A Novel Hybrid Differential Evolution Algorithm: A Comparative Study on Function Optimization

Ching-Hung Lee, *Member, IAENG*, Pei-Yuan Chung, and Hao-Han Chang

***Abstract*— This paper presents a novel differential evolution algorithm (called CCDE) using self-adaptive scaling factors, chemotaxis by BP technique, and cooperative strategy to enhance the performance of traditional DE algorithm. The CCDE consists of modified mutation strategy which is implemented from each individual and includes the momentum of each individual, the self-adaptive scaling factor strategy which adopts the fuzzy logic system to improve convergence speed, the chemotaxis by BP technique, and optimal learning rate which enhances the local search ability and convergence. The cooperative strategy could generate extra vectors to search the solution. Finally, the CCDE algorithm is demonstrated by a comparative study on function optimization.**

***Index Terms*—Optimization, Differential evolution, Chemotaxis, Cooperative Strategy, back-propagation**

I. INTRODUCTION

IN the past decades, many investigators have proposed many effective approaches to solve the optimization problem, such as genetic algorithm, electromagnetism-like algorithm, particle swarm optimization, etc [2, 4-5, 8, 14]. Evolutionary algorithms (EAs) have received attention for their potential as global optimization techniques [3, 10, 13]. As a method of solving optimization problems, a powerful stochastic global optimization differential evolution (DE) algorithm was proposed [15-16]. DE utilizes mutation and recombination operations as searching mechanisms and selection operators to determine the most promising regions of the search space. It creates new candidate solutions by combining the parent individual and several other individuals by randomly choosing from the same population. A candidate replaces the parent only if it has a better fitness value. This is a rather greedy selection scheme that often outperforms traditional EAs [3, 10].

Although the DE algorithm benefits from the aforementioned advantages, the DE convergence velocity is slow for high dimensional optimization problems. In addition, DE is sensitive to algorithm parameters, such as low precision, and it cannot easily determine the global optimal solution if these parameters are not properly met. Therefore, the selection of the scaling factor F, and CR is very important and difficult. In our experience, the larger value of F can converge quickly, but usually at a local minimum. Inversely, small value of F converges slowly. In literature [19], F was selected by experienced human operators.

In order to improve the performance of DE algorithm, this paper proposes a DE algorithm (called CCDE algorithm) with novel strategies, includes the modified mutation strategy, the self-adaptive scaling factor using fuzzy logic system, chemotaxis by BP technique with adaptive learning rates, and the simple cooperative strategy. The modified mutation strategy consists of a mutation strategy of traditional DE, the best information, and the momentum term. In addition, the momentum term provides the past information, which advances the ability of searching the optimal solution. The adaptive scaling factor F is established by fuzzy logic systems. The generation number and change of fitness value are used to generate changes in F using a fuzzy approach. Subsequently, the chemotaxis by BP technique supplies the algorithm with the searching ability in the neighborhood. In addition, the concept of the chemotaxis is derived from the bacterial foraging algorithm [6, 17]. In the neutral medium, if the bacterium swims up a nutrient gradient, its behavior seeks increasingly favorable environments. We adopt the chemotaxis concept for our CCDE algorithm. Many literatures [1, 11-12] indicate that the cooperative strategy lead to a significant improvement in performance.

This paper is organized as follows. Section II introduces the proposed novel hybrid differential evolution algorithm CCDE. The experimental results of function optimization are introduced in Section III. Finally, the conclusion is given.

II. A NOVEL HYBRID DIFFERENTIAL EVOLUTION ALGORITHM- CCDE

This section introduces the proposed novel hybrid differential evolution algorithm CCDE. Figure 1 shows the flowchart of the CCDE algorithm. The CCDE algorithm is proposed by using (a) the modified mutation strategy which is implemented from each individual and includes the momentum of each individual; (b) the chemotaxis by BP technique with the optimal learning rates to search the better situation in the neighborhood; (c) the simple cooperative strategy to generation extra vectors for searching the solution; and (d) self-adaptive scaling factors to improve the performance. The CCDE algorithm would improve the precision, and enhance the convergence speed. Firstly, the optimization problem is formula as

This work was supported in part by the National Science Council, Taiwan, R.O.C., under contracts NSC-97-2221-E-155-033-MY3.

Ching-Hung Lee is with Department of Electrical Engineering, Yuan-Ze University, Chung-li, Taoyuan 320, Taiwan. (phone: +886-3-4638800, ext: 7119; fax: +886-3-4639355, e-mail: chlee@saturn.yzu.edu.tw).

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Minimize $f(\Theta)$

subject to $\Theta \in S$, $S = \{\Theta \in \Re^D | \theta_d \in \Re, d = 1,...,D\}$

where D is dimension of the problem and; $f(\Theta)$ is the objective function that is minimized. Each vector Θ represents a solution.

CCDE algorithm starts with initial population vectors, which are randomly generated when no preliminary experimental knowledge about the searching space is available, in order to start CCDE algorithm, we must decide the population size P_S, the maximum generation G, and D-dimensional parameters vectors, and we must generate parameters randomly.

The pth vector of the population at generation g is denoted as $\Theta_p(g) = (\theta_{p1}, \theta_{p2}, \cdots, \theta_{pD})$, p=1,…, P_S, g=1,2,…,G.

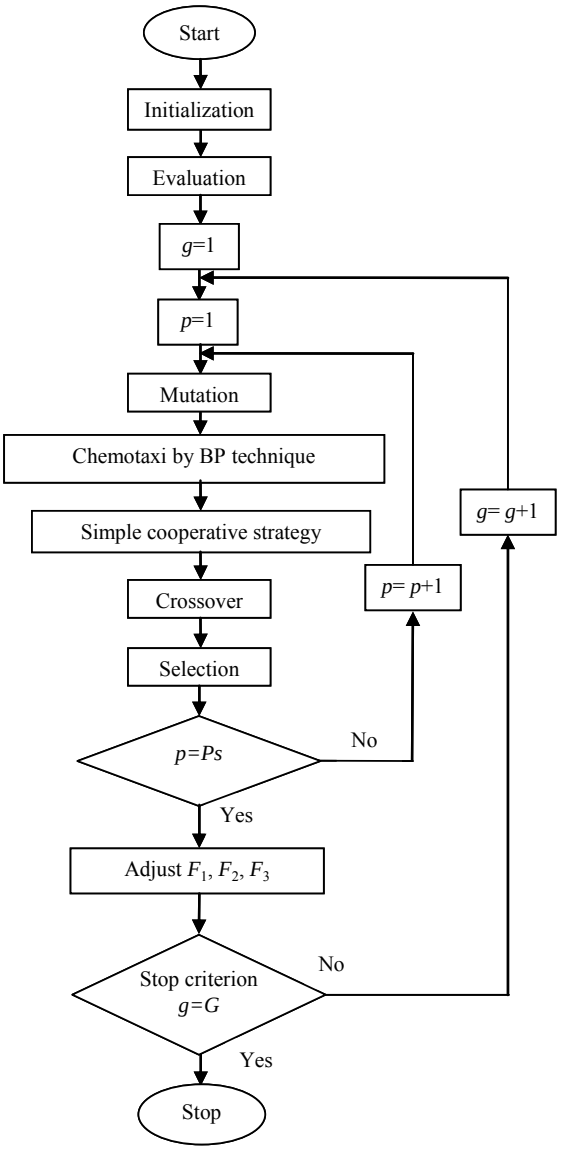

Fig. 1. Flowchart of the proposed CCDE algorithm.

TABLE I: THE 3×3 FUZZY RULES TABLE FOR THE SCALING FACTORS

Δe / Generation	Zero	Small	Big
Small	Increasing	Increasing	Fixing
Medium	Increasing	Fixing	Decreasing
Big	Fixing	Decreasing	Decreasing

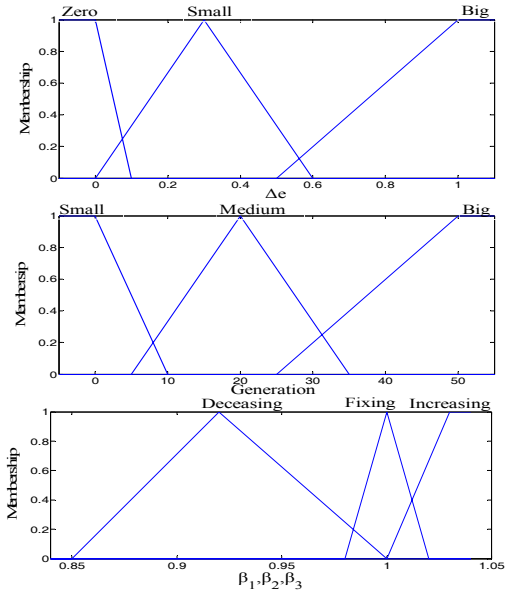

Fig. 2. Membership functions of fuzzy system; (a) input variable-Δ*e*. (b) input variable- generation; (c) output variable- β_1, β_2, and β_3.

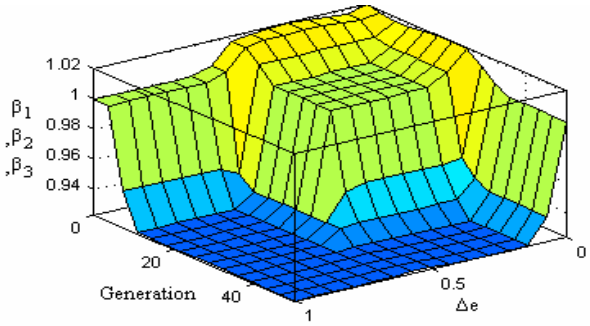

Fig. 3. Membership functions of fuzzy system; (d) The corresponding surface of β_1, β_2, and β_3.

A. *Mutation with Self-adaptive Scaling Factor F*

A differential vector which is composed of the best vector Θ_{best} and each vector $\Theta_p(g)$ and the momentum term $\Theta_p(g) - \Theta_p(g-1)$ to constitute the modified mutation strategy [20], utilizing the modified mutation strategy could maintain the diversity since the agent searches the optimal solution from each vector $\Theta_p(g)$ rather than the best individual Θ_{best}.

By information sharing concept, the differential vector which is composed of the best individual Θ_{best} and each vector $\Theta_p(g)$ could provide a proper direction to the optimal solution. For each population vector $\Theta_p(g)$, p = 1, 2, 3,…, P_S, g=1,2,…,G, the mutation vector is generated as follows:

$$v_p(g+1) = \Theta_p(g) + F_1 \cdot (\Theta_{r_1}(g) - \Theta_{r_2}(g)) + F_2 \cdot (\Theta_{best} - \Theta_p(g)) + F_3 \cdot (\Theta_p(g) - \Theta_p(g-1)) \quad (1)$$

where F_1, F_2, and $F_3 \in [0,1]$ are the scaling factors (or mutation factors), r_1, $r_2 \in \{1,2,\ldots, P_S\}$, and $p \neq r_1 \neq r_2$. The differential vector is the deviation between two randomly selected population vectors $\Theta_{r_1}(g)$ and $\Theta_{r_2}(g)$.

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The scaling factors F_1, F_2, and F_3 are also adjusted by fuzzy logic systems, the inputs of the fuzzy system are changes of fitness value, Δe, and generation number, g, and the corresponding outputs are β_1, β_2 and β_3. The adjustment is shown as follows:

$$F_i = F_i \times \beta_i, \quad i = 1, 2, 3. \tag{2}$$

The 3×3 fuzzy rules table are described in Table I. The corresponding membership functions for inputs Δe, generation, and output β_1, β_2, and β_3 are shown in Fig. 2. (a), 2(b), and 2(c), respectively. The consequence β_1, β_2, and β_3 is deduced from Δe and generation by taking the max-min composition. The center of area (COA) method is used as the defuzzification strategy. Figure 3 shows the corresponding surface of β_1, β_2, and β_3. The fuzzy system is constructed by the following IF-THEN rules

IF Δe is Zero and generation is Small,
THEN β_1, β_2, and β_3 are Increasing.
IF Δe is Small and generation is Small,
THEN β_1, β_2, and β_3 are Increasing.
IF Δe is Big and generation is Small,
THEN β_1, β_2, and β_3 are Fixing.
IF Δe is Zero and generation is Medium,
THEN β_1, β_2, and β_3 are Increasing.
IF Δe is Small and generation is Medium,
THEN β_1, β_2, and β_3 are Fixing.
IF Δe is Big and generation is Medium,
THEN β_1, β_2, and β_3 are Decreasing.
IF Δe is Zero and generation is Big,
THEN β_1, β_2, and β_3 are Fixing.
IF Δe is Small and generation is Big,
THEN β_1, β_2, and β_3 are Decreasing.
IF Δe is Big and generation is Big,
THEN β_1, β_2, and β_3 are Decreasing.

B. Chemotaxis By Back-propagation Technique

The idea of the chemotaxis by BP technique is deduced from bacterial foraging algorithm [6, 17]. Bacterial foraging algorithm is inspired by the pattern exhibited by bacterial foraging behaviors. Bacteria have the tendency to gather in the nutrient-rich areas by an activity called "chemotaxis". If a bacterium finds a favorable environment, it searches a better situation until it cannot find any more. We adopt this concept and BP technique to search the optimal solution in the neighborhood.

The update law of trial vector $v'_p(g+1)$ is represented as

$$\begin{aligned} v'_{pd}(g+1) &= v_{pd}(g+1) + \Delta v_{pd}(g+1) \\ &= v_{pd}(g+1) + \eta_d(g)\left(-\frac{\partial E(v_p(g+1))}{\partial v_{pd}(g+1)}\right), \end{aligned} \tag{3}$$

where $d = 1,2,...,D$ and $\eta_d(g)$ is the learning rate. After we get the trial vector $v'_p(g+1)$, the fitness value of the trial vector $f(v'_p(g+1))$ is compared with the fitness value of the parent vector $f(\Theta_p(g))$. If the trial vector $v'_p(g+1)$ is better than the parent vector $\Theta_p(g)$, the trial vector $v'_p(g+1)$ is going to get into the chemotaxis procedure. Otherwise, the trial vector $v'_p(g+1)$ becomes the parent vector $\Theta_p(g)$, and skip the chemotaxis procedure. The update is expressed as follows:

$$v'_p(g+1) = \begin{cases} v'_p(g+1) & \text{if } f(v'_p(g+1)) < f(\Theta_p(g)) \\ \Theta_p(g) & \text{otherwise.} \end{cases} \tag{4}$$

We implement the back-propagation algorithm to get the chemotaxis vector $v''_p(g+1)$, which is denoted as

$$\begin{aligned} v''_{pd}(g+1) &= v'_{pd}(g+1) + \Delta v'_{pd}(g+1) \\ &= v'_{pd}(g+1) + \eta_d(g)\left(-\frac{\partial E(v'_p(g+1))}{\partial v'_{pd}(g+1)}\right), \end{aligned} \tag{5}$$

where $d = 1,2,...,D$. After we get the chemotaxis vector $v''_p(g+1)$, the fitness value of the chemotaxis vector $f(v''_p(g+1))$ is compared with the fitness value of the trial vector $f(v'_p(g+1))$. If the chemotaxis vector $v''_p(g+1)$ is better than the trial vector $v'_p(g+1)$, the chemotaxis vector $v''_p(g+1)$ replaces the trial vector $v'_p(g+1)$. Repeat the chemotaxis procedure until the chemotaxis vector $v''_p(g+1)$ is not better than the trial vector $v'_p(g+1)$. The update is expressed as follows:

$$v'_p(g+1) = \begin{cases} v''_p(g+1) & \text{if } f(v''_p(g+1)) < f(v'_p(g+1)) \\ v'_p(g+1) & \text{otherwise.} \end{cases} \tag{6}$$

Moreover, the learning rates play an important role in the chemotaxis by BP technique. If the proper learning rates are chosen, the convergence is guaranteed. According to the literature [7], we can obtained the optimal learning rate from the Lyapunov stability analysis is

$$\eta^*(g+1) = \frac{1}{D\left(\frac{\partial E(v'_p(g))}{\partial v'_{pd}(g)}\right)^2} \tag{7}$$

where D is the dimension.

C. The Simple Cooperative Strategy

Although competition among individual vectors usually improves their performance, much greater improvements can be obtained through cooperation. Many researchers apply the cooperative approach to improve their algorithm. Various cooperative approaches have been proposed and analyzed.

A typical example is the cooperative co-evolutionary genetic algorithm which is proposed [12]. The cooperative strategy is that the solution vector is split into its constituent components and assigned to multiple populations [1]. In this configuration, each population is then optimized a single component of the solution vector (i.e. a one-dimensional optimization problem). We adopt the spirit of cooperative strategy into CCDE algorithm.

The simple cooperative strategy in CCDE algorithm is to generate extra individuals by the populations. On the other hand, the solution vector is split into populations areas. Now, we obtain P_s copies of the solution vector. The P_s *jth* in *j*th copy are replaced with the P_s *jth* in the parent $\Theta_p(g)$

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

(j=1,2,…, P_s). For example, we use the trial vector $v'_p(g+1)$ (p=1,2, …, P_s) which is obtained by the chemotaxis by BP technique to generate the cooperative vector C_j (j=1,2,…, Ps). The cooperative vector C_j is generated by replacing the trial vector $v'_p(g+1)$ with the parent $\Theta_p(g)$. The cooperative process is described in Fig. 4 and Fig. 5. If the cooperative vector C_j is better than the trial vector $v'_p(g+1)$, the cooperative vector C_j replaces the trial vector $v'_p(g+1)$. The trial vector $v'_p(g+1)$ is updated as follows:

$$v'_p(g+1)=\begin{cases} C_j & \text{if } f(C_j)\le f(v'_p(g+1)) \\ v'_p(g+1) & \text{otherwise} \end{cases}, \quad (8)$$

where j=1,2,..., P_s .

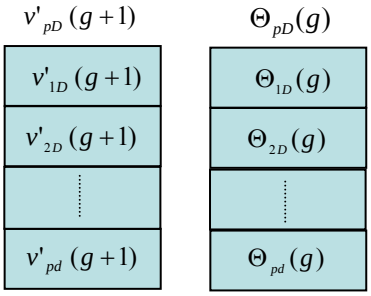

Fig. 4. Representation of trial vector $v'_p(g+1)$ and $\Theta_p(g)$.

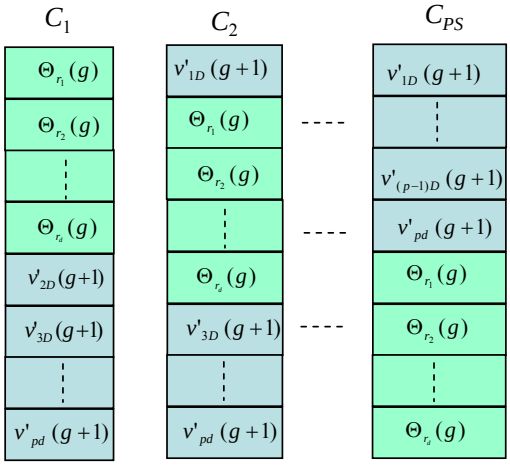

Fig. 5. The simple cooperative process for D dimension.

D. Crossover

The crossover vector $u_p(g+1)$ is chosen by the trial vector $v'_p(g+1)$ and the mutation vector $v_p(g+1)$ rather than the parent $\Theta_p(g)$. We give a corresponding number rand(d) for each parameter of the mutation vector $v_p(g+1)$ where rand(d) $\in$ [0,1].

The corresponding numbers are chosen randomly. If the random number is smaller than crossover rate (CR), we choose the parameter of the trial vector $v'_p(g+1)$ into the corresponding site of the crossover vector $u_p(g+1)$. Otherwise, we choose the parameter of the mutation vector

$$u_p(g+1) = (u_{p1}(g+1), u_{p2}(g+1),\ldots, u_{pD}(g+1)) \quad (9)$$

$$u_{pd}(g+1)=\begin{cases} v'_{pd}(g+1) & \text{if (rand}(d)\le CR) \\ v_{pd}(g+1) & \text{if (rand}(d)> CR) \end{cases} \quad (10)$$

After the crossover vector $u_p(g+1)$ has been generated, the fitness value of $u_p(g+1)$ will be compared with the fitness value of the trial vector $v'_p(g+1)$, i.e.,

$$v'_p(g+1)=\begin{cases} u_p(g+1) & \text{if } f(u_p(g+1))\le f(v'_p(g+1)) \\ v'_p(g+1) & \text{otherwise.} \end{cases} \quad (11)$$

E. Selection

In order to produce better offspring, selection is an important operation. If the trial vector $v'_p(g+1)$ has a less or equal fitness value than that of parent $\Theta_p(g)$, it replaces the parent vector in the next generation; otherwise, the parent is retained in the population. The corresponding equation is as follows:

$$\Theta_p(g+1)=\begin{cases} v'_p(g+1) & \text{if } f(v'_p(g+1))\le f(\Theta_p(g)) \\ \Theta_p(g) & \text{otherwise} \end{cases}. \quad (12)$$

The best individual Θ_{best} is selected in each selection is

$$\Theta_{best}=\begin{cases} \Theta_p(g+1) & \text{if } f(\Theta_p(g+1))\le f(\Theta_{best}) \\ \Theta_{best} & \text{otherwise} \end{cases}. \quad (13)$$

After implementing the selection, we will check whether all vectors have been done in CCDE algorithm. If yes, the scaling factors F_1, F_2, and F_3 will be adjusted by fuzzy logic systems. Finally, the stop criterion in maximum generation G_{max} is checked in this paper.

III. EXPERIMENTAL RESULT

In order to illustrate the performance and viability of the proposed CCDE algorithm, the CCDE and DE are applied to solve the optimization of benchmark functions [18]. In addition, comparision results for illustrating the effectiveness of our modifications (modified mutation with self-adapting scaling factor, Chemotaxis by BP technique, cooperative strategy, and Chemotaxis by BP technique with optimal learning rate) are shown in Table III. The used benchmark test functions are introduced in Table II, detailed description of these twenty function can be found in [9, 18], where D denoted the dimension of test function, searching range is denoted range of variable which is defined to be symmetric about the origin, $f_{\min}$ is optimum value of function.

Herein, the dimension of functions f_1, f_3, f_5, f_6, f_8, f_{14}, f_{15}, f_{16}, f_{17}, and f_{18} is set to be two. The dimension of functions f_{19}, f_{20} is four. The value of five for the dimension of function f_2, f_4, f_7, f_9, f_{10}, f_{11}, f_{12}, and f_{13} are selected. In this paper, we choose F =0.5 and CR=0.5, and maximum number of generation is 20 and population size is 50.

A. Comparison Results between CCDE and DE Algorithms

We set the population size is 50. The populations for all DE and CCDE variants were initialized using same random seeds, maximum number of generation is 20, F = 0.5 and CR = 0.5, the average results of 30 independent runs in Table III.

For the function f_1-f_{20}, the comparison shows that CCDE gives better performance than DE, CCDE algorithm performs better than DE.

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

B. *Illustration Discussions*

To illustrate the performance and effectiveness of CCDE, several comparative experiments are shown to demonstrate the ability and advantages of the adopted modified strategies, e.g., chemotaxis by BP technique, self-adapting scaling factor, cooperative strategy, BP's optimal learning. In order to make the comparison on fair, the populations for all algorithm comparsion variants were initialized using same random seeds, population size is 50, maximum number of generation is 20, F_1=0.5, F_2=0.5, F_3=0.5 and CR=0.5,the average results of 30 independent runs in Table III. Some observations are found from Table III. From each column values of Table III (optimization results of twenty functions), we can observe that the CCDE's results have better accuracy compared with the results using CCDE without using chemotaxis by BP technique, self-adapting scaling factor, cooperative strategy, and BP's optimal learning, respectively. Thus, we could conclude that the chemotaxis by BP technique, self-adapting scaling factor, cooperative strategy, and BP's optimal learning can improve the ability of optimization searching.

IV. Conclusion

The CCDE algorithm adopts the modified mutation strategy to increase the diversity. The modified mutation strategy also provides a correct direction to the optimal solution by the momentum term and the best individual which is updated instantly. The self-adaptive scaling factor strategy improvement could increase the precision and convergence velocity. The chemotaxis by BP technique supplies the capability of searching in the neighborhood. In addition, we use the optimal learning rates to improve the ability of convergence. The simple cooperative strategy provides the capability of searching optimal solution by replacing the parameters of one population of the trial vector with the parameters of the same population of the parent vector. Therefore, CCDE algorithm has the ability of global optimization, advantages of faster convergent speed and higher precision. Experimental results of function optimization are shown to demonstrate the effectiveness of the proposed CCDE.

REFERENCES

[1] F. Bergh and A. P. Engelbrecht, "A Cooperative Approach to Particle Swarm Optimization," *IEEE Trans. on Evolutionary Computation*, Vol. 8, No. 3, pp. 225-239, June, 2004.

[2] M. Clerc and J. Kenney, "The Particle Swarm-explosion, Staility, and Convergence in A Multidimensional Complex Space," *IEEE Trans. on Evolutionary Computation*, Vol. 6, No. 1, pp. 58-73, 2002.

[3] W. A. Farag, V. H. Quintana, and L. T. Germano, "A Genetic-based Neuro-fuzzy Approach for Modeling and Control of Dynamical Systems," *IEEE Trans. on Neural Networks*, Vol. 9, No. 5, pp. 756-767, 1998.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, 1989.

[5] V. G. Gudise and G. K. Venayagamoorthy, "Comparison of Particle Swarm Optimization and Backpropagation as Training Algorithm for Neural Network," *IEEE Swarm Intelligence Symp.*, pp. 110-117, April, 2003.

[6] C. F. Juang, "A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design," *IEEE Trans. on Systems, Man, Cybernetics- Part: B*, Vol. 34, No. 2, pp. 997-1006, 2004.

[7] D. H. Kim, A. Abraham, and J. H. Cho, "A Hybrid Genetic Algorithm and Bacterial Foraging Approach for Global Optimization," *Information Sciences*, Vol. 177, No. 18, pp. 3918-3937, 15 Sep, 2007.

[8] C. H. Lee and M. H. Chiu, "Adaptive Nonlinear Control Using TSK-type Recurrent Fuzzy Neural Network System," *Lecture Notes in Computer Science*, Vol. 4491, pp. 38-44, 2007.

[9] C. H. Lee and F. K. Chang, "Recurrent Fuzzy Neural Controller Design for Nonlinear Systems Using Electromagnetism-like Algorithm," *Far East Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 1, No. 1, pp. 5-22, 2008.

[10] C. Y. Lee and X. Yao, "Evolutionary programming using mutations based on the Lévy probability distribution," *IEEE Trans. Evol. Comput*., Vol. 8, No. 1, pp. 1–13, 2004.

[11] K. E. Parsopoulos, "Cooperative Micro-Differential Evolution for High-dimensional Problems," *Genetic and Evolutionary Computation Conference 2009 (GECCO 2009)*, pp. 531-538, Montreal, Canada, 2009.

[12] M. A. Potter and K. De Jong, "A Cooperative Coevolutionary Approach to Function Optimization," *In Proceedings from the Third Parallel Problem Solving from Nature*, Vol. 2, pp. 249-257, 1994.

[13] R. Sarker, M. Mohammadian, and X. Yao, *Evolution Optimization*, Kluwer Academic Pub., 2002.

[14] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 24, No. 4, pp. 656-667, 1994.

[15] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 24, No. 4, pp. 656-667, 1994.

[16] R. Storn, K. Price, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, Berlin, 2005.

[17] W. J. Tang, Q. H. Wu, and J. R. Saunders, "Bacterial Foraging Algorithm for Dynamic Environments," *2006 IEEE Congress on Evolutionary Computation*, (*CEC 2006*), pp. 1324-1330, Canada, 2006.

[18] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput*., vol. 3, no. 2, p. 82, Jul. 1999.

[19] X. Zhang, W. Chen, C. Dai, and A. Guo, "Self-adaptive Differential Evolution Algorithm for Reactive Power Optimization," *2008 IEEE Int. Con. on Natural Computation (ICNC 2008)*, pp. 560-564, 2008.

[20] D. Xu, S. Li, and F. Qian, "An Improved Differential Evolution Algorithm and Its Application in Reaction Kinetic Parameters Estimation," *2007 International Conference on Intelligent Systems and Knowledge Engineering* (*ISKE 2007*), Oct. 2007.

TABLE II
BENCHMARK FUNCTION

Test function	D	Search Range	$f_{\min}$
$f_1(x)=\sum_{i=1}^{D} x_i^{\,2}$	2	$[-100, 100]^D$	0
$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	5	$[-10, 10]^D$	0
$f_3(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j^{\,2}\right)$	2	$[-100, 100]^D$	0
$f_4(x)=\max\lvert x_i\rvert\ ,1\le i\le D$	5	$[-100, 100]^D$	0
$f_5(x)=\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	2	$[-30, 30]^D$	0

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$f_6(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	2	$[-100, 100]^D$	0
$f_7(x)=\sum_{i=1}^{D} ix_i^4+random[0, 1)$	5	$[-1.28, 1.28]^D$	0
$f_8(x)=\sum_{i=1}^{D}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	2	$[-500, 500]^D$	-837.29
$f_9(x)=\sum_{i=1}^{D}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	5	$[-5.12, 5.12]^D$	0
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{D}x_i^2}{D}}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)+20+e$	5	$[-32, 32]^D$	0
$f_{11}(x)=\frac{1}{4000}\sum_{i}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	5	$[-600, 600]^D$	0
$f_{12}(x)=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D}(y_i-1)^2[1+\sin^2(\pi y_i+1)]+(y_D-1)\}+\sum_{i=1}^{D}u(x_i,10,100,4),$ $y_i=1+\frac{1}{4}(x_i+1), u(x_i,10,100,4)=\begin{cases}k(x_i-a)^m, & x_i>a,\\ 0, & -a\le x_i\le a,\\ k(-x_i-a)^m, & x_i<a.\end{cases}$	5	$[-50, 50]^D$	0
$f_{13}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+10\sin^2(\pi y_i+1)]+(x_D-1)^2[1+10\sin^2(2\pi x_D)]\}+\sum_{I=1}^{D}u(x_i,5,100,4)$	5	$[-50, 50]^D$	0
$f_{14}(x)=\left[\frac{1}{500}+\sum_{J=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right]^{-1}$	2	$[-65.536, 65.536]^D$	0.998004
$f_{15}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	$[-5, 5]^D$	-1.031628 5
$f_{16}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	$[-5, 5]^D$	0.398
$f_{17}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	$[-2, 2]^D$	3
$f_{18}(x)=-\sum_{i=1}^{10}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	2	$[0, 10]^D$	-10.5364
$f_{19}(x)=-\sum_{i=1}^{5}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	4	$[0, 10]^D$	-10.1532
$f_{20}(x)=-\sum_{i=1}^{7}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	4	$[0, 10]^D$	-10.4029

TABLE III
COMPARISON RESULTS OF CCDE AND DE ALGORITHMS IN AVERAGED 30 INDEPENDENT RUNS.

F	Gen.	CCDE	DE	CCDE without chemotaxis BP	CCDE without Self-adapting scaling factor	CCDE without cooperative strategy	CCDE without optimal η
		Mean value	Mean value	Mean value	Mean value	Mean value	Mean value
f_1	20	1.3562E+00	2.2537E+02	1.3076E+02	6.7779E+00	2.9760E+01	2.9631E+01
f_2	20	2.8408E+00	2.3733E+01	1.3084E+01	3.6327E+00	1.6967E+01	3.1112E+00
f_3	20	1.7387E+00	4.0347E+02	3.0619E+02	2.0463E+00	7.1724E+00	5.8062E+00
f_4	20	8.3552E-02	2.1333E+00	1.3163E+00	9.7480E-02	9.7530E-02	8.8005E-02
f_5	20	2.2572E+00	2.9130E+02	9.4253E+01	4.4352E+01	1.0797E+02	1.5113E+01
f_6	20	6.8424E+00	2.3768E+02	1.4165E+02	2.2339E+01	2.9735E+01	1.8024E+01
f_7	20	1.5800E+00	9.4880E+00	7.8914E+00	1.6702E+00	5.3505E+00	1.7683E+00
f_8	20	-1.5188E+02	-9.7757E+01	-1.0964E+02	-1.1982E+02	-1.3662E+02	-1.0743E+02
f_9	20	2.1718E+00	1.5467E+01	1.4194E+01	2.2667E+00	1.4998E+01	2.4451E+00
f_{10}	20	5.2389E+00	1.4634E+01	1.3302E+01	6.3802E+00	1.3360E+01	5.8441E+00
f_{11}	20	1.4009E+00	2.4807E+01	1.9503E+01	3.5073E+00	1.1604E+01	1.5150E+00
f_{12}	20	1.9340E+00	1.1237E+02	1.6367E+01	3.3453E+00	6.7860E+00	2.0628E+00
f_{13}	20	2.8607E-01	6.1504E+02	1.8021E+01	7.2370E-01	6.1410E+00	1.2461E+01
f_{14}	20	2.9406E+00	3.4973E+01	2.3064E+01	3.3578E+00	6.5264E+00	4.1973E+00
f_{15}	20	-1.0257E+00	-7.2105E-01	-9.1144E-01	-1.0253E+00	-9.5154E-01	-9.8484E-01
f_{16}	20	6.1544E-01	2.1481E+00	1.7251E+00	7.2490E-01	8.5132E-01	6.9795E-01
f_{17}	20	3.4254E+00	4.2136E+00	3.4987E+00	4.0041E+00	3.5695E+00	3.8161E+00
f_{18}	20	-1.0258E+01	-1.8559E+00	-1.0176E+01	-1.0022E+01	-9.9662E+00	-8.7192E+00
f_{19}	20	-9.6705E+00	-2.1524E+00	-9.4761E+00	-9.0402E+00	-9.4559E+00	-9.2140E+00
f_{20}	20	-9.7425E+00	-1.3156E+00	-9.6996E+00	-9.6168E+00	-9.3156E+00	-9.1345E+00

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)