International Journal of Computer Applications (0975 - 8887)
Volume 124 - No.4, August 2015

Computational Chemotaxis in Micro Bacterial Foraging Optimization for High Dimensional Problems: A Comparative Study on Numerical Benchmark

Yunus Emre Yildiz
Epoka University
Tirana, Albania

Oguz Altun
Yldz Tech. University
Istanbul, Turkey

Ali Osman Topal
Epoka University
Tirana, Albania

ABSTRACT

Nature and bio-inspired algorithms have been recently used for solving high dimensional search and optimization problems. In this context, bacterial foraging optimization algorithm (BFOA) has been widely employed as a global optimization technique inspired from social foraging behavior of *Escheria coli* bacteria. In this paper, a novel hybrid technique called micro Chemotaxis Differential Evolution Optimization Algorithm (CDEOA) that uses a small population is proposed. In this technique, we incorporate the principles of DE (Differential Evolution) into BFOA. The best bacterium retains its position whereas the rest of the population are reinitialized on the search space. CDEOA was compared with classical BFOA with two different population sizes and micro BFOA (BFOA) over a suite of 16 numerical optimization problems taken from P.N. Suganthan. Statistics of the computer simulations indicate that CDEOA outperforms, or is comparable to, its competitors in terms of its convergence rates and quality of final solution for complex high dimensional problems.

General Terms

Evolutionary Algorithms (EAs), Neural Networks

Keywords

Micro Bacterial Algorithms, Differential Evolution, Nature-Inspired Algorithms, Hybrid BFOA, Metaheuristics

1. INTRODUCTION

In the past few decades, the micro algorithms have been studied by several researchers in order to solve high dimensional optimization problems. High dimensionality makes the problems hard and computational time consuming due to the fact that it increases the number of parameters to be optimized. In this context, in case that the population size remains large as in its original algorithm, it would not be that easy for the parameters to converge to the optimal values. As a remedy to this challenge, Krishnakumar et al. [5] proposed to use micro-genetic algorithm which is based on a very small population size. It is clear that although small population size algorithm is good at exploiting the promising areas of the search space, it is not able to preserve the diversity of population. However, when the diversity of population fails, the population can be reinitialized and the best individuals are kept on the search space. This not only leads to prevent the premature convergence but also makes the individuals explorative [8].

Micro algorithms are the techniques with a small population size (e.g. 2, 3, 5, or 6). Recently, several studies have been conducted regarding the micro bio and nature inspired algorithms to solve the high dimensional optimization problems. Caraffino et al. [2] proposed micro Differential Evolution (DE) that incorporates an extra search move into DE to improve the best solution. Chu et al.[3] proposed Fast Bacterial Swarming Algorithm that hybridizes BFOA and PSO. Parsopoulos [11] proposed a cooperative micro technique, Cooperative Micro Differential Evolution (CDE), to solve high dimensional problems. Parsopoulos et al. [12] also introduced a parallel master-slave model for CPSO. Olorunda [10] presented cooperative differential evolution that divides the high dimensional problem space into smaller parts and have each part optimized by a separate population. Sotelo-Figueroa [16] proposes a novel approach called Micro Differential Algorithm that evolves an indirect representation of bin packing problem. Fuentes et al.[1] presents a particle swarm optimizer that solves constrained optimization problems. Also, Rahnamayan et al. [15] proposed micro Opposition based DE (-ODE) that deals with minimization of dissimilarity between the input grey-level image and the bi-level (thresholded) image in image processing field. Olguin-Carbajal et al. [9] proposed the micro DE Local Search (-DELS) that incorporates local search technique into micro DE.

Micro algorithms have proved to be an efficient tool in solving optimization problems for high dimensional (e.g. 500, 750, and 1000) problems that standard nature-inspired and bio-inspired techniques fail. Here we propose a micro algorithm which hybridizes BFOA and DE.

We were inspired by the ideas of micro Bacterial Foraging Optimization Algorithm (BFOA) [4] which is successfully used to solve high-dimensional optimization problems. BFOA does not use the reproduction operator to avoid premature convergence whereas the chemotaxis operator is employed for updating the position of a bacterium. In BFOA, which uses three bacteria, the best bacterium retains its position in the swarm; the second best bacterium is re-positioned in the vicinity of the best bacterium; and the third bacterium is dispersed to a random location. This approach aims to avoid premature convergence and helps to maintain the search diversity. In our study, in order to increase the convergence per-

formance and quality of the final solution, after re-initializing the population, the bacteria are ranked according to their cost function values. The best bacterium's position is preserved in the population. The second best bacterium is reinitialized in the neighborhood of the best bacterium based on the ideas of DE technique, whereas the rest of the bacteria (four bacteria) are dispersed at random on the search space.
The remaining parts of this paper is organized as follows: Section 2 introduces the related processes of classical BFOA. Section 3 outlines the operators of the DE algorithm. Section 4 provides the details of the proposed algorithms. Section 5 includes a short summary of simulation results and performance comparison. Section 6 presents the source codes of the framework and the methods that were employed. Section 7 concludes the paper.

2. CLASSICAL BACTERIAL FORAGING OPTIMIZATION ALGORITHM

The bacterial foraging system consists of three principal mechanisms, namely chemotaxis, reproduction, and elimination-dispersal ([13]). Below we briefly describe each of these operators.

2.1 Chemotaxis

A *E.coli* tends to accumulate the food in the nutrient-rich areas by a process called chemotaxis. This process is performed with consecutive *tumble* and *swim (run)* steps via flagella.[1] A unit walk in the same direction with the previous unit walk is called a *run (or swim)*, whereas a unit walk in a different direction than the previous unit walk is called a *tumble*. *E.coli* alternates between these two modes of operation throughout its entire life-time. Suppose $\theta(i,j,k,l)$ represents the position of the i-th bacterium at j-th chemotactic, k-th reproductive and l-th elimination-dispersal step. The position of the bacterium in the next step may be represented by Eq. eq:t vector and Eq. eq:step of bac.,

$$t(j) = \frac{\Delta(i)}{\sqrt{\Delta^T(i) * \Delta(i)}} \tag{1}$$

$$\theta(i,j+1,k,l) = \theta(i,j,k,l) + C(i) * t(j) \tag{2}$$

where $C(i)$ is the predefined constant length of the unit walk, $t(j)$ (Eq. 1) is the direction angle of the step, and $\Delta(i)$ is a random vector whose elements lie in $[-1,1]$. In a *run* step, $t(j)$ remains the same as $t(j-1)$; in a *tumble* step, $t(j)$ is generated randomly from uniform distribution in the range $[0, 2pi]$.

2.2 Reproduction

The total cost of each bacterium is computed as the sum of the cost function values calculated during its life-time after all chemotaxis steps. Then, all bacteria are sorted according to their cost in ascending order. Only the first half of the swarm (the healthiest ones) survive and the healthiest bacteria split into two. Hence there are two bacteria in each position in this step. Since the remaining 50% with poor health is discarded, the population size is kept fixed.

[1] Note that swim and run can be used interchangeably in the literature of BFOA.

2.3 Elimination and dispersal

In addition to chemotaxis for local search and reproduction for accelerating the convergence rate, there is a need of an operator for global searching to make the algorithm explorative. Elimination and dispersal will help the bacterium avoid getting stuck and getting trapped in local optima. In order to imitate these events in BFOA, some bacteria are liquidated randomly with a constant probability (P_{ed}) while the new replacements are initialized randomly in the search space.

3. DIFFERENTIAL EVOLUTION (DE)

Differential evolution (DE) is a population based bio-inspired technique which utilizes mutation, crossover, and selection operators to minimize an objective function. At generation G, a new population is generated out of the current population members according to a uniform probability distribution $x_{i,G}, i = 1,2,...,N$ where N is the population size. After initialization, DE undergoes mutation, crossover, and selection operators [17].
At each generation G, a mutant vector $v_{i,G}$ is generated for each target vector $x_{i,G}, i = 1,2,...N$ in the current population. Some of the most used mutation strategies[2] in the literature are as follows:

—DE/rand/1

$$v_{i,G} = x_{r_0,G} + F * (x_{r_1,G} - x_{r_2,G}) \tag{3}$$

—DE/current-to-best/1

$$v_{i,G} = x_{r_0,G} + F*(x_{r_1,G} - x_{r_2,G}) + F*(x_{r_3,G} - x_{r_4,G}) \tag{4}$$

—DE/best/1

$$v_{i,G} = x_{best,G} + F * (x_{r_0,G} - x_{r_1,G}) \tag{5}$$

where r_0, r_1, r_2, r_3, and r_4 are distinct integers randomly chosen from the current population and are different from i. $x_{best,G}$ is the best individual vector in the current generation G, and F is the mutation factor which is generally within the range of $[0, 1+]$. In our approach, DE/best/1 strategy (Eq. 5) is employed due to its fast convergence speed and strong exploitation capability.
After mutation, a binomial crossover operation is carried out on $v_{i,G}$ and $x_{i,G}$ to generate a new trial vector $u_{i,G} = (u_{i,1,G}, u_{i,2,G}, ..., u_{i,D,G})$ using Eq. (6),

$$u_{i,j,G} = \{ \ v_{i,j,G},\ if\ R_j(0,1) \le C_r\ or\ j = j_{rand}\, x_{i,j,G},\ if\ R_j(0,1) > C$$ (6)

where $j = 1,2,...,D$, j_{rand} is a randomly chosen integer in $[1,D]$, $R_j(0,1)$ is uniformly generated random number between 0 and 1 for each j, and $C_r \in [0,1]$ is the crossover rate parameter.
A selection process is carried out to choose the better of the parent vector $x_{i,G}$ and the trial child vector $u_{i,G}$. In case of a minimization problem, the selected parent vector in the next generation is given by Eq. eq:selectioneq,

$$x_{i,G+1} \{ \ u_{i,G},\ if\ \ f(u_{i,G}) < f(x_{i,G})\ x_{i,G},\ otherwise \tag{7}$$

where $f(\cdot)$ is the function for minimization. If trial vector $u_{i,G}$ produces a better fitness value, it replaces its parent in the next generation; otherwise the parent is kept in the population.

[2] `http://www.icsi.berkeley.edu/~storn/code.html`

Algorithm 1 Detailed pseudo-code of CDEOA

[1] Parameters: $S \leftarrow$ total number of bacteria in the population $N_{ed} \leftarrow$ number of elimination dispersal steps $N_c \leftarrow$ number of chemotaxis steps $N_s \leftarrow$ swimming steps $C(i) \leftarrow$ the run-length unit $f \leftarrow$ objective function to be minimized //Initialize some local variables $\theta_{best} \leftarrow$ random position in the search space $f_{best} \leftarrow f(\theta_{best})$ $M_{fes} \leftarrow$ maximum number of FEs allowed $N_{fes} \leftarrow 0$ //current number of function evaluations // Define a helper function J that will call the actual objective function f. This helper function also updates the N_{fes}, θ_{best}, and f_{best} variables. $J(\theta)$: $v \leftarrow f(\theta)$ $N_{fes} \leftarrow N_{fes}$ +1//update number of FEs v ¡ f_{best} $\theta_{best} \leftarrow \theta$ //update global best location $f_{best} \leftarrow v$ //update global best function value end//if v end// function N_{fes} ¡ M_{fes}//FEs control loop j `from 1 to` N_{ed}//Elimination dispersal loop$j$ `from 1 to` N_c//Chemotaxis loop$i$ `from 1 to` S//Tumble-Swim loop $J_{last} \leftarrow J(\theta(i,j,k))$// $J(\cdot)$ computes the fitness $\Delta(i) \leftarrow$ random vector within [-1,1] //tumble $\theta(i,j+1,k)$ = $\theta(i,j,k)$+ $C(i) * \frac{\Delta(i)}{\sqrt{\Delta^T(i)*\Delta(i)}}$ //Swim: m `from 1 to` N_s// Swim loop $J(\theta(i,j+1,k))$ ¡ J_{last} J_{last} = $J(\theta(i,j+1,k))$ $\theta(i,j+1,k)$ = $\theta(i,j,k)$+$C(i) * \frac{\Delta(i)}{\sqrt{\Delta^T(i)*\Delta(i)}}$ m = N_s//Break from switch loop end//if end//Swim loop end//Tumble-Swim loop end //Chemotaxis loop Bacterium with rank 2 undergoes DE operators as in Eq. (5), Eq.(6), and Eq. (7) The rest (4 bacteria) except rank 1-2 are dispersed to the random positions.
i from 1 to S //Elimination dispersal loop random number ¿ P_{ed} Disperse to a random position end/if end//Elimination dispersal loop end // FEs control loop θ_{best}

4. MICRO CDEOA

In CDEOA, a population of six bacteria which make consecutive *tumble* and *run* steps (chemotaxis) throughout their lifetime. After a chemotaxis loop, all the bacteria are sorted according to their objective function values. A bacterium which is close to the global optimum is called the best bacterium (rank 1). The second best bacterium (rank 2) attempts to approach the neighborhood of the best bacterium through the means of DE operators (mutation, crossover, and selection). The rest of the population (four bacteria) are dispersed to the random positions in the search space. Unlike the population size of BFOA, the population size of CDEOA is increased to an appropriate value, 6, due to the number of the individuals chosen in mutation strategies (Eq. 3, Eq. 4, and Eq. 5). The second best bacterium (rank 2) is positioned in the vicinity of the best bacterium. This is carried out through the means of DE mutation strategies. In our study, we employed DE/best/1 (Eq 5) mutation strategy which yields a best solution based trial vector.
In order to figure out the behavior of the virtual bacteria in BFOA, we illustrated the six bacteria in a one dimensional search space in Fig. (1). Objective is the minimization of 1-dimensional sphere function Eq. (8) which is a widely employed unimodal function with a minimum equal to 0.

$$f(x) = x^2 \tag{8}$$

In Eq. (8), the parameter x is the position of a bacterium, and $f(x)$ is the objective function value. The roles of six bacteria may change after a chemotaxis process. The closest position to the global optimum of the search space is retained by the best bacterium (rank 1). The dispersal of the second best bacterium (rank 2) to a position close to the best bacterium will ease local search for the next chemotaxis process. Maintaining the population diversity and avoiding premature convergence are performed by the worst bacteria (rank 3-6). A flowchart of the classical BFOA adapted from Dasgupta [4] is given in Fig. 2.

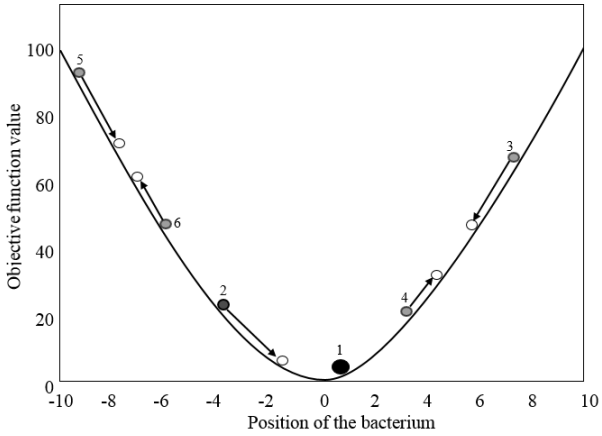

Fig. 1: Behavior of the bacteria on one dimension

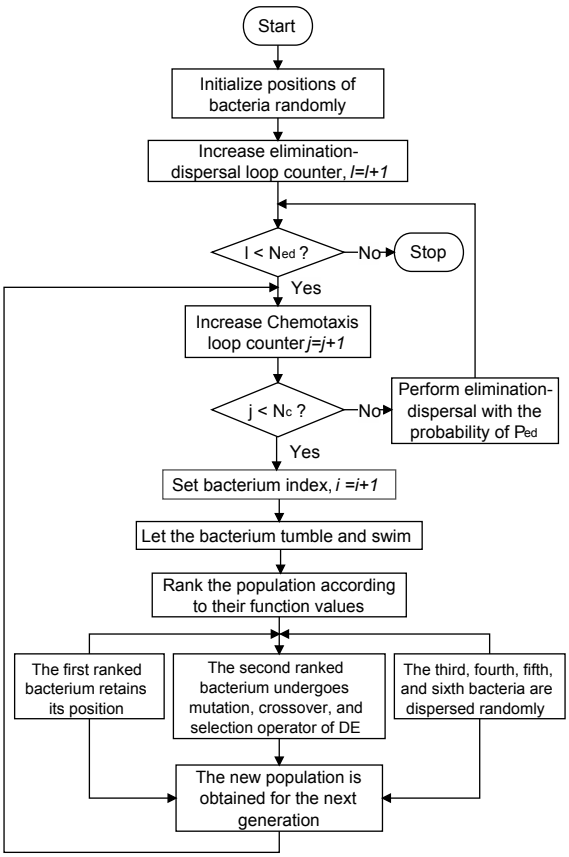

Fig. 2: Flowchart of the CDEOA

5. EXPERIMENTAL STUDY AND RESULTS

The CDEOA was tested using a set of 16 unimodal and multimodal benchmark functions (see Section 5.1) taken from IEEE CEC special sessions and competitions on single objective real parameter numerical optimization [18], [6]. Unlike standard benchmark functions, the shifted functions shift the global optimum to a random position, i.e., $F(x) = f(x - o_{new})$, where $F(x)$ is the new function, $f(x)$ is the old function, and o_{new} is the new global optimum with different values for different dimensions. Its global optimum is not situated at the center of the search space. The rotated functions rotate the function $F(x) = f(Mx)$, where M is an orthogonal rotation matrix [7]. The descriptions of these functions are given in Table 1. Functions 1-6 are unimodal and functions 7-16 are simple multimodal functions.

Table 1. : Global optimum, the search ranges and the global best (f(x*)) of 500 dimensional test functions. ***0*** is the shifted vector, C=Characteristics of test functions: U=Unimodal, M=Multimodal, S=Separable, N=Non-Separable

f	Global Optimum x*	f(x*)	C	Search Range
f_1	***0***	0	US	(-2,2)
f_2	(0,0,,0)	0	UN	(-500,500)
f_3	***0***	0	UN	(-500,500)
f_4	***0***	0	UN	(-500,500)
f_5	(0,0,,0)	0	UN	(-500,500)
f_6	***0***	0	UN	(-100,100)
f_7	***0***	0	MN	(-2,2)
f_8	***0***	0	MN	(-2,2)
f_9	***0***	0	MN	(-2,2)
f_{10}	***0***	0	MN	(-2,2)
f_{11}	***0***	0	MN	(-10,10)
f_{12}	***0***	0	MN	(-10,10)
f_{13}	***0***	0	MS	(-2,2)
f_{14}	***0***	0	MN	(-2,2)
f_{15}	(420.96,,420.96)	0	MN	(-2,2)
f_{16}	(420.96,,420.96)	0	M	(-500,500)

5.1 Test Functions

(1) Shifted Sphere Function $f_1(x) = \sum_{i=1}^{D} z_i^2$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(2) Schwefel problem 1.2 $f_2(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} z_j)^2$

(3) Shifted Schwefel problem 1.2 $f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} z_j)^2$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(4) Shifted Schwefel problem 1.2 with noise in fitness $f_4(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} z_j)^2 * (1 + 0.4|N(0,1)|)$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(5) Schwefel problem 2.21 $f_6(x) = max\{|x_i, 1 \leq i \leq D|\}$

(6) Shifted and rotated high conditioned elliptic function $f_6 = \sum (10^6)^{\frac{i-1}{D-1}} z_i^2$
$z = M(x - o)$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(7) Shifted Rosenbrock's function $f_7(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$
$z = x - o + 1$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(8) Shifted and rotated Rosenbrock's function $f_8(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$
$z = M(x - o)$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(9) Shifted Ackley's function $f_9(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2})$
$- exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi z_i)) + 20 + e,$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(10) Shifted rotated Ackley's function $f_{10}(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2})$
$- exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi z_i)) + 20 + e,$
$z = M(x - o)$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(11) Shifted Griewank's function $f_{11}(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000}$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(12) Shifted and rotated Griewank's function $f_{12}(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000}$
$z = M(x - o)$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(13) Shifted Rastrigin's function $f_{13}(x) = \sum_{i=1}^{D} (z_i^2 - 10cos(2\pi z_i) + 10)$
$z = x - o$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(14) Shifted and rotated Rastrigin's function $f_{14}(x) = \sum_{i=1}^{D} (z_i^2 - 10cos(2\pi z_i) + 10)$
$z = M(x - o)$
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(15) Shifted noncontinuous Rastrigin's function $f_{15}(x) = \sum_{i=1}^{D} (y_i^2 - 10cos(2\pi y_i) + 10)$
$y_i = \{ round(2z_i)/2, |z_i| >= 1/2 z_i, |z_i| < 1/2$
`for i = 1,2,...,D`
$o = [o_1, o_2, ...o_D] : shiftedglobaloptimum$

(16) Shwefel's function $f_{16}(x) = 418.9829 * D - \sum_{i=1}^{D} x_i sin(|x_i|^{1/2})$

5.2 Parametric setup

We use the same parameter values as in the original papers in each technique. DE is sensitive to the mutation scaling factor F and crossover rate C_r. Choosing C_r=0.9 or 1.0 not only speeds up convergence but also diversifies the population by means of one of the best solution dependent DE mutation strategies, DE/best/1. In this context, C_r parameter of DE was set to be 0.9. F is selected within the range of $[0-2.0]$. It is reported that a smaller F value (e.g., 0.5) can lead to a statistically better performance than the other parameter values ([17], [14]). Therefore, F of -CDEOA was set to be 0.5. For the proposed technique and the classical BFOA, the following parameter values were set: N_s=12, N_{re}=16, $C(i)$=0.1.

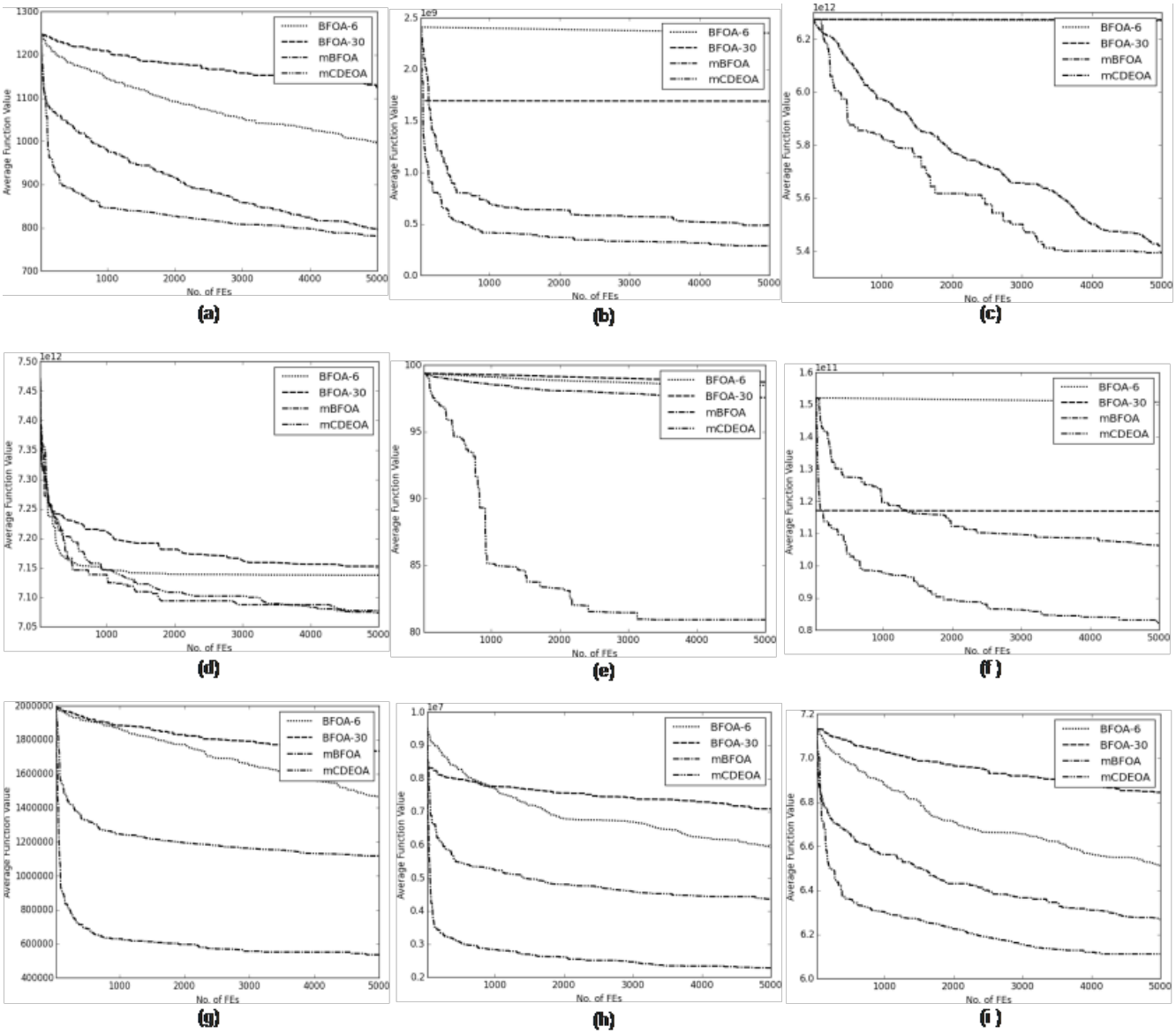

Fig. 3: Median convergence graphs of BFOA-6, BFOA-30, BFOA, and -CDEOA for test functions (a) f_1: Shifted Sphere; (b) f_2: Schwefel problem 1.2; (c) f_3: Shifted Schwefel problem 1.2; (d) f_4: Shifted Schwefel problem 1.2 with noise in fitness;(e) f_5: Schwefel problem 2.21; (f) f_6: Shifted and rotated high conditioned elliptic; (g) f_7: Shifted Rosenbrock; (h) f_8: Shifted and rotated Rosenbrock; (i) f_9: Shifted Ackley.

5.3 Simulation

The study introduced in this paper aims to test the quality of the final solution and the convergence speed at the end of a fixed number of function evaluations (FEs). The maximum number of FEs was set to 5 10^3. All simulations were done on 500-D problems. Each algorithm and the objective function pair were run 50 times. For convenience of illustration, the convergence graph was plotted for the first 9 functions in Fig. 3. The horizontal axis of these graphs is the number of function evaluations and the vertical axis is the mean of function values. Table 2 and Table 3 report the best final function value (BFV), the worst final function value (WFV), the mean of the final best function value (Mean), the median of the final best function value (Median). These values comply $(F(x) - F(x^*))$ for evaluating the success of five algorithms, where x is the best value of the bacterium in a run and x^* is the global best of the test function (Table 1). The standard deviation of the final best function value and the mean time spent per trial in seconds are also reported.

5.4 Comparison of CDEOA with three nature-inspired techniques

The performance of CDEOA technique was compared with classical BFOA with 6 and 30 population sizes, BFOA-6, BFOA-30

Table 2. : Comparison of BFOA-6, BFOA-30, BFOA, and -CDEOA. BFV: the best final function value, WFV: the worst final function value, Mean: mean of the final best function value, Median: median of the final best function value, StdDev: the standard deviation of the final best function value, Time: mean time spent per trial in seconds

Functions	Algorithm	BFV	WFV	Mean	StdDev	Med	Time
f_1	BFOA-6	9.17E+02	1.05E+03	9.95E+02	3.22E+01	9.96E+02	0.47
	BFOA-30	1.07E+03	1.16E+03	1.12E+03	2.3E+01	1.12E+03	0.56
	-BFOA	6.77E+02	9.05E+02	7.91E+02	5.97E+01	7.96E+02	0.65
	-CDEOA	7.16E+02	8.68E+02	7.82E+02	3.93E+01	7.80E+02	0.55
f_2	BFOA-6	2.33E+09	2.37E+09	2.35E+09	7.48E+06	2.35E+09	27.61
	BFOA-30	1.69E+09	1.69E+09	1.69E+09	9.80E+05	1.69E+09	27.77
	-BFOA	2.26E+08	6.11E+08	4.24E+08	7.93E+07	4.27E+08	28.08
	-CDEOA	1.17E+08	5.28E+08	2.92E+08	8.65E+07	2.89E+08	27.32
f_3	BFOA-6	6.27E+12	6.27E+12	6.27E+12	5.04E+08	6.27E+12	27.74
	BFOA-30	6.27E+12	6.27E+12	6.27E+12	2.58E+08	6.27E+12	28.35
	-BFOA	4.47E+12	5.70E+12	5.07E+12	2.74E+11	5.04E+12	28.42
	-CDEOA	5.23E+12	6.18E+12	5.60E+12	2.19E+11	5.63E+12	28.43
f_4	BFOA-6	7.14E+12	7.17E+12	7.14E+12	6.51E+09	7.14E+12	28.33
	BFOA-30	7.14E+12	7.22E+12	7.16E+12	1.82E+10	7.15E+12	28.27
	-BFOA	6.91E+12	7.18E+12	7.08E+12	5.85E+10	7.09E+12	28.23
	-CDEOA	6.95E+12	7.21E+12	7.11E+12	4.86E+10	7.11E+12	28.52
f_5	BFOA-6	9.81E+01	9.87E+01	9.84E+01	1.24E-01	9.85E+01	0.91
	BFOA-30	9.85E+01	9.90E+01	9.87E+01	1.25E-01	9.88E+01	0.86
	-BFOA	9.51E+01	9.80E+01	9.73E+01	5.85E-01	9.74E+01	0.96
	-CDEOA	7.30E+01	8.94E+01	8.21E+01	3.55E+00	8.17E+01	0.90
f_6	BFOA-6	1.46E+11	1.51E+11	1.51E+11	9.73E+08	1.51E+11	1.14
	BFOA-30	1.14E+11	1.17E+11	1.17E+11	3.68E+08	1.17E+11	1.12
	-BFOA	8.33E+10	1.11E+11	1.02E+11	6.92E+09	1.03E+11	1.27
	-CDEOA	7.21E+10	1.08E+11	8.84E+10	7.77E+09	8.86E+10	1.15
f_7	BFOA-6	1.30E+06	1.62E+06	1.47E+06	6.13E+04	1.47E+06	0.55
	BFOA-30	1.60E+06	1.82E+06	1.73E+06	4.47E+04	1.73E+06	0.55
	-BFOA	3.57E+05	5.79E+05	4.90E+05	5.58E+04	4.97E+05	0.59
	-CDEOA	8.13E+05	1.25E+06	1.09E+06	7.43E+04	1.10E+06	0.52
f_8	BFOA-6	5.16E+06	6.87E+06	5.91E+06	3.41E+05	5.87E+06	1.27
	BFOA-30	6.43E+06	7.52E+06	7.07E+06	2.52E+05	7.08E+06	1.28
	-BFOA	1.32E+06	2.59E+06	2.13E+06	2.80E+05	2.20E+06	1.42
	-CDEOA	3.68E+06	4.91E+06	4.16E+06	2.91E+05	4.14E+06	1.11

[13] and BFOA [4]. The compared algorithm (BFOA) was chosen due to fact that it possesses very small population size (3) as in its original paper. As for BFOA, we aimed to test its performance with small population size (6) and large population size (30).

5.4.1 Unimodal functions f_1 - f_6 . As reported in Table 2, in these six unimodal functions, the micro techniques exhibit their superiority to their classical counterparts in terms of quality of the final solution. CDEOA shares the first place with BFOA in f_1 and f_4 functions. All the algorithms show similar performances in f_4 function. We can also observe that the proposed technique performs better than that of other techniques in f_5 and f_6 functions. In contrast, CDEOA remains behind BFOA in f_2 and f_3 functions. The classical BFOA-6 and BFOA-30 fail in most of the unimodal functions. We can infer that the success of a technique is problem-dependent. While BFOA-6 outperforms the BFOA-30 in only one function f_1, BFOA-30 performs better than BFOA-6 in two functions, f_2 and f_6.

5.4.2 Simple multi modal functions $f_7 - f_{16}$. In these ten multimodal functions, overall, the proposed technique performs better than its competitors. CDEOA exhibits better performance in 4 functions, f_{10}, f_{13}, f_{14}, and f_{15} by outperforming its counterparts. On the other hand, CDEOA exhibits similar performance with BFOA in 2 functions, f_{11} and f_{12} while BFOA outperforms the CDEOA in 2 functions, f_7 and f_8. Overall, classical BFOA-6 and BFOA-30 do not succeed in most of the multimodal functions. However, we can observe that these two classical counterparts catch up with micro algorithms in 2 functions, f_9 and f_{16}

The convergence map of BFOA-6, BFOA-30, BFOA, and CDEOA in Fig. 3 implies that the proposed CDEOA technique has significantly faster and reliable convergence speed than that of its competitors. BFOA is the second fastest technique. We can observe that the classical counterparts exhibit poor convergence speed, thereby getting trapped in local optima whereas the micro techniques tend to move to global optimum Fig. 3.

In summary, although there are slight differences at the quality of final solution and convergence speed of the algorithms, overall, the proposed technique presents superior performance to the other techniques on unimodal and multimodal functions.

6. SOURCE CODES

The Python source codes of the algorithms classical BFOA, BFOA, and proposed CDEOA can be accessed from Y. Emre Yildizs homepage (`https://sites.google.com/site/yeyildiz12/`). These codes are written to be compatible with the opn global optimization framework available in Ouz Altuns Bit-

Table 3. : Comparison of BFOA-6, BFOA-30, BFOA, and -CDEOA. BFV: the best final function value, WFV: the worst final function value, Mean: mean of the final best function value, Median: median of the final best function value, StdDev: the standard deviation of the final best function value, Time: mean time spent per trial in seconds

Functions	Algorithm	BFV	WFV	Mean	StdDev	Med	Time
f_9	BFOA-6	6.29E+00	6.69E+00	6.49E+00	8.88E-02	6.50E+00	0.56
	BFOA-30	6.69E+00	6.91E+00	6.83E+00	4.79E-02	6.83E+00	0.54
	-BFOA	5.90E+00	6.46E+00	6.22E+00	1.20E-01	6.22E+00	0.63
	-CDEOA	5.91E+00	6.32E+00	6.09E+00	8.63E-02	6.06E+00	0.78
f_{10}	BFOA-6	8.52E+00	8.97E+00	8.80E+00	1.23E-01	8.83E+00	1.39
	BFOA-30	8.76E+00	9.06E+00	8.92E+00	5.31E-02	8.92E+00	1.37
	-BFOA	8.01E+00	8.67E+00	8.41E+00	1.34E-01	8.44E+00	1.83
	-CDEOA	7.65E+00	8.22E+00	7.88E+00	1.28E-01	7.87E+00	1.18
f_{11}	BFOA-6	8.13E+00	8.56E+00	8.39E+00	8.78E-02	8.40E+00	1.24
	BFOA-30	8.50E+00	8.74E+00	8.63E+00	6.03E-02	8.64E+00	1.11
	-BFOA	5.65E+00	7.19E+00	6.36E+00	3.74E-01	6.29E+00	1.39
	-CDEOA	5.81E+00	6.84E+00	6.23E+00	2.06E-01	6.22E+00	1.07
f_{12}	BFOA-6	1.66E+01	1.75E+01	1.71E+01	1.98E-01	1.71E+01	1.75
	BFOA-30	1.72E+01	1.79E+01	1.77E+01	1.78E-01	1.76E+01	1.78
	-BFOA	1.15E+01	1.50E+01	1.29E+01	8.26E-01	1.27E+01	2.19
	-CDEOA	1.15E+01	1.34E+01	1.25E+01	4.14E-01	1.26E+01	2.09
f_{13}	BFOA-6	1.02E+04	1.16E+04	1.10E+04	2.85E+02	1.11E+04	0.54
	BFOA-30	1.18E+04	1.23E+04	1.20E+04	1.02E+02	1.20E+04	0.51
	-BFOA	9.52E+03	1.09E+04	1.03E+04	3.39E+02	1.03E+04	0.65
	-CDEOA	8.94E+03	1.01E+04	9.53E+03	2.80E+02	9.49E+03	0.63
f_{14}	BFOA-6	1.95E+04	2.12E+04	2.05E+04	4.52E+02	2.06E+04	1.41
	BFOA-30	2.06E+04	2.15E+04	2.11E+04	2.09E+02	2.11E+04	1.31
	-BFOA	1.66E+04	1.99E+04	1.84E+04	6.64E+02	1.84E+04	1.52
	-CDEOA	1.46E+04	1.70E+04	1.60E+04	5.41E+02	1.60E+04	1.12
f_{15}	BFOA-6	1.23E+04	1.27E+04	1.25E+04	1.08E+02	1.25E+04	0.73
	BFOA-30	1.20E+04	1.24E+04	1.22E+04	9.18E+01	1.22E+04	0.95
	-BFOA	9.90E+03	1.15E+04	1.09E+04	2.69E+02	1.09E+04	0.74
	-CDEOA	9.39E+03	1.04E+04	9.87E+03	2.28E+02	9.86E+03	0.76
f_{16}	BFOA-6	2.04E+05	2.04E+05	2.04E+05	1.24E+02	2.04E+05	0.77
	BFOA-30	1.97E+05	1.98E+05	1.98E+05	4.98E+01	1.98E+05	0.45
	-BFOA	1.87E+05	1.98E+05	1.94E+05	2.25E+03	1.94E+05	0.50
	-CDEOA	1.92E+05	2.02E+05	1.98E+05	2.21E+03	1.98E+05	0.48

bucket repository (`https://bitbucket.org/oaltun/opn`). Algorithm 1 does not correspond one to one to CDEOA code given, as we wanted to hide unnecessary details of the framework used. The lines 14-21 in Algorithm 1 summarize what opn does to make given code runnable.

7. CONCLUSION

The experimental studies in this paper were performed on 16 single objective numerical optimization problems taken from IEEE CEC. CDEOA was compared with classical BFOA with 6 and 30 population sizes, and BFOA. Nature-inspired algorithms with small population demonstrated the superiority to the classical counterparts in more complex forms of the classical high dimensional problems in terms of the quality of the final solution and the convergence speed. In particular, overall, CDEOA exhibited better performance than its competitors in unimodal and multimodal functions.
As a future work, the micro algorithm proposed will be improved in terms of quality of final solution to compete with the techniques in large scale optimization field .

8. REFERENCES

[1] Juan C. Fuentes Cabrera and Carlos A. Coello Coello. Handling constraints in particle swarm optimization using a small population size. In *MICAI 2007: Advances in Artificial Intelligence*, pages 41–51. Springer, 2007.

[2] Fabio Caraffini, Ferrante Neri, and Ilpo Poikolainen. Micro-differential evolution with extra moves along the axes. In *Differential Evolution (SDE), 2013 IEEE Symposium on*, pages 46–53. IEEE, 2013.

[3] Ying Chu, Hua Mi, Huilian Liao, Zhen Ji, and Q.H. Wu. A Fast Bacterial Swarming Algorithm for high-dimensional function optimization. In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)*, pages 3135–3140, June 2008.

[4] Sambarta Dasgupta, Arijit Biswas, Swagatam Das, Bijaya K. Panigrahi, and Ajith Abraham. A micro-bacterial foraging algorithm for high-dimensional optimization. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 785–792. IEEE, 2009.

[5] Kalmanje Krishnakumar. Micro-Genetic Algorithms For Stationary And Non-Stationary Function Optimization. volume 1196, pages 289–296, 1990.

[6] J. J. Liang, B. Y. Qu, and P. N. Suganthan. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory*, 2013.

[7] J. J. Liang, P. N. Suganthan, and K. Deb. Novel composition test functions for numerical global optimization. In *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pages 68–75. IEEE, 2005.

[8] Sergio Nesmachnow, Hctor Cancela, and Enrique Alba. A parallel micro evolutionary algorithm for heterogeneous computing and grid scheduling. *Applied Soft Computing*, 12(2):626–639, February 2012.

[9] Mauricio Olguin-Carbajal, Enrique Alba, and Javier Arellano-Verdejo. Micro-differential evolution with local search for high dimensional problems. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 48–54. IEEE, 2013.

[10] Olusegun Olorunda and Andries Petrus Engelbrecht. Differential evolution in high-dimensional search spaces. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 1934–1941. IEEE, 2007.

[11] Konstantinos E. Parsopoulos. Cooperative micro-differential evolution for high-dimensional problems. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 531–538. ACM, 2009.

[12] Konstantinos E. Parsopoulos. Parallel cooperative micro-particle swarm optimization: A masterslave model. *Applied Soft Computing*, 12(11):3552–3579, 2012.

[13] Kevin M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *Control Systems, IEEE*, 22(3):52–67, 2002.

[14] A. K. Qin and Xiaodong Li. Differential evolution on the CEC-2013 single-objective continuous optimization testbed. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1099–1106. IEEE, 2013.

[15] S. Rahnamayan and H.R. Tizhoosh. Image thresholding using micro opposition-based Differential Evolution (Micro-ODE). In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)*, pages 1409–1416, June 2008.

[16] Marco Aurelio Sotelo-Figueroa, Hctor Jos Puga Soberanes, Juan Martn Carpio, Hctor J. Fraire Huacuja, Laura Cruz Reyes, and Jorge Alberto Soria Alcaraz. Evolving bin packing heuristic using micro-differential evolution with indirect representation. In *Recent Advances on Hybrid Intelligent Systems*, pages 349–359. Springer, 2013.

[17] Rainer Storn. On the usage of differential evolution for function optimization. In *Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American*, pages 519–523. IEEE, 1996.

[18] Ponnuthurai N. Suganthan, Nikolaus Hansen, Jing J. Liang, Kalyanmoy Deb, Y.-Po Chen, Anne Auger, and S. Tiwari. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *KanGAL report*, 2005005, 2005.