An Efficient Function Optimization Algorithm based on Culture Evolution

Xuesong Yan[1,2], Qinghua Wu[3,4], Can Zhang[1], Wei Chen[1], Wenjing Luo[1] and Wei Li[1]

**[1] School of Computer Science, China University of Geosciences
Wuhan, Hubei 430074, China**

**[2] Department of Computer Science, University of Central Arkansas
Conway, AR 72035, USA**

**[3] Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology
Wuhan, Hubei 430073, China**

**[4] School of Computer Science and Engineering, Wuhan Institute of Technology
Wuhan, Hubei 430073, China**

Abstract

Optimization problems arise in many real-world applications. Cultural Algorithms are a class of computational models derived from observing the cultural evolution process in nature, compared with genetic algorithm the cultural algorithms have high convergence speed. Aiming at the disadvantages of basic cultural algorithms like being trapped easily into a local optimum, this paper improves the basic cultural algorithms and proposes a new algorithm to solve the overcomes of the basic cultural algorithms. The new algorithm keeps not only the fast convergence speed characteristic of basic cultural algorithms, but effectively improves the capability of global searching as well. For the case studies, this means has proved to be efficient and the experiment results show that the new means have got the better results.

Keywords: *Function Optimization, Cultural Algorithm, genetic algorithm, Culture, Population.*

1. Introduction

Candidate solutions to some problems are not simply deemed correct or incorrect but are instead rated in terms of quality and finding the candidate solution with the highest quality is known as optimization. Optimization problems arise in many real-world scenarios. Take for example the spreading of manure on a cornfield, where depending on the species of grain, the soil quality, expected amount of rain, sunshine and so on, we wish to find the amount and composition of fertilizer that maximizes the crop, while still being within the bounds imposed by environmental law.

Several challenges arise in optimization. First is the nature of the problem to be optimized which may have several local optima the optimizer can get stuck in, the problem may be discontinuous, candidate solutions may yield different fitness values when evaluated at different times, and there may be constraints as to what candidate solutions are feasible as actual solutions to the real-world problem. Furthermore, the large number of candidate solutions to an optimization problem makes it intractable to consider all candidate solutions in turn, which is the only way to be completely sure that the global optimum has been found. This difficulty grows much worse with increasing dimensionality, which is frequently called the curse of dimensionality, a name that is attributed to Bellman, see for example [1]. This phenomenon can be understood by first considering an n-dimensional binary search-space. Here, adding another dimension to the problem means a doubling of the number of candidate solutions. So the number of candidate solutions grows exponentially with increasing dimensionality. The same principle holds for continuous or real-valued search-spaces, only it is now the volume of the search-space that grows exponentially with increasing dimensionality. In either case it is therefore of great interest to find optimization methods which not only perform well in few dimensions, but do not require an exponential number of fitness evaluations as the dimensionality grows. Preferably such optimization methods have a linear relationship between the dimensionality of the problem and the number of candidate solutions they must evaluate in order to achieve satisfactory results, that is, optimization methods should ideally have linear time-complexity O(n) in the dimensionality n of the problem to be optimized.

Another challenge in optimization arises from how much or how little is known about the problem at hand. For example, if the optimization problem is given by a simple formula then it may be possible to derive the inverse of that formula and thus find its optimum. Other families of problems have had specialized methods developed to optimize them efficiently. But when nothing is known about the optimization problem at hand, then the No Free Lunch (NFL) set of theorems by Wolpert and Macready states that any one optimization method will be as likely as any other to find a satisfactory solution [2]. This is especially important in deciding what performance goals one should have when designing new optimization methods, and whether one should attempt to devise the ultimate optimization method which will adapt to all problems and perform well. According to the NFL theorems such an optimization method does not exist and the focus of this thesis will therefore be on the opposite: Simple optimization methods that perform well for a range of problems of interest.

Cultural Algorithms (CA) proposed by Reynolds in 1994 [3]. Cultural algorithm is in-depth analysis of the superiority of the original evolution theory on the basis of drawing on the social (cultural) evolution theory in the social sciences and has achieved broad consensus on the research results, and proposed a new algorithm. Cultural algorithm is used to solve complex calculations of the new global optimization search algorithms, cultural algorithms in the optimization of the complex functions of its superior performance.

2. Cultural Algorithm

Evolutionary computation (EC) [4,5] methods have been successful in solving many diverse problem in search and optimization due to the unbiased nature of their operations which can still perform well in situation with little or no domain knowledge. However, there can be considerable improvement in their performance when problem specific knowledge is used to bias the problem solving process in order to identify patterns in their performance environment. These patterns are used to promote more instances of desirable candidates or to reduce the number of less desirable candidates in the population. In either case, this can afford the system an opportunity to reach the desired solution more quickly.

Adaptive evolutionary computation takes place when an EC system is able to incorporate such information into its representation and operators in order to facilitate the pruning and promoting activities mentioned above. Some research works have shown that self-adaptation can take place on several levels within a system such as the population level, the individual level, and the component level. At the population level, aspects of the system parameters that control all elements of the population can be modified. At the individual level, aspects of the system that control the action of specific individual can be modified. If the individual is specified as s collection of components then component level adaptation is possible. This involves the adaptation of parameters that control the operation of one or more components that make up an individual.

In human societies, culture can be a vehicle for the storage of information in a form that is independent of the individual or individuals that generated and are potentially accessible to all members of the society. As such culture is useful in guiding the problem solving activities and social interaction of individuals in the population. This allows self-adaptive information as well as other knowledge to be stored and manipulated separately from the individuals in the social population. This provides a systematic way of utilizing self-adaptive knowledge to direct the evolution of a social population. Thus, cultural systems are viewed as a dual inheritance system where, at each time step, knowledge at both the population level and the level of acquired beliefs is transmitted to the next generation. This acquired knowledge is viewed to act as beacons by which to guide individuals towards perceived good solutions to problems and away from less desirable ones at a given time step. Cultural Algorithms in order to model the evolution of cultural systems based upon principles of human social evolution taken from the social science literature.

2.1 Basic Cultural Algorithm

Cultural Algorithms are a class of computational models derived from observing the cultural evolution process in nature [3, 6, 7]. In this algorithm, individuals are first evaluated using a performance function. The performance information represents the problem-solving experience of an individual. An acceptance function determines which individuals in the current population are able to impact, or to be voted to contribute, to the current beliefs. The experience of these selected individual is used to adjust the current group beliefs. These group beliefs are then used to guide and influence the evolution of the population at the next step, where parameters for self-adaptation can be determined from the belief space. Information that is stored in the belief space can pertain to any of the lower levels, e.g. population, individual, or component. As a result, the belief space can be used to control self-adaptation at any or all of these levels. The cultural algorithm is a dual inheritance system with evolution taking place at the population level and at the belief level.

The two components interact through a communications protocol. The protocol determines the set of acceptable individuals that are able to update the belief space. Likewise the protocol determines how the updated beliefs are able to impact and influence the adaptation of the population component.

The Cultural Algorithm is a dual inheritance system that characterizes evolution in human culture at both the macro-evolutionary level, which takes place within the belief space, and at the micro-evolutionary level, which occurs at the population space. CA consists of a social population and a belief space. Experience of individuals selected from the population space by the acceptance function is used to generate problem solving knowledge that resides in the belief space. The belief space stores and manipulates the knowledge acquired from the experience of individuals in the population space. This knowledge can control the evolution of the population component by means of the influence function. As a result, CA can provide an explicit mechanism for global knowledge and a useful framework within which to model self-adaptation in an EC system. The population level component of the cultural algorithm will be Evolutionary Programming (EP). The global knowledge that has been learned by the population will be expressed in terms of both normative and situational knowledge as discussed earlier.

A pseudo-code description of the Cultural Algorithms is described as follows:

```
Framework of cultural algorithm
1    BEGIN
2      t=0;
3    Initialize population P(t);
4    Initialize belief space B(t);
5    Repeat
6    Evaluate P(t) ;
7    Update(B(t), accept(P(t))) ;
8    Generate (P(t), influence(B(t)) ;
9    Select P(t) from P(t-1) ;
10   t+=1 ;
11   Until (termination condition)
12   END
```

In this algorithm, first the belief space and the population space are initialized. Then, the algorithm will repeat processing for each generation until a termination condition is achieved. Individuals are evaluated using the performance function. The two levels of Cultural Algorithm communicate through the acceptance function and the influence function. The acceptance function determines which individuals from the current population are selected to impact the belief space. The selected individuals' experiences are generalized and applied to adjust the current beliefs in the belief space via the update function. The new beliefs can then be used to guide and influence the evolutionary process for the next generation.
Cultural algorithms as described above consist of three components. First, there is a population component that contains the social population to be evolved and the mechanisms for its evaluation, reproduction, and modification. Second there is a belief space that represents the bias that has been acquired by the population during its problem-solving process. The third component is the communications protocol that is used to determine the interaction between the population and their beliefs.

2.2 Design Basic Cultural Algorithm

In the basic cultural algorithms, the belief space uses the {S,N} structure represented [15]. The formal syntax for the belief space, B, used in this study is: $B = S|N|\langle S, N\rangle$, where S denotes structure for situational knowledge and N denotes structures for normative knowledge. The definition above means the belief space can consist of situational knowledge only, normative knowledge only, or both. The situational knowledge S is represented formally as a pair wise structure: $S = \langle < E_1, E_2, ..., E_e >, adjust_E(e)\rangle$, where E_i represent an ith best exemplar individual in the evolution history. There can be e best exemplars in S as s set that constitutes the situational knowledge. Each exemplar individual has n parameters and a performance value. $adjust_E(e)$ is the belief space operator applied to update e number of exemplar individuals in S. The normative knowledge, N, a set of interval information for each of the n parameters is defined formally as 4-tuple: $N = \langle I_j, L_j, U_j, adjust_N\rangle, j = 1,2,...,n$, where I_j denotes the closed interval of variable j, that is a continuous set of real numbers x represented as a ordered number pair: $I_j = [l_j, u_j] = \{x \mid l_j \le x \le u_j, x \in R\}$. l_j (lower bound) and u_j (upper bound) are initialized by the give domain values. L_j represents the performance score of the lower bound l_j for parameter j. U_j represents the performance score of the upper bound u_j for parameter j.

For the update function, we defined like this: S={st}, select the best individual st update the situation knowledge S in belief space. The update process follows the equation (1):

$$s^{t+1} = \begin{cases} x^t_{best} & f(x^t_{best}) < f(s^t) \\ s^t & \text{others} \end{cases} \quad (1)$$

where x^t_{best} denotes tth best individual.

Update the normative knowledge N in belief space uses the equation (2):

$$\begin{aligned} l_i^{t+1} &= \begin{cases} x_{j,i} & x_{j,i} \le l_i^t \text{ or } f(x_j) < L_i^t \\ l_i^t & \text{others} \end{cases} \\ L_i^{t+1} &= \begin{cases} f(x_j) & x_{j,i} \le l_i^t \text{ or } f(x_j) < L_i^t \\ L_i^t & \text{others} \end{cases} \\ u_i^{t+1} &= \begin{cases} x_{j,i} & x_{j,i} \ge u_i^t \text{ or } f(x_j) < U_i^t \\ u_i^t & \text{others} \end{cases} \\ U_i^{t+1} &= \begin{cases} f(x_j) & x_{j,i} \ge u_i^t \text{ or } f(x_j) < U_i^t \\ U_i^t & \text{others} \end{cases} \end{aligned} \quad (2)$$

In basic CA, the knowledge represented in the belief space can be explicitly used to influence the creation of the offspring via an influence function. In our sliding window model, the strategy can be simply described as follows. The first is if a parent is in a promising region, the offspring are created by randomly changing the problem parameters of the parent just a little. In this case, the normative knowledge applies. The offspring $x^{t+1}_{j,i}$, will be created by using this normative knowledge as a beacon to attract the parent $x^t_{j,i}$ to move a copy toward the current sliding window, the influence function defined by the equation (3).

$$x^{t+1}_{j,i} = \begin{cases} x^t_{j,i} + |size(I_i) * N(0,1)| & x^t_{j,i} < l^t_i \\ x^t_{j,i} - |size(I_i) * N(0,1)| & x^t_{j,i} > u^t_i \\ x^t_{j,i} + \lambda_1 * size(I_i) * N(0,1) & others \end{cases} \quad (3)$$

The second is if a parent is in an unpromising region, moving a copy of the parent to a more promising region can be used to create a new offspring. In this case, the constraint knowledge applies. The creation of offspring will be affected by the characteristic of the cells within the sliding window, the influence function defined by the equation (4).

$$x^{t+1}_{j,i} = \begin{cases} x^t_{j,i} + |size(I_i) * N(0,1)| & x^t_{j,i} < S^t_i \\ x^t_{j,i} - |size(I_i) * N(0,1)| & x^t_{j,i} > S^t_i \\ x^t_{j,i} + \lambda_1 * size(I_i) * N(0,1) & others \end{cases} \quad (4)$$

Footnotes should be typed in singled-line spacing at the bottom of the page and column where it is cited. Footnotes should be rare.

2.3 Implementation of Basic CA

Compared with genetic algorithm (GA), the most obvious advantage of CA is that the convergence speed is very high because of the dual inheritance system that characterizes evolution. In order to verify the convergence speed of the CA, we selected four benchmarks function and compared the results with traditional genetic algorithm.

F1: Schaffer function

$$\min f(x_i) = 0.5 - \frac{(\sin^2\sqrt{x_1^2 + x_2^2} - 0.5)}{[1 + 0.001(x_1^2 + x_2^2)]^2}, -100 \le x_i \le 100$$

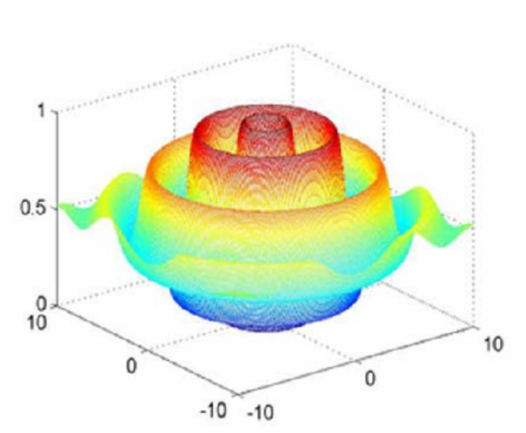

Fig. 1 Schaffer function

In this function the biggest point is in the situation where xi= (0, 0) and the global optimal value is 1.0, the largest in the overall points for the center, and 3.14 for the radius of a circle on the overall situation from numerous major points of the uplift. This function has a strong shock; therefore, it is difficult to find a general method of its global optimal solution.

F2: Shubert function

$$\min f(x,y) = \left\{\sum_{i=1}^{5} i\cos[(i+1)x + i]\right\} \times \left\{\sum_{i=1}^{5} i\cos[(i+1)y + i]\right\},$$
$$x, y \in [-10, 10]$$

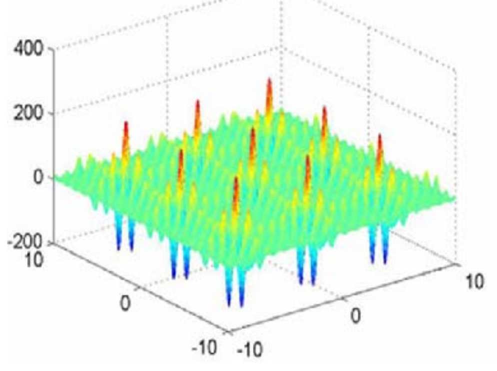

Fig. 2 Shubert function

This function has 760 local minimum and 18 global minimum, the global minimum value is -186.7309.

F3: Hansen function

$$\min f(x,y)=\sum_{i=1}^{5} i\cos((i-1)x+i)\sum_{j=1}^{5} j\cos((j+1)y+j),$$
$$x,y\in[-10,10]$$

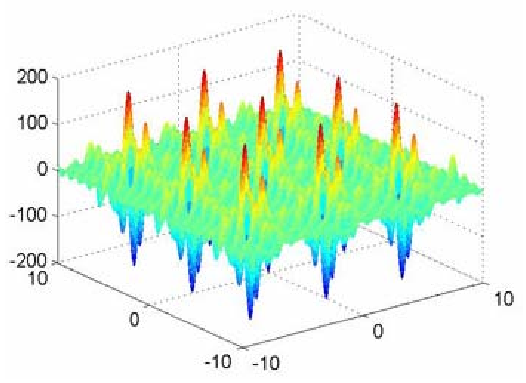

Fig. 3 Hansen function

This function has a global minimum value -176.541793, in the following nine point (-7.589893, -7.708314)、(-7.589893 , -1.425128) 、 (-7.589893 , 4.858057) 、 (-1.306708 , -7.708314) 、 (-1.306708 , -1.425128) 、 (-1.306708 , 4.858057) 、 (4.976478 , -7.708314) 、 (4.976478, -7.708314)、(4.976478, 4.858057) can get this global minimum value, the function has 760 local minimum.

F4: Camel function

$$\min f(x,y)=\left(4-2.1x^2+\frac{x^4}{3}\right)x^2+xy+\left(-4+4y^2\right)y^2,$$
$$x,y\in[-100,100]$$

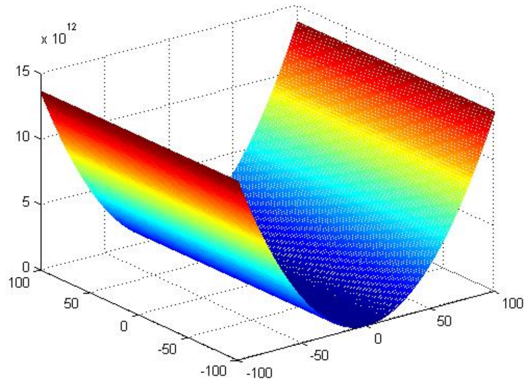

Fig. 4 Camel function

Camel function has 6 local minimum (1.607105, 0.568651) 、 (-1.607105, -0.568651) 、 (1.703607, -0.796084) 、 (-1.703607, 0.796084) 、 (-0.0898,0.7126) and (0.0898,-0.7126), the (-0.0898,0.7126) and (0.0898,-0.7126) are the two global minimums, the value is -1.031628.

Table 1: experiment results comparison (100 runs for each case)

Function	***Algorithm***	***Convergence Times***	***Optimal Solution***
F1	GA	72	1.0000000
	CA	75	1.0000000
F2	GA	75	-186.730909
	CA	80	-186.730909
F3	GA	85	-176.541793
	CA	90	-176.541793
F4	GA	23	-1.031628
	CA	56	-1.031628

In the experiment, each case is repeated for 100 times. Table 1 shows the statistics of our experimental results in terms of accuracy of the best solutions. GA found the known optimal solution to F1 72 times out of 100 runs, found the known optimal solution to F2 75 times out of 100 runs, found the known optimal solution to F3 85 times out of 100 runs, found the known optimal solution to F4 23 times out of 100 runs; CA is efficiency for the four cases: found the known optimal solution to F1 75 times out of 100 runs, found the known optimal solution to F2 80 times out of 100 runs, found the known optimal solution to F3 90 times out of 100 runs and found the known optimal solution to F4 56 times out of 100 runs.

3. Function Optimization Algorithm based on Culture Evolution

In the basic CA, the convergence speed of individuals is fast, but the adjustments of cognition component and social component make individuals search in the belief space. Here a fatal weakness may result from this characteristic. According to influence function and update function, once the best individual in the population is trapped into a local optimum, then will attract other individuals to approach this local optimum gradually, and in the end the whole population will be converged at this position, and the capacity of population jump out of a local optimum is rather weak. The probability of the occurrence is especially high so far for multi-peaks functions, we have test the algorithm for the multi-peaks functions to verify these.

3.1 Orthogonal Initialization

The traditional method of function optimization algorithms is randomly initialized population, that is, generate a series of random numbers in the solution space of the question. In this paper, the new algorithm using the orthogonal initialization [8] in the initialization phase. For the general condition, before seeking out the optimal solution the location of the global optimal solution is impossible to know. For some high-dimensional and multi-mode functions to optimize, the function itself has a lot of poles, and the global optimum location of the function is unknown. If the initial population of chromosomes can be evenly distributed in the feasible solution space, the algorithm can evenly search in the solution space for the global optimum. Orthogonal initialization is to use the orthogonal table has the dispersion and uniformity comparable; the individual will be initialized uniformly dispersed into the search space, so the orthogonal design method can be used to generate uniformly distributed initial population.

3.2 New Influence Function

In the basic CA, the influence function think the population space as global, and think the individuals in the population space as objects, once the best individual in the population is trapped into a local optimum, then the algorithm think it has find the global optimum and the algorithm is convergence. In the improved algorithm, we change some strategy of the influence function. In the new influence function, the best individual is considered as global and the gene of the best individual is considered as object, when the gene of the best individual changed, then the new population will be generated. The process of the operation is described as follows: X is the best individual, x_i is ith variable of X. Using equation (5) do the operation for x_i and X, then generate the new individual $X^{'}$.

$$x_i^{'} = \begin{cases} x_i + |size(I_i) * N(0,1)| & x_{j,i}^t < S_i^t \\ x_i - |size(I_i) * N(0,1)| & x_{j,i}^t > U_i^t \\ x_i + \lambda * size(I_i) * N(0,1) & others \end{cases} \quad (5)$$

For the new individual $X^{'}$, if $f(X^{'}) > f(X)$ then $X = X^{'}$. In this process, the parameter λ is very important, for the value of λ in the improved CA there has three different situations as follows:
1. When the algorithm is convergence, in order to generate the best individual, the value of the λ maybe the fraction between in (0 , 1), then the whole procedure is actually a fine-tuning of the best individual;
2. When the algorithm can not convergence, in order to fast the convergence speed of the algorithm, the value of the λ maybe a large value, best meets $\lambda * size(I_i) >= (u_{global} - j_{global})/2$. In here, the u_{global} is the minimum ceiling of the given function's domain, the j_{global} is the maximum limit of the given function's domain. This value makes a higher degree of change about the individual or a single variable, and then the algorithm can jump out of the local optimum;
3. For some special function, the algorithm is easy trapped into the local optimum, we can combine the above two method to generate the best individual. The detail process as follows: for the generations before the p*generation, the value of the λ is the fraction between in (0 , 1); for the generations after the p*generation, the value of the λ is a large value, best meets $\lambda * size(I_i) >= (u_{global} - j_{global})/2$.

3.3 Elite Selection Mechanism

Genetic algorithm is usually complete the selection operation based on the individual's fitness value, in the mechanism of elite, the population of the front generation mixed with the new population which generate through crossover and mutation operations, in the mixed population select the optimum individuals according to a certain probability. The specific procedure is as follows:
Step1: using crossover and mutation operations for population P1 which size is N then generating the next generation of sub-populations P2;
Step2: The current population P1 and the next generation of sub-populations P2 mixed together form a temporary population;
Step3: Temporary population according to fitness values in descending order, to retain the best N individuals to form new populations P1.

The characteristic of this mechanism is mainly in the following aspects. First is robust, because of using this selection strategy, even when the crossover and mutation operations to produce more inferior individuals, as the results of the majority of individual residues of the original population, does not cause lower the fitness value of the individual. The second is in genetic diversity maintaining, the operation of large populations, you can better maintain the genetic diversity of the population evolution process. Third is in the sorting method, it is good to overcome proportional to adapt to the calculation of scale.

3.4 Empirical Study

In order to verify the improvement of the improved cultural algorithm (ICA), we also use other benchmarks function to test the algorithm's performance and compare the results with basic cultural algorithm. Specific details of the test function see table 1. In the Table 2, n behalf of the dimension number of the function, S behalf of the range of variables, f_{min} behalf of the minimization of the function.
The two algorithms with the same experimental parameters set. Each function in Table 2 is run 50 times with the two algorithms, their experimental results such as Table 3. By analyzing the experimental results we know, in solving function f1, f4 and f7, use the basic CA is easily into local optimum, but use the ICA, convergence soon, and can find better solution, the average fitness and the best fitness was both superior to basic CA. For the function f2, the ICA and CA all can find the global optimal, these two algorithm for this test function is very effective. For function f5, the two algorithms can find the best solutions are the same (see Table 3), and the new algorithm to get the best value of the average is better than CA. In sum, we can see that in solving function f1, f4, f5 and f7, the ICA more efficient than CA, in solving other function, the performance almost same of the two algorithms. In short, this new algorithm has the following value: better global search capability.

4. Conclusions

This paper introduce a new algorithm based on the basic CA algorithm, for the basic CA algorithm the new algorithm has done the following improvements: 1. By introducing a new population initialization method and the new individual selection mechanism, make the individuals within the population space can maintain combined with the best individuals, thus enlarge global searching space and reduce the possibility of individuals to be trapped into a local optimum; 2. By improving the influence function, decreased the possibility of being trapped into a local optimum. Compared with the basic CA algorithm, the new algorithm enlarges the searching space and the complexity is not high. For the empirical studies this new algorithm has proved to be efficient, and the experiments results shown the new algorithm are effective for function optimization.

Acknowledgments

This paper is supported by the Provincial Natural Science Foundation of Hubei (No. 2011CDB334 and 2011CDB346), the Fundamental Research Founds for National University, China University of Geosciences (Wuhan) and National Civil Aerospace Pre-research Project of China.

References

[1] R. Bellman, "Dynamic Programming", Princeton University Press, 1957.
[2] D.H. Wolpert and W.G. Macready, "No free lunch theorems for optimization", IEEE Transactions on Evolutionary Computation, 1(1), 1997, pp.67-82.
[3] R. Reynoids, "An introduction to cultural algorithms", in Proceedings of the 3rd Annual Conference on Evolutionary Programming, Sebald, AX; Fogel, L.J. (Editors), River Edge, NJ, World Scientific Publishing, 1994, pp.13 1-139.
[4] Goldbeg, D. E, "Genetic Algorithms in Search, Optimization and Machine Learning", Reading, Mass.: Addison-Wesley, 1989.
[5] Michalewicz, Z, "Genetic Algorithms + Data Structures = Evolution Programs", 3rd Ed. Berlin: Springer – Verlag, 1996.
[6] CHUNG C, "Knowledge-based approaches to self-adaptation in cultural algorithms", Ph.D. Thesis, Wayne State University, Detroit, Michigan, USA, 1997.
[7] ZHANG Yin, "Cultural algorithm and its application in the portfolio", Master Thesis, Harbin University of Science and Technology, Harbin, China, 2008.
[8] X.S. Yan et.al, "Designing Electronic Circuits Using Cultural Algorithms", Proceedings of Third International Workshop on Advanced Computational Intelligence, 2010, pp.299-303.
[9] X.S. Yan et.al, "Electronic Circuits Optimization Design Based On Cultural Algorithms", International Journal of Information Processing and Management, Vol.2(1), 2011, pp.49-56.
[10] X.S. Yan, Q.H. Wu, "Function Optimization Based on Cultural Algorithms", Journal of Computer and Information Technology, Vol.2, 2012, pp.152-158.
[11] X.S.Yan, Qing Hua Wu et.al; "A New Optimizaiton Algorithm for Function Optimization", Proceedings of the 3rd International Symposium on Intelligence Computation & Applications, 2009, pp. 144-150.

Xuesong Yan associate professor received him B.E. degree in Computer Science and Technology in 2000 and M.E. degree in Computer Application from China University of Geosciences in 2003, received he Ph.D. degree in Computer Software and Theory from Wuhan University in 2006. He is currently with School of Computer Science, China University of Geosciences, Wuhan, China and now as a visiting scholar with Department of Computer Science, University of Central Arkansas, Conway, USA. He research interests include evolutionary computation, data mining and computer application.

Qinghua Wu lecturer received her B.E. degree in Computer Science and Technology in 2000, M.E. degree in Computer Application in 2003 and Ph.D. degree in Earth Exploration and Information Technology Theory from China University of Geosciences in 2011. She is currently with School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China. Her research interests include evolutionary computation, image processing and computer application.

Can Zhang received him B.E. degree in Computer Science and Technology in 2011. He is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Wei Chen received him B.E. degree in Computer Science and Technology in 2012. He is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Wenjing Luo received her B.E. degree in Computer Science and Technology in 2012. She is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Wei Li received her B.E. degree in Computer Science and Technology in 2012. She is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Table 2: Test function

Test Function	n	S	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	(-100,100)	0
$f_2(x) = 6 \cdot \sum_{i=1}^{5} \lfloor x_i \rfloor$	30	(-5.12, 5.12)	0
$f_3(x) = \sum_{i=1}^{n} i \cdot x_i^4 + U(0,1)$	30	(-1.28,1.28)	0
$f_4(x) = \frac{1}{4000} \sum_{i=1}^{n} (x_i - 100)^2 - \prod_{i=1}^{n} \cos(\frac{x_i - 100}{\sqrt{i}}) + 1$	30	(-300.0,300.0)	0
$f_5(x) = -20 \cdot \exp(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi \cdot x_i)) + 20 + e$	30	(-32.0,32.0)	0
$f_6(x) = \sum_{i=1}^{n} 100((x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	30	(-2.048,2.048)	0
$f_7(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	(-500,500)	-12569.5

Table 3: Experiment Results Comparison

Function	*Algorithm*	*Best Value*	*Worst Value*	*Standard deviation*
$f_1(x)$	CA	1495.71	7032.89	201.038
	ICA	4.13731E-29	1.0882E-24	2.28015E-26
$f_2(x)$	CA	0	0	0
	ICA	0	0	0
$f_3(x)$	CA	0.00177094	0.00833963	0.000210055
	ICA	0.00193565	0.0103595	0.000259903
$f_4(x)$	CA	72.5069	123.954	1.52289
	ICA	2.18559E-12	8.63194E-25	0.00177512
$f_5(x)$	CA	-3.19744E-14	4.4229	0.148418
	ICA	-3.19744E-14	1.50229	0.0749509
$f_6(x)$	CA	1.84889E-28	8.63194E-25	1.83991E-26
	ICA	2.55147E-28	1.20401E-23	2.41678E-25
$f_7(x)$	CA	-5038.62	-3233.13	54.0123
	ICA	-9535.19	-8203.56	45.1661