Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

# Knowledge-based artificial fish-swarm algorithm

**Ying Wu*, Xiao-Zhi Gao**, and Kai Zenger****

**Control Theory and Engineering Department, Harbin Institute of Technology Harbin, 150001, China (e-mail: wying_hit@hotmail.com)*
***Department of Automation and Systems Technology, Aalto University School of Electrical Engineering Espoo, P.O.BOX 15500, FI-00076 Aalto, Finland (e-mail: gao@cc.hut.fi, kai.zenger@aalto.fi)*

**Abstract:** In this paper, a knowledge-based Artificial Fish-Swarm (AFA) optimization algorithm with crossover, CAFAC, is proposed to enhance the optimization efficiency and combat the blindness of the search of the AFA. In our CAFAC, the crossover operator is first explored. The knowledge in the Culture Algorithm (CA) is next utilized to guide the evolution of the AFA. Both the normative knowledge and situational knowledge is used to direct the step size as well as direction of the evolution in the AFA. Ten high-dimensional and multi-peak functions are employed to investigate this new algorithm. Numerical simulation results demonstrate that it can indeed outperform the original AFA.

*Keywords:* Artificial Fish-Swarm Algorithm (AFA); Crossover operator; Culture Algorithm (CA).

## 1. INTRODUCTION

The Artificial Fish-Swarm Algorithm (AFA) is a simulation behaviour and population based optimization method, which was firstly developed by Li Xiao-lei in 2002. (Li, *et al.,* 2002). The AFA can search for the global optimum effectively, and has an adaptive ability for search space. The AFA individual behaviour is to hunt for the local optimum. Therefore, avoiding individual premature becomes indeed difficult. In this case, artificial fish can be stuck into local optima when dealing with multi-modal optimization problems. To improve the global convergence of the AFA, we embed the crossover operator into the process of the AFA, and apply culture algorithms to guide the evolution of artificial fish.

The Cultural Algorithm (CA) was proposed by Reynolds in 1994, which has been utilized in various evolutionary algorithms (Reynolds, 1994; Chung and Reynolds, 1996; Reynolds and Chung, 1997; Coelho and Mariani, 2006; Wu, et al., 2010a, b).

In this paper, we add the crossover operation into the AFA, and a fusion of the CA and AFA with crossover, CAFAC, is proposed to overcome the drawback of the blind search of the regular AFA. A total of four versions of the CAFAC algorithm are explored.

The rest of this paper is organized as follows. Section 2 briefly introduces the background of the AFA and CA. Section 3 discusses the underlying principle of the four versions of CAFAC. In Section 4, ten test functions are used to investigate the behaviour of our CAFAC and compare it with the original AFA.

## 2. ARTIFICIAL FISH-SWARM ALGORITHM AND CULTURAL ALGORITHMS

### *2.1 The Artificial Fish-Swarm Algorithm*

In lakes and seas, the fish can find the areas with the most nutriment. The basic idea of the AFA is to imitate the fish behaviours, such as praying, swarming and chasing. Suppose that the problem under consideration has *D*-dimensions. Initialize the swarm with *N* artificial fish. The state of one artificial fish can be formulated as: $X_i = (x_{i1}, \ldots, x_{iD}) \quad for\, i = 1, \ldots, N$ , where $X_i$ represents the target variable for the problem under consideration. $y = f(X_i)$ stands for the food concentration of the artificial fish currently, where it is the objective function. The parameters in the AFA can be depicted as in Table 1:

**Table 1. Parameters in AFA**

| | |
|---|---|
| $d_{ij} = \|X_j - X_i\|$ | distance between $X_i$ and $X_j$ |
| Visual | visual distance of the artificial fish individual |
| step | size of the movement of artificial fish |
| $\delta$ | crowd factor of the artificial fish |

The basic behaviours of the artificial fish are explained as follows:
(1). Preying: suppose the current state of the artificial fish is $X_i$, the artificial fish selects a state $X_j$ randomly within the visual distance, such as $X_j = X_i + rand(0,1) \times visual$ . If $f(X_j) < f(X_i)$ the artificial fish moves from

$X_i$ towards $X_j$, which means that $X_i^t \to X_i^{t+1}$. The formulation can be explained as follows:

$$X_i^{t+1} = X_i^t + rand(0,1) \times step \times \frac{X_j^t - X_i^t}{\left\| X_j^t - X_i^t \right\|} . \quad (1)$$

If $f(X_j) > f(X_i)$, the artificial fish selects another state randomly again. If the artificial fish cannot meet the requirement in a given time, it moves one step randomly as:

$$X_i^{t+1} = X_i^t + rand(0,1) \times step . \quad (2)$$

(2). Swarming: suppose the current state of the artificial fish is $X_i$, and $nf$ is the number of its fellows within the visual distance, which is equal to the number of elements in the set of $B = \{X_j \mid d_{ij} \le Visual\}$. If $nf \neq 0$, which means the set $B$ is not empty, let $X_{centre} = \sum_{j=1}^{nf} X_j / nf$ and $y_{center} = f(X_{center})$ stand for the fitness of the centre position. If $nf \times y_{center} < \delta \times y_i$, this area is not crowded. If $y_{center} < y_i$, the artificial fish moves one step towards the centre position:

$$X_i^{t+1} = X_i^t + rand(0,1) \times step \times \frac{X_c^t - X_i^t}{\left\| X_c^t - X_i^t \right\|} . \quad (3)$$

Otherwise, it executes the behaviour of preying.

(3). Chasing: suppose the current state of the artificial fish is $X_i$, and $X_{\min}$ stands for the best artificial fish individual within $X_i$'s visual distance. $nf$ is the number of $X_{\min}$'s fellows within the visual distance. $y_{\min} = f(X_{\min})$, if $y_{\min} < y_i$ and $nf \times y_{\min} < \delta \times y_i$, the artificial fish moves one step to $X_{\min}$:

$$X_i^{t+1} = X_i^t + rand(0,1) \times step \times \frac{X_{\min}^t - X_i^t}{\left\| X_{\min}^t - X_i^t \right\|} . \quad (4)$$

Otherwise, it executes the behaviour of preying.

### *2.2 Culture Algorithms*

As proposed by Reynolds, the CA is composed of the population space, belief space, and communication protocol (Chung, 1997). The framework of the CA can be given in Fig. 1.

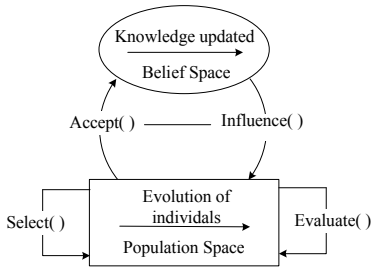


Fig. 1. The Framework of Culture Algorithms

## 3. NOVEL KNOWLEDGE-BASED AFA WITH CROSSOVER

The AFA has many interesting features, such as tolerance of initial values and parameters settings. However, due to the random step length and random behaviour, the artificial fish converges slowly at the mature stage of the algorithm. Thus, the optimization precision usually cannot be high enough, which result in the blindness of search and poor ability of maintaining the balance of exploration and exploitation. To combat with these shortcomings, a novel cultured AFA with crossover, CAFAC, is proposed and studied in this paper to enhance the optimization performance of the original AFA, especially the precision of the AFA.

We first add the crossover operation to the AFA in order to change the state of the artificial fish such that it can search for the possible solution in more diverse areas. When the artificial fish hunts for food, it generally moves to a better direction. The exploitation of the crossover operator can help the artificial fish not only jump out the blindness search but also inherits the advantage of its parents. Next, inspired by the principle of the CA, we apply the normative knowledge and situational knowledge stored in belief space (Reynolds and Peng, 2004) in the CA into the AFA. Here, the fish swarm is regarded as the population space, where the domain knowledge is extracted from. Hence, the domain knowledge is formed and stored in belief space so as to model and impact the evolution of the population at iteration. Note that in this paper, the optimization problems under consideration are all static and constraint-free. In the four versions of the CAFAC, we use the situational knowledge and normative knowledge to guide the direction and step size of the evolution procedure. Both of them can be depicted as follows.

### *3.1 Structures of Belief Space in CAFAC*

The situational knowledge provides a set of exemplars or best individuals, which are available for the interpretation of specific individual experience (Chung, 1997). Here, the situational exemplar set consists of only the best particle found so far, that is, at $t$ iteration, $S = < S^t \mid S^t = \{s_1^t, s_2^t, \ldots, s_n^t\} >$, In other words, it can be initialized with the best particle in the initial fish swarm, and updated by the following function:

$$s_j^{t+1} = \begin{cases} X_{gbest,j}^{t+1} & if \quad f(X_{gbest}^{t+1}) < f(s^t) \\ s_j^t & otherwise \end{cases} , \quad (5)$$

where $X_{gbest}^{t+1}$ denotes the best artificial fish individual in the swarm at generation $t+1$.

The normative knowledge describes the feasible solution space of the optimization problems (Chung, 1997). It is a set of information for each variable, and is given as:

$$N = \langle I, U, L, D \rangle , \quad (6)$$

where $U$, $L$ and $D$ are $n$ -dimensional vectors, and $I=\{x \mid l \le x \le u\}$, $n$ is the number of the variables, $l_j$ and $u_j$ are the lower and upper bounds for the $j^{th}$ variable, respectively, $L_j$ and $U_j$ are the values of the fitness function associated with the bound $l_j$ and $u_j$. Generally, $l_j$ and $u_j$ are initialized with the lower and upper bounds of individuals. $L_j$ and $U_j$ are usually initialized with positive infinity. The normative knowledge is updated as follows:

$$l_j^{t+1}=\begin{cases} x_{i,j} & if\ x_{i,j} \le l_j^t\ \ or\ f(\mathrm{X}_i) < L_j^t \\ l_j^t & otherwise \end{cases}$$
$$u_j^{t+1}=\begin{cases} x_{k,j} & if\ x_{k,j} \ge u_j^t\ or\ f(X_k) < U_j^t \\ u_j^t & otherwise \end{cases}, \quad (7)$$

$$L_j^{t+1}=\begin{cases} f(X_i) & if\ x_{i,j} \le l_j^t\ or\ f(X_i) < L_j^t \\ L_j^t & otherwise \end{cases}$$
$$U_j^{t+1}=\begin{cases} f(X_k) & if\ x_{k,j} \ge u_j^t\ or\ f(X_k) < U_j^t \\ U_j^t & otherwise \end{cases}, \quad (8)$$

where the $i^{th}$ individual affects the lower bound for variable $j$, and the $k^{th}$ individual affects the upper bound for variable $j$. Note, $t$ denotes the current generation of the belief space.

### *3.2 Acceptance Function in CAFAC*

The acceptance function determines which individuals and their performances can have impact on the knowledge in the belief space. The number of the individuals accepted for the update of the belief space is obtained according to the following function (Reynolds and Chung, 1994):

$$f_a(N,t)=N\cdot\beta+\lfloor N\cdot\beta/t \rfloor, \quad (9)$$

where, $N$ is the size of the swarm, $t$ is the iteration number, and $\beta=0.2$.

### *3.3 Influence Functions in CAFAC*

The belief space can influence the evolution in the population space in three different ways：

- Determining the step size of the evolution.
- Determining the direction of the evolution.
- Determining the visual distance of the AFA.

More precisely, if the normative knowledge is used to determine the step size of the evolution and visual distance in the AFA, our knowledge-based AFA is named as CAFAC (Ns). The influence function for the CAFAC is defined as in Tables 2, 3, and 4.

If the situational knowledge is used to guide the direction of the evolution, our knowledge-based AFA is named as CAFAC (Sd).

If the normative knowledge guides the step size and visual distance and the situational knowledge is used to determine the direction of the evolution, respectively, our knowledge-based AFA is named as CAFAC (Ns+Sd).

If the normative knowledge is used to determine the step size and direction of the evolution and visual distance, our knowledge-based AFA is named as CAFAC (Ns+Nd).

**Table 2. Influence Function for Swarming**

| | |
|---|---|
| swarming (Ns) | $x_{ik}^{t+1}=x_{ik}^t+\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}$ |
| swarming (Sd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^t+\left\lvert\frac{rand(0,1)\times step\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t<s_k^t \\ x_{ik}^t-\left\lvert\frac{rand(0,1)\times step\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t>s_k^t \\ x_{ik}^t+\frac{rand(0,1)\times step\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|} & otherwise \end{cases}$ |
| swarming (NsSd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^t+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t<s_k^t \\ x_{ik}^t-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t>s_k^t \\ x_{ik}^t+\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|} & otherwise \end{cases}$ |
| swarming (NsNd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^t+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t<l_k^t \\ x_{ik}^t-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|}\right\rvert & if\ x_{ik}^t>u_k^t \\ x_{ik}^t+\frac{size(I_k)\times rand(0,1)\times(x_{ck}^t-x_{ik}^t)}{\left\|X_c^t-X_i^t\right\|} & otherwise \end{cases}$ |

**Table 3. Influence Function for Preying**

| | |
|---|---|
| selection operator | $X_i^{t+1}=X_i^t+rand(0,1)\times size(I_i)$ |
| $i \le try\_num$ | |
| preying(Ns) | $x_{ik}^{t+1}=x_{ik}^t+\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^t)}{\left\|X_i^{t+1}-X_i^t\right\|}$ |
| preying(Sd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^t+\left\lvert\frac{rand(0,1)\times step\times(x_{ik}^{t+1}-x_{ik}^t)}{\left\|X_i^{t+1}-X_i^t\right\|}\right\rvert & if\ x_{ik}^t<s_k^t \\ x_{ik}^t-\left\lvert\frac{rand(0,1)\times step\times(x_{ik}^{t+1}-x_{ik}^t)}{\left\|X_i^{t+1}-X_i^t\right\|}\right\rvert & if\ x_{ik}^t>s_k^t \\ x_{ik}^t+\frac{rand(0,1)\times step\times(x_{ik}^{t+1}-x_{ik}^t)}{\left\|X_i^{t+1}-X_i^t\right\|} & otherwise \end{cases}$ |

| | |
|---|---|
| preying (NsSd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}<s \\ x_{ik}^{t}-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}>s \\ x_{ik}^{t}+\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\ otherwis \end{cases}$ |
| preying (NsNd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}<l \\ x_{ik}^{t}-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}>u \\ x_{ik}^{t}+\frac{size(I_k)\times rand(0,1)\times(x_{ik}^{t+1}-x_{ik}^{t})}{\left\lVert X_i^{t+1}-X_i^{t}\right\rVert}\ otherwis \end{cases}$ |
| $i>try\_num$ | |
| preying(Ns) | $x_{ik}^{t+1}=x_{ik}^{t}+size(I_k)\times rand(0,1)$ |
| preying (Sd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\lvert rand(0,1)\times step\rvert if\ x_{ik}^{t}<s_k^{t} \\ x_{ik}^{t}-\lvert rand(0,1)\times step\rvert if\ x_{ik}^{t}>s_k^{t} \\ x_{ik}^{t}+rand(0,1)\times step\ \ otherwise \end{cases}$ |
| preying (NsSd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\lvert size(I_k)\times rand(0,1)\rvert if\ x_{ik}^{t}<s_k^{t} \\ x_{ik}^{t}-\lvert size(I_k)\times rand(0,1)\rvert if\ x_{ik}^{t}>s_k^{t} \\ x_{ik}^{t}+size(I_k)\times rand(0,1)\ \ otherwise \end{cases}$ |
| preying (NsNd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\lvert size(I_k)\times rand(0,1)\rvert if\ x_{ik}^{t}<l_k^{t} \\ x_{ik}^{t}-\lvert size(I_k)\times rand(0,1)\rvert if\ x_{ik}^{t}>u_k^{t} \\ x_{ik}^{t}+size(I_k)\times rand(0,1)\ \ otherwise \end{cases}$ |

**Table 4. Influence Function for Chasing**

| | |
|---|---|
| Chasing(Ns) | $x_{ik}^{t+1}=x_{ik}^{t}+\frac{size(I_k)\times rand(0,1)\times(x_{muink}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}$ |
| Chasing (Sd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\left\lvert\frac{rand(0,1)\times step\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}<s_k^{t} \\ x_{ik}^{t}-\left\lvert\frac{rand(0,1)\times step\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}>s_k^{t} \\ x_{ik}^{t}+\frac{rand(0,1)\times step\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\ otherwise \end{cases}$ |
| Chasing (NsSd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}<s_k^{t} \\ x_{ik}^{t}-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}>s_k^{t} \\ x_{ik}^{t}+\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\ otherwise \end{cases}$ |
| Chasing (NsNd) | $x_{ik}^{t+1}=\begin{cases} x_{ik}^{t}+\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}<l_k^{t} \\ x_{ik}^{t}-\left\lvert\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\right\rvert if\ x_{ik}^{t}>u_k^{t} \\ x_{ik}^{t}+\frac{size(I_k)\times rand(0,1)\times(x_{\min k}^{t}-x_{ik}^{t})}{\left\lVert X_{\min}^{t}-X_i^{t}\right\rVert}\ otherwise \end{cases}$ |

In Table 2, 3, and 4, $size(I_k)=u_k-l_k$ is the size of the belief interval, which is decided by the normative knowledge for the $k^{th}$ variable. The rand(0,1) is a random number uniformly distributed within interval of (0,1). The other parameters are given in 2.1 and 3.1

### *3.4 Crossover Operator*

A criterion is set up to judge whether the algorithm falls into the local optimum:

$$\left|\frac{f(X_i^{t})-f(X_i^{t-1})}{f(X_i^{t-1})}\right|<0.1. \qquad (10)$$

When the criterion is satisfied, the crossover operator (Wang and Cao, 2002) will be applied to the $i^{th}$ artificial fish $X_i(i=1,\ldots,N)$:

$$x_i^{'}=x_{r1}+\alpha\times(x_{r2}-x_{r1}), \qquad (11)$$

where $x_{r2},x_{r1}$ are two individuals selected randomly, $r_1,r_2$ are two integers, which are generated randomly in the interval of [1,N] and $r_1\neq r_2\neq i$. $\alpha$ is random number uniformly distributed in interval of $[-d,1+d]$, and $d$ is chosen as 0.25. Evaluate the child $x_i^{'}$, and replace the individual $x_i$ with the child, if $x_i^{'}$ performs better.

In summary, the basic procedure of our CAFAC algorithm can be described as:

(1). Set all the values for the parameters, and initialize the $N$ artificial fish in the search ranges with random positions.

(2)Evaluate all the artificial fishes using the fitness function y, and initialize the belief space.

(3). For each $i^{th}$ artificial fish, simulate the preying pattern, swarming, and chasing patterns separately, and select the best child fish. If the child is better, replace the $i^{th}$ artificial fish with the child.

(4). Update the belief space.

(5). If the crossover criterion is satisfied, apply the crossover operator to the $i^{th}$ artificial fish from Step 3.

(6). Return back to Step 3 until the termination criterion is satisfied.

## 4. SIMULATIONS

In this section, a total of 15 nonlinear functions are used to investigate the optimization capability of our CAFAC. All these functions are multi-modal functions, as given in Table 5, and they have a lot of local optima around the global optima. As mentioned above, there are a total of four versions of our CAFAC: CAFAC(Ns), CAFAC(Sd), CAFAC(NsSd) and CAFAC(NsNd). The parameters of the four CAFAC variants are chosen as: N=40, D=30, visual=250, step=225, $\delta = 24$, and try_num=10.

**Table 5. Test Functions**

| |
|---|
| $f^1_{\text{Ackley}}(x) = -20\exp(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}) - \exp(\frac{1}{30}\sum_{i=1}^{30}\cos 2\pi x_i) + 20 + e$ |
| $f^2_{\text{CM}}(x) = -0.1\sum_{i=1}^{n}\cos(5\pi x_i) + \sum_{i=1}^{n} x_i^2$ |
| $f^3_{\text{DeJongF4}}(x) = \sum_{i=1}^{30} i \cdot x_i^4$ |
| $f^4_{\text{Expfun}}(x) = \exp(-0.5\sum_{i=1}^{n} x_i^2)$ |
| $f^5_{\text{Griewank}}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30}\cos(\frac{x_i}{\sqrt{i}}) + 1$ |
| $f^6_{\text{Hyperelliptic}}(x) = \sum_{i=1}^{20} i^2 x_i^2$ |
| $f^7_{\text{LM1}}(x) = (\pi / n)(10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + \sin^2(\pi y_{i+1})] + (y_n - 1)^2)$ |
| $f^8_{\text{LM2}}(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)])$ |
| $f^9_{\text{Neumaier3}}(x) = \sum_{i=1}^{n}(x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$ |
| $f^{10}_{\text{Rastrigrin}}(x) = \sum_{i=1}^{30}(x_i^2 - 10\cos(2\pi x_i) + 10)$ |
| $f^{11}_{\text{Rosenbrock}}(x) = \sum_{i=1}^{29}(100(x_{i+1} - x_i^2) + (x_i - 1)^2)$ |
| $f^{12}_{Sal}(x) = 1 - \cos(2\pi\sqrt{\sum_{i=1}^{30} x_i^2}) + 0.1\sqrt{\sum_{i=1}^{30} x_i^2}$ |
| $f^{13}_{\text{Schwefel}}(x) = 418.9829 \cdot 10 + \sum_{i=1}^{10}(-x_i \sin(\sqrt{\lvert x_i \rvert}))$ |
| $f^{14}_{\text{Schaffer}}(x) = \sum_{i=1}^{30}(x_i^2 + x_{i+1}^2)^{0.25}\left\{\left[\sin 50(x_i^2 + x_{i+1}^2)^{0.1}\right]^2 + 1\right\}$ |
| $f^{15}_{\text{Sphere}}(x) = \sum_{i=1}^{30} x_i^2$ |

The four versions of our CAFAC are compared with the original AFA, and the comparison results are provided in Table 6. It shows that the proposed CAFAC behaves much better than the AFA for almost all the functions except for the Sal function and Schwefel function. For the DeJongf4 function, Expfun function, Griewank function, Sphere function and Neumaier3 function, the CAFAC (NsSd) can find the global optimum successfully. For the other functions, the four versions of CAFAC yield significant performance improvement in optimization compared with AFA. Moreover, we can observe from Table 6 that the CAFAC (NsSd) is superior to the other three CAFAC variants. From Table 6 we can see that, in most cases, the best optimization performance can be achieved by using the knowledge in the belief space to determine both the step size and direction of the evolution in the AFA.

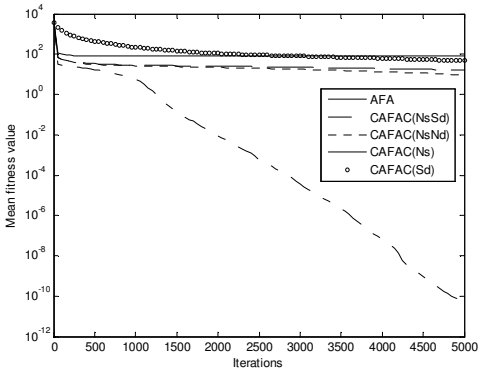


Fig. 2. Optimization results of Rosenbrock function

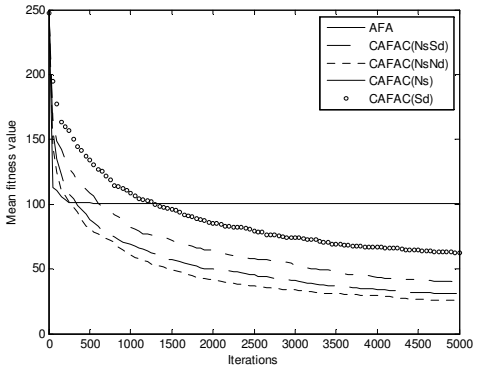


Fig. 3. Optimization results of Rastrigin function

Figures 2-3 illustrate the comparison of convergence performance among the four versions of the CAFAC. In order to obtain a clear picture, e.g., in Fig. 2, a logarithmic (base 10) scale is used for the vertical axis and we draw the points every 50 iterations. The mean best value is the average over 10 separate trials for each of these algorithms. From the above figures, it can be discovered that the original AFA has a fast convergence speed at the beginning, but after that it is trapped into a local optimum, and cannot jump out of that.

## 5. CONCLUSIONS

As we know that the AFA is good at the global search at the beginning of its optimization stage. However, when dealing with high dimension problems, the artificial fish can be

easily stuck into the blindness hunting. In order to improve the performance of the regular AFA, the cultural framework is introduced and utilized in this paper. The knowledge, namely situational knowledge and normative knowledge, is stored in the belief space to guide the evolution of the AFA. In addition, the crossover operator is utilized to increase the diversity of the fish population. From the simulation results, we can find out that the knowledge-based AFA does perform much better than the original algorithm. Therefore, the knowledge in the cultural framework can be viewed as a effective and directive term in the evolution of the AFA.

## ACKNOWLEDGEMENTS

This research work was funded by the Academy of Finland under Grants 135225 and 127299 and the NSFC under Grant No. 60874084.

## REFERENCES

Chung, C.J. and Reynolds, R.G. (1996). A Testbed for solving optimization problems using cultural algorithms. *Evolutionary Programming V: Proceedings of the Fifth Annual Conference on Evolutionary Programming,* San Diego, CA, pp. 225-236.

Chung, C.J. (1997). *Knowledge-based approaches to self-adaptation in cultural algorithms*. Ph.D. Dissertation, Wanyne State University.

Coelho, L.S. and Mariani, V.C. (2006). An efficient particle swarm optimization approach based on cultural algorithm applied to mechanical design. *IEEE Congress on Evolutionary Computation*, Veancouver, Canada, pp. 1099-1104.

Li, X.L, Z.J., Shao, Z.J., and Qian, J.X. (2002). An optimizting method based on autonomous animates: fish-swarm algorithm. *Systems Engineering-theory & Practice*, 11, pp. 32-38.

Reynolds, R.G. (1994). An introduction to cultural algorithms. *Proceedings of the 3rd annual Conference on Evolution Programming*, San Diego, CA, pp. 131-139.

Reynolds, R.G. and Chung, C. J. (1997). Knowledge-based self-adaptation in evolutionary programming using cultural algorithms. *IEEE International Conference on Evolutionary Computation*, Indianapolis, Hoosier State, pp. 71-76.

Reynolds, R.G. and Peng, B. (2004). Cultural algorithms: modeling of how cultures learn to solve problems. *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, Boca Raton, FL, pp. 166-172.

Wang, X.P. and Cao, L.M. (2002). *Genetic algorithm: theory, application and software implementation.* Xi'an JiaoTong University Press, Xi'an.

Wu, Y. , Gao, X.Z., Huang, X.L., and Zenger, K. (2010a). A cultural particle swarm optimization algorithm. *Proceedings of the Sixth International Conference on Natrual Computation*, Yantai, China, pp.2505-2509.

Wu, Y. , Gao, X.Z., Huang, X.L., and Zenger, K. (2010b). A hybrid optimization method of particle swarm optimization and cultural algorithm. *Proceedings of the Sixth International Conference on Natrual Computation*, Yantai, China,pp.2515-2519.

**Table 6. Function Optimization Performance Comparison**

| Functions | Global optimum | CAFAC (NsSd) | CAFAC (NsNd) | CAFAC (Sd) | CAFAC (Ns) | AFA |
|---|---|---|---|---|---|---|
| Ackley | 0 | $7.9936\times10^{-15}$ | $1.0027\times10^{-8}$ | 2.2365 | $1.6214\times10^{-8}$ | 1.1898 |
| CM | -3 | -2.3497 | -2.6453 | -0.8500 | -2.7488 | -0.5258 |
| DeJongf4 | 0 | 0 | $3.1247\times10^{-22}$ | $8.6335\times10^{-4}$ | $2.0326\times10^{-19}$ | 0.2141 |
| Expfun | 1 | 1.000000 | 1.0000000 | 1.000062 | 1.000004 | 1.411369 |
| Griewank | 0 | 0 | $1.5588\times10^{-14}$ | $3.4002\times10^{-5}$ | $2.2271\times10^{-14}$ | 0.0236 |
| Hyperelliptic | 0 | $2.4506\times10^{-98}$ | $1.5812\times10^{-8}$ | 0.0972 | $2.0207\times10^{-7}$ | 74.0210 |
| LM1 | 0 | $1.7771\times10^{-32}$ | $2.6380\times10^{-12}$ | 0.0105 | $1.5996\times10^{-11}$ | 0.3498 |
| LM2 | 0 | $1.3498\times10^{-32}$ | $1.1150\times10^{-12}$ | 0.4738 | $3.4690\times10^{-11}$ | 0.1593 |
| Neumaier3 | -4930 | -4930.0000 | -4929.8467 | -4929.4759 | -4929.9993 | -1143.6718 |
| Rastrigin | 0 | 39.8072 | 25.7239 | 62.1831 | 30.8390 | 96.2181 |
| Rosenbrock | 0 | $5.7642\times10^{-11}$ | 9.5057 | 46.6662 | 16.3719 | 96.7020 |
| Sal | 0 | 0.2099 | 0.1899 | 0.3799 | 0.1999 | 0.1199 |
| Schwefel | 0 | $4.8565\times10^{3}$ | $1.4793\times10^{3}$ | $3.4749\times10^{3}$ | $1.3070\times10^{3}$ | 0.3500 |
| Schaffer | 0 | 12.7555 | 14.9599 | 27.7994 | 14.5671 | 18.1480 |
| Sphere | 0 | 0 | $2.8001\times10^{-13}$ | $6.9378\times10^{-4}$ | $1.0066\times10^{-12}$ | 0.6160 |