed PSO on all 10 benchmark functions for each experimental setting tested. This result is supported by statistical analysis with a *z*-test at $\alpha = 0.1$. It can be seen that CatfishPSO performed admirably for different population sizes and dimensions and can thus play an important role in reducing the computational cost of real-time online problems where time is critical.

Table 3. Mean function value for Sphere function (Unimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	2.98E-20±8.10E-20	1.86E-12±1.32E-11	**0.000±0.000**	**0.000±0.000**
	20	1500	0	2.23E-11±1.09E-10	2.00E-05±9.35E-05	**0.000±0.000**	**0.000±0.000**
	30	2000	0	600.00±2398.979	2.96E-02±1.02E-01	**0.000±0.000**	**0.000±0.000**
40	10	1000	0	6.67E-25±1.54E-24	1.01E-83±6.47E-83	**0.000±0.000**	**0.000±0.000**
	20	1500	0	5.83E-15±1.40E-14	4.12E-13±9.92E-12	**0.000±0.000**	**0.000±0.000**
	30	2000	0	5.88E-11±9.83E-11	1.25E-08±8.83E-08	**0.000±0.000**	**0.000±0.000**
80	10	1000	0	3.86E-28±1.29E-27	3.4E-100±1.61E-99	**0.000±0.000**	**0.000±0.000**
	20	1500	0	2.55E-18±4.08E-08	8.87E-72±3.12E-71	**0.000±0.000**	**0.000±0.000**
	30	2000	0	1.98E-13±4.67E-13	1.13E-57±4.31E-57	**0.000±0.000**	**0.000±0.000**
160	10	1000	0	6.83E-32±2.18E-31	7.5E-116±4.7E-115	**0.000±0.000**	**0.000±0.000**
	20	1500	0	2.66E-21±6.33E-21	1.72E-86±8.38E-86	**0.000±0.000**	**0.000±0.000**
	30	2000	0	4.90E-16±1.10E-15	2.33E-71±9.61E-71	**0.000±0.000**	**0.000±0.000**
Average				50.00±0199.915	2.20E-01±5.51E-01	**0.000±0.000**	**0.000±0.000**
p-value				0.646	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	No
Rank				0	1	2	**2**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at $\alpha = 0.1$ between the tested method and another method by *z*-test.

3.3.3 F-CatfishPSO compared to both F-PSO and CatfishPSO

Although CatfishPSO performed well on the above-mentioned ten benchmark functions, it can still be improved by fuzzy adaptation of the inertia weight. The inertia weight of the original CatfishPSO is linearly decreasing from 0.9 to 0.4 throughout the search process and thus can not be adapted [15]. Previous experimental results for PSO and F-PSO indicate that using a fuzzy system to dynamically adapt the inertia weight of CatfishPSO can improve the performance [3]. Tables 3 to 12 show that F-CatfishPSO outperformed both CatfishPSO and F-PSO on all benchmark functions, a fact that could again be validated by statistical analysis with a z-test at $\alpha = 0.1$.

The quality of solutions achieved by CatfishPSO is further improved when a fuzzy system adapts the inertia weight (F-CatfishPSO). The rank of F-CatfishPSO for the Ellipsoid, Sum of difference power, Ridge, Rosenbrock and Ackley functions (see Tables 4, 5, 7, 9 and 12) is higher than the rank of CatfishPSO. The rank indicates the numbers of significance at $\alpha = 0.1$ between the tested method and another method and is determined by the z-test. These experimental results and the statistical analysis thereof clearly demonstrate that the performance of F-CatfishPSO is superior to PSO, F-PSO, and CatfishPSO.

Figures 8 to 17 plot the mean best fitness in the form of logarithm values over the number of generations for PSO [8], F-PSO [3], CatfishPSO [15] and F-CatfishPSO with 20 particles on the ten 30-dimensional benchmark functions. Tables 3, 8, 10 and 11 indicate that CatfishPSO and F-CatfishPSO are both capable of finding optimal solutions, i.e., the fitness value were $< 10^{-300}$. For the sake of convenience, the graphs in Figures 8, 13, 15 and 16 are only displayed until the mean best fitness values reaches 10^{-3}. The figures again show that the search efficiency of F-CatfishPSO is superior to PSO, F-PSO, and CatfishPSO on all ten benchmark functions. In addition, Figures 8, 13, 15 and 16 demonstrate that F-CatfishPSO and CatfishPSO are capable of finding optimal solutions within 1200 generations, a fact also demonstrated in Tables 3, 8, 10 and 11, where no standard deviation can be given for CatfishPSO and F-CatfishPSO. In Figs. 8 to 17, both CatfishPSO and F-CatfishPSO display a distinctly step-shaped curve, a phenomenon indicative of CatfishPSO's capability to leave local optimal solutions and renew the search in other regions of the search space. The F-PSO curve, on the other hand indicates that F-PSO reaches near optimal solutions very early on in the search, but it lacks the capability of CatfishPSO to break through locally optimal solutions. It can be concluded that by combining a fuzzy system dynamically adapts the inertia weight with CatfishPSO a significant improvement of performance can be achieved.

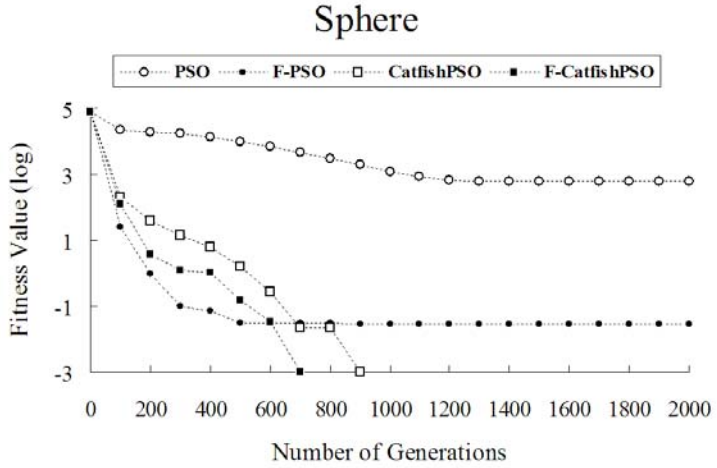


Figure 8. ***Sphere*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

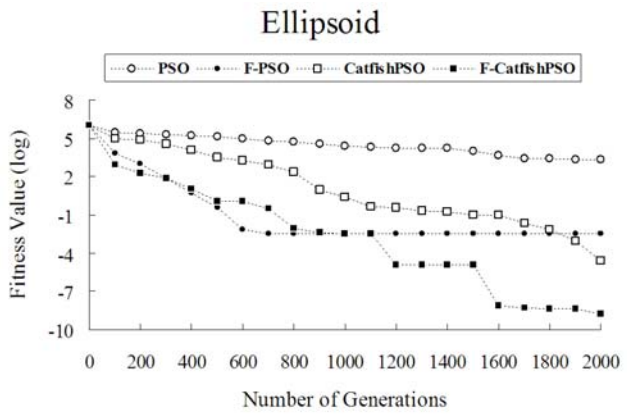


Figure 9. Ellipsoid function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

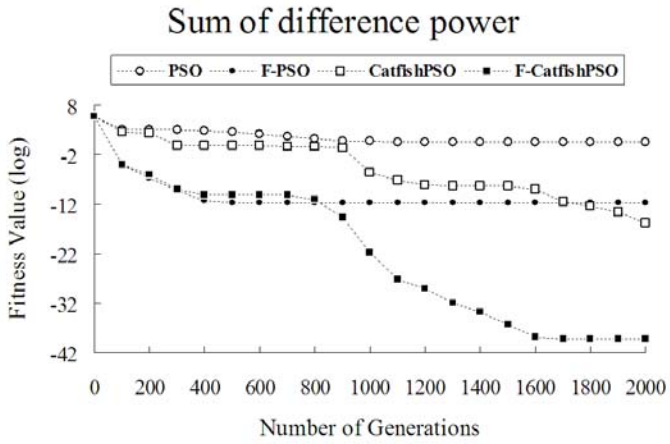


Figure 10. ***Sum of difference power*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

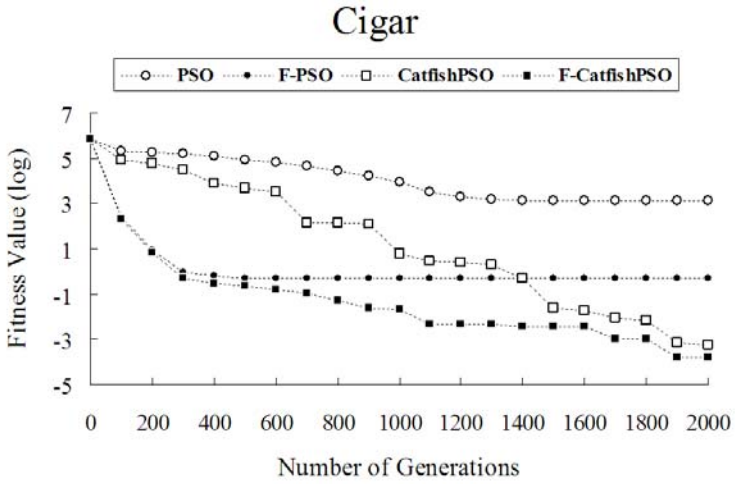


Figure 11. ***Cigar*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

 ISSN 1925-4040 E-ISSN 1925-4059

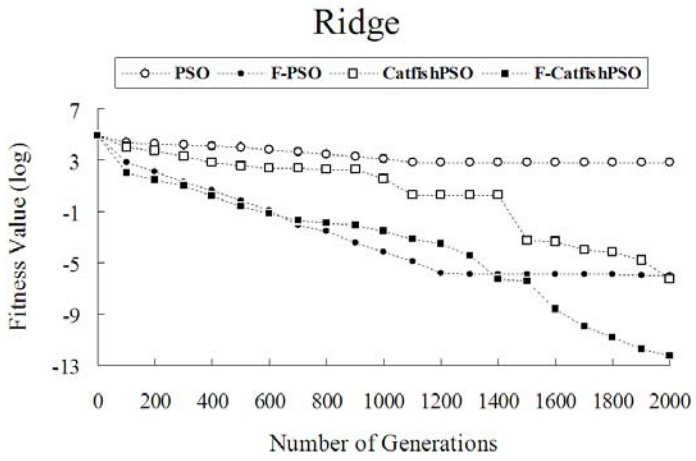


Figure 12. ***Ridge*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

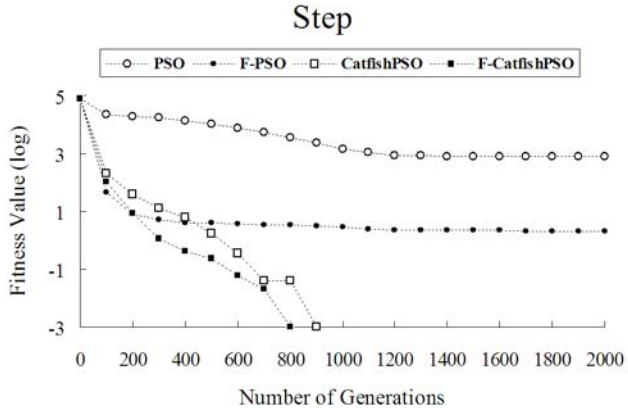


Figure 13. ***Step*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

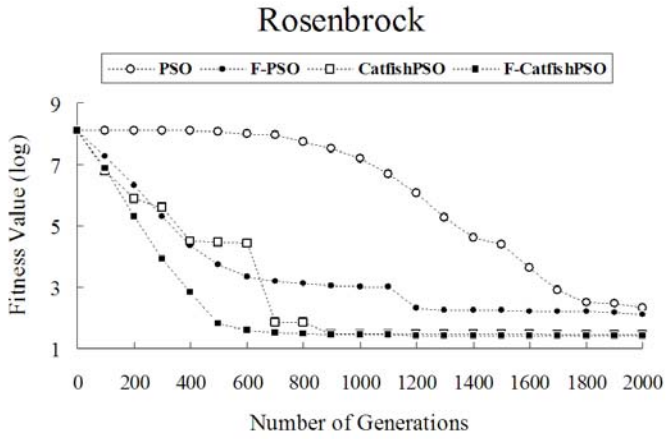


Figure 14. ***Rosenbrock*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

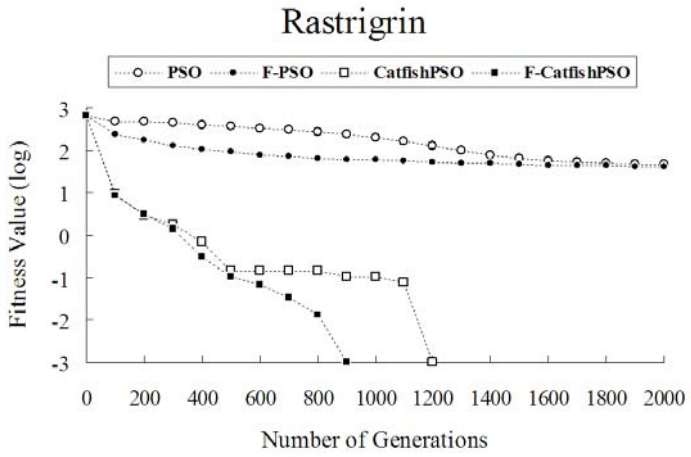


Figure 15. ***Rastrigrin*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

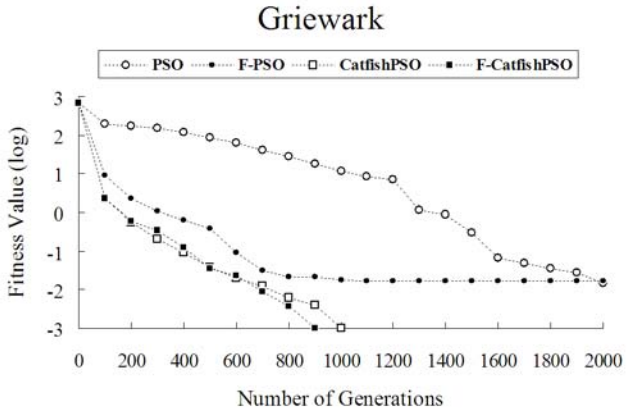


Figure 16. ***Griewark*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

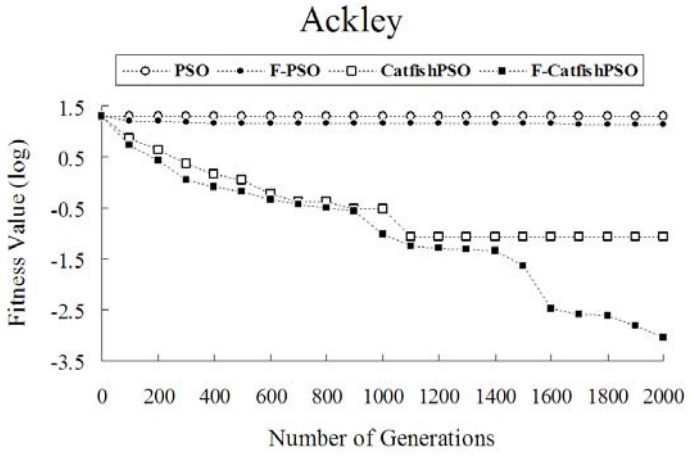


Figure 17. ***Ackley*** function for PSO, F-PSO, CatfishPSO and F-CatfishPSO

ISSN 1925-4040 E-ISSN 1925-4059

Table 4. Mean function value for *Ellipsoid* function (Unimodal)

Pop.	Dim	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	8.94E-22±6.01E-21	**2.65E-71±1.74E-70**	8.09E-26±5.72E-25	2.20E-70±1.56E-69
	20	1500	0	2200.00±4184.520	4.22E-05±2.00E-04	8.64E-09±6.09E-08	**9.02E-11±4.55E-10**
	30	2000	0	2200.00±4184.520	3.38E-03±9.48E-03	2.46E-05±8.94E-05	**1.67E-09±1.18E-08**
40	10	1000	0	400.00±1979.487	4.16E-89±2.87E-88	6.92E-54±3.48E-53	**6.57E-91±4.58E-90**
	20	1500	0	1800.00±1880.879	3.31E-56±1.90E-55	4.33E-20±3.05E-19	**1.67E-62±1.15E-61**
	30	2000	0	2800.00±4535.574	8.44E-11±5.97E-10	5.07E-10±2.39E-09	**5.28E-45±3.41E-44**
80	10	1000	0	4.86E-31±1.30E-30	9.2E-107±5.8E-106	6.21E-73±4.39E-72	**8.8E-107±5.8E-106**
	20	1500	0	2400.00±4314.191	2.98E-70±2.06E-69	5.71E-33±4.03E-32	**1.33E-74±9.41E-74**
	30	2000	0	3200.00±4712.121	1.68E-53±1.14E-52	1.77E-13±1.25E-12	**3.37E-58±2.37E-57**
160	10	1000	0	1.11E-34±2.18E-34	8.1E-123±4.9E-122	2.3E-151±1.1E-150	**7.5E-123±4.9E-122**
	20	1500	0	1200.00±3282.607	1.45E-86±6.47E-86	7.34E-41±5.19E-40	**3.82E-92±1.84E-91**
	30	2000	0	3800.00±4903.144	3.22E-68±1.20E-67	1.41E-18±9.95E-18	**1.72E-74±8.32E-74**
Average				1666.667±2998.087	2.85E-04±8.07E-04	2.05E-06±7.45E-06	**1.46E-10±1.02E-09**
p-value				0.798	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	Yes
Rank				0	1	2	**3**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 5. Mean function value for *Sum of difference Power* function (Unimodal)

Pop.	Dim	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	0.060±00.424	6.98E-12±4.16E-11	1.50E-24±1.06E-23	**4.24E-46±2.91E-45**
	20	1500	0	0.660±01.944	1.40E-09±5.55E-09	8.35E-14±5.90E-13	**1.85E-45±1.27E-44**
	30	2000	0	2.820±11.570	1.73E-12±9.68E-12	3.54E-16±2.50E-15	**4.83E-40±3.41E-39**
40	10	1000	0	1.03E-29±6.84E-29	7.39E-29±5.22E-28	5.21E-47±3.68E-46	**3.11E-56±2.20E-55**
	20	1500	0	0.060±0.424	1.12E-15±6.67E-15	8.43E-24±5.94E-23	**1.84E-62±1.30E-61**
	30	2000	0	0.480±1.529	3.20E-10±2.26E-09	4.87E-20±3.42E-19	**8.44E-52±5.13E-51**
80	10	1000	0	4.44E-34±1.50E-33	6.0E-100±4.20E-99	1.02E-69±7.23E-69	**7.4E-103±3.2E-102**
	20	1500	0	8.74E-28±3.56E-27	5.01E-49±3.54E-48	1.79E-31±1.27E-30	**9.70E-92±5.64E-91**
	30	2000	0	0.960±4.040	4.46E-74±2.98E-73	3.10E-21±2.16E-20	**2.67E-76±1.33E-75**
160	10	1000	0	2.05E-36±6.69E-36	2.44E-109±1.6E-108	2.06E-67±1.33E-66	**2.40E-109±1.6E-108**
	20	1500	0	0.120±0.594	7.4E-105±3.7E-104	5.40E-44±2.78E-43	**4.3E-105±3.0E-104**

(Table 5continued on page 164)

Table 5. (continued)

Pop.	Dim	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
	30	2000	0	0.120±0.594	7.73E-82±4.65E-81	1.11E-25±7.83E-25	**3.99E-84±2.82E-83**
Average				0.440±1.760	1.44E-10±6.56E-10	6.98E-15±4.94E-14	**4.02E-41±2.85E-40**
p-value				0.646	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	Yes
Rank				0	1	2	**3**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 6. Mean function value for *Cigar* function (Unimodal)

Pop.	Dim	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	600.00±2398.979	2.04E-19±1.44E-18	**0.000±0.000**	2.15E-61±1.52E-60
	20	1500	0	1600.00±3703.280	0.110±0.726	2.51E-09±1.57E-08	**4.74E-10±3.35E-09**
	30	2000	0	1400.00±3505.098	0.500±1.816	5.89E-04±4.00E-03	**1.56E-04±9.62E-04**
40	10	1000	0	200.00±1414.214	1.82E-82±1.28E-81	**0.000±0.000**	1.81E-82±1.28E-81
	20	1500	0	1200.00±3282.607	1.33E-56±4.64E-56	2.83E-12±2.00E-11	**1.60E-56±5.93E-56**
	30	2000	0	2400.00±4314.191	**7.60E-10±5.37E-09**	2.68E-06±1.89E-05	8.12E-09±5.74E-08
80	10	1000	0	1200.00±3282.607	7.27E-99±4.37E-98	**0.000±0.000**	7.27E-99±4.37E-98
	20	1500	0	2400.00±4314.191	3.19E-70±1.47E-69	3.80E-22±2.68E-21	**2.06E-70±1.35E-69**
	30	2000	0	2600.00±4430.875	**5.65E-57±1.92E-56**	8.72E-10±6.16E-09	7.59E-57±2.91E-56
160	10	1000	0	200.00±1414.214	7.8E-113±5.5E-112	**0.000±0.000**	7.8E-113±5.5E-112
	20	1500	0	2600.00±4430.875	5.84E-86±3.12E-85	1.88E-39±1.33E-38	**5.76E-86±3.12E-85**
	30	2000	0	3000.00±4629.100	1.20E-70±6.40E-70	1.99E-15±1.26E-14	**1.89E-71±4.79E-71**
Average				1616.667±3426.686	5.09E-02±2.12E-01	4.93E-05±3.35E-04	**1.30E-05±8.02E-05**
p-value				0.760	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	No
Rank				0	1	2	2

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 7. Mean function value for *Ridge* function (Unimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	2.98E-20±8.10E-20	2.62E-24±1.85E-23	**0.000±0.000**	2.29E-66±1.39E-65
	20	1500	0	2.23E-11±1.09E-10	4.19E-08±2.92E-07	2.14E-12±2.84E-10	**0.000±0.000**
	30	2000	0	600.00±2398.979	1.01E-06±3.89E-06	5.75E-07±5.12E-05	**5.38E-13±3.53E-12**
40	10	1000	0	6.67E-25±1.54E-24	1.43E-71±1.01E-70	**0.000±0.000**	3.08E-85±1.46E-84
	20	1500	0	5.83E-15±1.40E-14	2.51E-46±1.77E-45	1.96E-15±2.51E-13	**4.00E-57±2.83E-56**
	30	2000	0	5.88E-11±9.83E-11	1.58E-23±8.02E-23	1.73E-10±1.98E-08	**0.000±0.000**
80	10	1000	0	3.86E-28±1.29E-27	2.54E-88±1.79E-87	**0.000±0.000**	2.48E-97±1.30E-96
	20	1500	0	2.55E-18±4.08E-18	3.86E-61±1.94E-60	1.76E-30±1.99E-28	**1.69E-76±1.19E-75**
	30	2000	0	1.98E-13±4.67E-13	3.65E-43±2.07E-42	1.23E-18±1.68E-16	**0.000±0.000**
160	10	1000	0	6.83E-32±2.18E-31	1.3E-103±9.5E-103	**0.000±0.000**	5.9E-113±4.1E-112
	20	1500	0	2.66E-21±6.33E-21	1.70E-72±8.25E-72	3.08E-39±4.36E-37	**4.65E-88±3.26E-87**
	30	2000	0	4.90E-16±1.10E-15	1.94E-56±1.27E-55	1.20E-20±1.70E-18	**0.000±0.000**
Average				50.00±0199.915	8.77E-08±3.49E-07	4.79E-08±4.27E-06	**4.49E-14±2.95E-13**
p-value				0.646	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	No	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	Yes
Rank				0	1	1	**3**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 8. Mean function value for *Step* function (Unimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	20	1500	0	**0.000±0.000**	0.140±0.405	**0.000±0.000**	**0.000±0.000**
	30	2000	0	800.204±2740.478	2.060±4.787	**0.000±0.000**	**0.000±0.000**
40	10	1000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	20	1500	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	30	2000	0	0.040±0.198	0.440±1.417	**0.000±0.000**	**0.000±0.000**
80	10	1000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	20	1500	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	30	2000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
160	10	1000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
	20	1500	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**

(Table 8 continued on page 166)

Table 8. (continued)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
	30	2000	0	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**	**0.000±0.000**
Average				66.687±228.390	0.220±0.551	**0.000±0.000**	**0.000±0.000**
p-value				0.655	**0**	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	Yes	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	No
Rank				0	1	2	2

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 9. Mean function value for *Rosenbrock* function (Unimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	117.427±268.636	52.880±162.146	5.957±0.532	**4.610±1.071**
	20	1500	0	226.368±316.326	132.127±231.694	16.237±0.416	**14.905±1.388**
	30	2000	0	206.789±262.795	128.186±191.861	26.376±0.388	**24.945±1.674**
40	10	1000	0	41.325±081.814	52.785±154.554	5.360±0.421	**4.056±1.215**
	20	1500	0	150.224±278.776	81.703±216.202	16.033±0.448	**13.276±0.578**
	30	2000	0	179.492±269.172	109.077±171.627	26.326±0.441	**23.441±0.887**
80	10	1000	0	31.540±035.831	17.620±038.041	5.198±0.442	**3.476±1.552**
	20	1500	0	70.342±154.499	70.546±157.172	15.698±0.398	**12.940±0.705**
	30	2000	0	276.720±393.668	82.744±160.830	26.332±0.605	**23.145±0.823**
160	10	1000	0	30.616±061.285	19.193±033.259	4.642±0.460	**2.694±0.977**
	20	1500	0	78.152±157.556	63.380±160.068	15.580±0.556	**12.479±0.662**
	30	2000	0	201.813±314.810	62.256±064.559	25.919±0.466	**22.952±1.011**
Average				134.234±216.264	72.708±145.168	15.805±0.464	**13.577±1.045**
p-value				0.756	0.574	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	No	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	Yes
Rank				0	0	2	**3**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 10. Mean function value for *Rastrigrin* function (Multimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	5.441±02.587	4.439±02.157	**0.000±0.000**	**0.000±0.000**
	20	1500	0	23.189±07.497	19.673±06.144	**0.000±0.000**	**0.000±0.000**
	30	2000	0	46.752±15.403	41.028±13.328	**0.000±0.000**	**0.000±0.000**
40	10	1000	0	3.662±02.093	2.707±01.621	**0.000±0.000**	**0.000±0.000**
	20	1500	0	17.558±05.723	15.749±05.402	**0.000±0.000**	**0.000±0.000**
	30	2000	0	37.837±09.537	30.642±10.903	**0.000±0.000**	**0.000±0.000**
80	10	1000	0	2.089±01.176	1.931±00.993	**0.000±0.000**	**0.000±0.000**
	20	1500	0	12.362±03.943	11.000±04.707	**0.000±0.000**	**0.000±0.000**
	30	2000	0	28.504±06.871	24.433±07.091	**0.000±0.000**	**0.000±0.000**
160	10	1000	0	1.194±01.025	1.194±00.964	**0.000±0.000**	**0.000±0.000**
	20	1500	0	8.894±02.755	6.917±02.430	**0.000±0.000**	**0.000±0.000**
	30	2000	0	24.097±06.949	19.859±06.631	**0.000±0.000**	**0.000±0.000**
Average				17.632±05.463	14.964±05.198	**0.000±0.000**	**0.000±0.000**
p-value				1	0.996	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	No	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	No
Rank				0	0	2	2

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 11. Mean function value for *Griewark* function (Multimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	0.099±0.050	0.081±0.052	**0.000±0.000**	**0.000±0.000**
	20	1500	0	0.034±0.027	0.032±0.034	**0.000±0.000**	**0.000±0.000**
	30	2000	0	0.015±0.019	0.017±0.017	**0.000±0.000**	**0.000±0.000**
40	10	1000	0	0.089±0.036	0.060±0.034	**0.000±0.000**	**0.000±0.000**
	20	1500	0	0.028±0.024	0.025±0.025	**0.000±0.000**	**0.000±0.000**
	30	2000	0	0.009±0.008	0.013±0.018	**0.000±0.000**	**0.000±0.000**
80	10	1000	0	0.068±0.037	0.068±0.040	**0.000±0.000**	**0.000±0.000**
	20	1500	0	0.037±0.033	0.024±0.027	**0.000±0.000**	**0.000±0.000**
	30	2000	0	0.013±0.013	0.012±0.014	**0.000±0.000**	**0.000±0.000**
160	10	1000	0	0.065±0.025	0.058±0.026	**0.000±0.000**	**0.000±0.000**
	20	1500	0	0.033±0.026	0.028±0.021	**0.000±0.000**	**0.000±0.000**

(Table 11 continued page 168)

Table 11. (continued)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
	30	2000	0	0.012±0.014	0.010±0.011	**0.000±0.000**	**0.000±0.000**
Average				0.042±0.026	0.036±0.027	**0.000±0.000**	**0.000±0.000**
p-value				0.958	0.890	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	No	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	No
Rank				0	0	2	2

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

Table 12. Mean function value for *Ackley* function (Multimodal)

Pop.	Dim.	Gen.	Optimal	PSO [8]	F-PSO [3]	CatfishPSO [15]	F-CatfishPSO
20	10	1000	0	19.999±2.22E-06	9.600±10.093	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	20	1500	0	19.999±5.52E-12	12.901±09.550	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	30	2000	0	19.999±2.05E-06	14.009±08.915	8.54E-02±5.46E-01	**8.84E-04±6.22E-03**
40	10	1000	0	19.599±0002.828	12.399±09.806	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	20	1500	0	19.999±3.57E-05	11.181±10.012	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	30	2000	0	19.999±2.74E-05	15.985±08.074	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
80	10	1000	0	19.999±1.29E-11	11.199±10.028	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	20	1500	0	19.999±9.61E-06	12.794±09.693	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	30	2000	0	19.999±5.42E-12	14.732±08.824	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
160	10	1000	0	19.999±5.80E-05	12.399±09.806	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	20	1500	0	19.999±2.32E-05	11.626±09.941	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
	30	2000	0	19.999±6.52E-12	12.773±09.678	**8.88E-16±9.96E-32**	**8.88E-16±9.96E-32**
Average				19.996±2.36E-01	12.633±09.535	7.12E-03±4.55E-02	**7.37E-05±5.18E-04**
p-value				1	0.826	**0**	**0**
Significant (p-value <α=0.1)							
F-PSO vs. PSO				No	No	-	-
F-PSO vs. CatfishPSO				-	No	Yes	-
F-PSO vs. F-CatfishPSO				-	No	-	Yes
CatfishPSO vs. PSO				No	-	Yes	-
F-CatfishPSO vs. PSO				No	-	-	Yes
F-CatfishPSO vs. CatfishPSO				-	-	No	Yes
Rank				0	0	2	**3**

Legend: Yes indicates the method is significant improved. No indicates the method is not significant improved. Rank indicates the numbers of significance at α = 0.1 between the tested method and another method by z-test.

ISSN 1925-4040 E-ISSN 1925-4059

4 Conclusion

In this paper, fuzzy CatfishPSO (F-CatfishPSO) is introduced, which adopts a fuzzy system to improve the performance of the CatfishPSO algorithm. In CatfishPSO, catfish particles initialize a new search from extremes of the search space when the gbest value has not made progress for a number of iterations. Better solutions can be found by guiding the whole swarm to more promising regions in the search space. CatfishPSO has the inherent capability to break through local optimal solution. F-CatfishPSO incorporates the advantages of both CatfishPSO and F-PSO, and achieved far better performance than the PSO, F-PSO and CatfishPSO algorithms. This fact was validated by statistical analysis with a z-test at $\alpha = 0.1$ on ten benchmark functions conducted under equal conditions. It can be concluded that the introduction of new individuals into a group (catfish particles) has a significantly positive effect on the entire swarm, and that the fuzzy system effectively improved the solution quality of CatfishPSO. F-CatfishPSO can conceivably be applied to problems in other areas in the future.

Acknowledgements

This work is partly supported by the National Science Council in Taiwan under grant NSC99-2622-E-151-019-CC3, and NSC99-2221-E-151-056.

References

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in IEEE International Conference on Neural Networks, Perth, Australia. 1995; 1942-1948.

[2] PSC, "Particle Swarm Central," 2012.

[3] Y. H. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in IEEE Congress on Evolutionary Computation, Seoul. 2001; 101-106.

[4] Y. Liu, Z. Qin, Z. W. Shi, and J. Lu, "Center particle swarm optimization," Neurocomputing. Jan 2007; 70: 672-679. http://dx.doi.org/10.1016/j.neucom.2006.10.002

[5] P. J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance differences vol. 1447. Berlin: Springer. 1998.

[6] Y. Jiang, T. S. Hu, C. Huang, and X. N. Wu, "An improved particle swarm optimization algorithm," Applied Mathematics and Computation. Oct 1 2007; 193: 231-239. http://dx.doi.org/10.1016/j.amc.2007.03.047

[7] Y. H. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in IEEE Congress on Evolutionary Computation, Anchorage, AK. 1998; 69-73.

[8] Y. H. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in IEEE Congress on Evolutionary Computation, Washington, DC. 1999; 1945-1949.

[9] B. Jiao, Z. G. Lian, and X. S. Gu, "A dynamic inertia weight particle swarm optimization algorithm," Chaos Solitons & Fractals. Aug 2008; 37: 698-705. http://dx.doi.org/10.1016/j.chaos.2006.09.063

[10] X. Yang, J. Yuan, J. Yuan, and M. H., "A modified particle swarm optimizer with dynamic adaptation," Applied Mathematics and Computation. 2007; 189: 1205-1213. http://dx.doi.org/10.1016/j.amc.2006.12.045

[11] B. Liu, L. Wang, Y. H. Jin, F. Tang, and D. X. Huang, "Improved particle swarm optimization combined with chaos," Chaos Solitons & Fractals. Sep 2005; 25: 1261-1271. http://dx.doi.org/10.1016/j.chaos.2004.11.095

[12] C. W. Jiang and E. Bompard, "A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimisation," Mathematics and Computers in Simulation. Feb 3 2005; 68: 57-65. http://dx.doi.org/10.1016/j.matcom.2004.10.003

[13] L. H. Tsoukalas and R. E. Uhrig, Fuzzy and Neural Approaches in Engineering vol. 1. New York, NY, USA John Wiley & Sons, Inc., 1996.

[14] Y. H. Shi, R. Eberhart, and Y. B. Chen, "Implementation of evolutionary fuzzy systems," Ieee Transactions on Fuzzy Systems. 1999; 7: 109-119. http://dx.doi.org/10.1109/91.755393

[15] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Catfish particle swarm optimization," in IEEE Swarm Intelligence Symposium, St. Louis, Missouri. 2008; 1-5. http://dx.doi.org/10.1109/SIS.2008.4668277

 ISSN 1925-4040 E-ISSN 1925-4059