An Effective Bacterial Foraging Optimization Based on Conjugation and Novel Step-Size Strategies

Ming Chen[1], Yikun Ou[2], Xiaojun Qiu[2], and Hong Wang[2](✉)

[1] College of Economics, Shenzhen University, Shenzhen, China
[2] College of Management, Shenzhen University, Shenzhen, China
ms.hongwang@gmail.com

Abstract. Bacterial Foraging Optimization (BFO) is a high-efficient meta-heuristic algorithm that has been widely applied to the real world. Despite outstanding computing ability, BFO algorithms can barely avoid premature convergence in computing difficult problems, which usually leads to inaccurate solutions. To improve the computing efficiency of BFO algorithms, the paper presents an improved BFO algorithm: Conjugated Novel Step-size BFO algorithm (CNS-BFO). It employs a novel step-size evolution strategy to address limitations brought by fixed step size in many BFOs. Also, the improved BFO algorithm adopts Lévy flight strategy proposed in LPBFO and the conjugation strategy proposed in BFO-CC to enhance its computing ability. Furthermore, Experiment on 24 benchmark functions are conducted to demonstrate the efficiency of the proposed CNS-BFO algorithm. The experiment results suggest that the proposed algorithm can deliver results with better quality and smaller volatility than other meta-heuristics, and hence sufficiently mitigate the limitation of premature convergence facing many meta-heuristics.

Keywords: BFO · Lévy flight · Conjugation · Adaptive step size

1 Introduction

The so-called nature-inspired algorithm has drawn considerable attention since its very beginning. Because of its excellent ability to tackle many optimization problems (especially multi-minimum and multi-constraints problems) [17], meta-heuristic algorithm is considered one of the most effective measures to address difficult real-world problems for researchers. Due to its advantages in solving those problems, meta-heuristic algorithm remains popular in varying fields.

The most popular nature-inspired meta-heuristics include particle swarm optimization algorithm (PSOA) [8], bacterial foraging optimization algorithm (BFOA) [19], ant colony optimization algorithm (ACOA) [5], genetic algorithm (GA) [7], and so on. Nature-inspired algorithms are mostly derived from observations of natural phenomena or biological behaviors. For example, particle swarm optimization algorithm [8] is inspired by swarm characteristics of birds seeking foods, and genetic algorithm [7] simulates the process of natural selection and genetic construction. Up until now, researchers are keen on proposing novel meta-heuristics for different optimizations, such as Novel Multi-Hop algorithm for wireless network [11], Novel Improved Bat algorithm for path planning [10], Genetic Frog Leaping for text document clustering [1].

Performance of nature-inspired meta-heuristics is mostly determined by the process of exploitation and exploration [24], which refer to the convergence to best candidate solutions and the diversion to broader search space separately. Exploitation is introduced to perform local search so as to speed up convergence, usually leading to premature convergence in which the algorithms get stuck in local optima. In contrast to exploitation, exploration centers on seeking various solutions by broadening search area to find sufficient candidate solutions for the problem, usually slowing down the convergence process and wasting computational efforts [20]. Lopsided emphasis on either search process can lead to low-efficiency in computation. Hence researchers have exerted enormous efforts in balancing the exploration and exploitation for meta-heuristics [25].

Bacterial foraging optimization algorithm is proposed by Passino in 2002 [19]. Under observation, researchers found the way that Escherichia coli approaches to nutrients and be eliminated under certain adverse conditions. Based on these observation, Passino built a basic four-stage process in the original BFO algorithm. Since the introduction of BFO algorithm, it has been broadly employed in all sorts of fields, including robotic cells [12], power generation [6], sensor network [4], and so on.

In order to boost the computation efficiency of BFO, numerous improved BFO algorithms have been proposed in recent years. Hybridization of BFO algorithm with other meta-heuristics attracts much attention from researchers. Researchers have proposed hybrid BFO algorithms combined with all sorts of algorithms, such as GABFO [9] and DEBFO [2]. Benefiting from advantages inherent in other meta-heuristics, hybrid BFOs dramatically improve their ability to find solutions with decent accuracy.

Aside from hybridizing BFO with other algorithms, introduction of communication mechanisms to BFOs also proved quite efficient in improving BFOs' computing efficiency. The introduction of communication mechanisms serves a purpose of taking better advantage of solutions found by different bacteria in BFOs, increasing the rate of convergence [16]. In this way, BFOs' ability to find solutions with good quality can be enhanced. Hence some researchers have been studying on introducing effective communication mechanisms to BFOs. For instance, social learning mechanism is incorporated into classical BFO algorithm by Yan et al. in 2012 [22]; Adaptive Comprehensive Learning BFO (ALCBFO) [21] is introduced by L. Tan et al. in 2015 to improve balance between exploitation and exploration in BFO; novel chemotaxis and conjugation strategies (BFO-CC) are employed by Yang et al. [23] to propel the development of BFO.

Besides, improvements on chemotaxis step are frequently studied by researchers. Original BFO is featured with fixed step size, which is unhelpful for BFO to fine-tune solutions. Therefore, by improving the step size of BFOs, accurate solutions are more likely to be found by BFOs. Some of the BFOs with improved step size include Adaptive Bacterial Foraging Optimization (ABFO) proposed by Dasgupta et al. in 2009 [3]. Linear decreasing BFO (BFO-LDC) and Non-linear Decreasing BFO (BFO-NDC) are modeled by Niu et al. in 2011 [14]. Gravitation search strategy is combined with chemotactic step by Zhao et al. in the proposed Effective BFO (EBFO) [26]. Recently, Pang B et al. proposed LPBFO [18], which incorporates Lévy flight to the reconstruction of BFO's step-size, proving quite efficient in mitigating premature convergence in BFOs.

Despite huge efforts exerted by researchers, the problem of premature convergence remain rather tricky. In order to propel the development of BFO algorithm, the paper proposes a Conjugated Novel Step-size BFO algorithm(CNS-BFO) to shed some light on the solution of premature convergence. The proposed algorithm employs a novel chemotaxis step-size evolution strategy that enable bacteria to approach best solutions efficiently. Moreover, the proposed CNS-BFO algorithm employes conjugation strategy proposed in BFO-CC algorithm and Lévy flight strategy proposed in LPBFO to enhance the computation ability of BFOs. The total of the three strategies are combined to mitigate the premature convergence facing BFOs, and hence improve their computational efficiency.

The rest of the paper is organized as follows. Classical BFO algorithm is explained in section two. In section three, the paper introduces the proposed CNS-BFO algorithm. Extensive experiments in both unimodal and multi-modal cases are conducted to test the performance of CNS-BFO algorithm in section four. Finally, section five draw conclusions on the proposed CNS-BFO.

2 Classical BFO Algorithm

Bacterial foraging optimization proposed by Passino in 2002 [19] is inspired by biological behaviors of *E. coli* bacteria. To summarize, classical bacterial foraging optimizations consist of three basic stages: chemotaxis, reproduction, and elimination-dispersal. Subsections below show how the three behaviors work.

2.1 Chemotaxis

E. coli bacteria move towards places of abundant nutrients by rotating flagella. The rotation of flagella can be in either clockwise direction or counter-clockwise direction. Counter-clockwise direction rotation pushes the bacteria to move towards nutrients, while clockwise direction leads to bacteria' s tumbling away from the current position to search more nutrients. Taking both actions, bacteria conduct chemotaxis to approach nutrients gradually.

Suppose bacterium i is denoted as $\theta^i(j,k,l)$, in which j denotes the j-th chemotaxis, k denotes the k-th reproduction, and l denotes the l-th elimination& dispersal, the behaviors of the i-th bacterium can be represented as follows:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{1}$$

where $C(i)$ denotes the step size of chemotaxis and $\varDelta(i)$ denotes random direction vector in which elements are within $[-1, 1]$. If the health status gets better after tumbling, bacteria will keep on swimming for several steps, in order to further approach optimal solutions.

2.2 Reproduction

After the stage of chemotaxis, fitness values of bacteria over the whole stage of chemotaxis are summarized. The smaller the summation of fitness values, the healthier the bacterium. The computation of health status J_{health} can be represented by the following equation:

$$J_{health} = \sum_{i=1}^{S} \sum_{j=1}^{N_c} J(i, j, k, l) \tag{2}$$

where N_c denotes the number of chemotaxis and J is the evaluated value of the health status of the i bacterium at the k reproduction and the l elimination. In classical BFOs, only the first half bacteria ranked by health status can survive, with their offsprings take the place of the other half of bacteria. Reproduction is designed to strengthen local search ability of BFO while maintaining stable population size of bacteria. The process of reproduction can be represented as follows:

$$\theta^{i+Sr}(j, k, l) = \theta^{i}(j, k, l) \tag{3}$$

where Sr is half of the population size.

2.3 Elimination & Dispersal

To improve global search ability, classical BFOs introduce the stage of elimination & dispersal. After stages of chemotaxis and reproduction, some bacteria face elimination triggered by adverse situations by a possibility P_{ed}. Bacteria will be dispersed to random places within the search range. By introducing stage of elimination & dispersal, the diversity of bacteria is improved and results in stronger global search ability.

3 Conjugated Novel Step-Size BFO Algorithm

In this Section, features of the proposed CNS-BFO are introduced. Pseudocode for the proposed BFO algorithm is shown in Algorithm 1.

3.1 Lévy Flight Step Length Strategy

Levy distribution, proposed by Lévy Paul [13], is an inverse-gamma distribution, whose formula can be expressed as $P_s = s^{-\lambda}$, in which s is a step size and λ is between 1 and 3. Subject to Lévy distribution, Lévy flight is a sort of random moving which can generate frequent small step sizes and occasional big step sizes for bacteria. The characteristics of Lévy flight enable bacteria to move with ununiformed step size, strengthening BFO's capability of exploitation and exploration at the same time: on the one hand, frequent small step sizes fortify the local search ability of bacteria, whereby the exploitation ability of the algorithm can be improved. On the other hand, occasional big step sizes prevent bacteria from over concentrating on exploiting the current position, guiding bacteria to a broader search area, whereby the exploration ability of BFO can be strengthened.

The paper adopts the Lévy flight strategy modified by Bao Pang et al. in 2018 [18]. The strategy can be represented by the equation below:

$$C^{'}(i) = \frac{\alpha}{t(i)}|\frac{u}{|v|^{\frac{1}{\beta}}}| \tag{4}$$

where α is a parameter for controlling the strength of change in step-size, $t = j + (k-1)N_c + (l-1) \cdot rand$. β is a random number between 0.3 and 1.99. u and v are two normal random variables with means equal to zero and with variances of σ_u and σ_v respectively. σ_u and σ_v are calculated as follows:

$$\sigma_u = \frac{\Gamma(1+\beta)sin(\frac{\pi\beta}{2})}{\Gamma[\frac{(1+\beta)}{2}]2^{\frac{\beta-1}{2}}\beta} \quad \sigma_v = 1 \tag{5}$$

Apparently, with bacteria go through more chemotaxis, the step sizes will dwindle as $t(i)$ gets greater, and hence convergence can be accelerated at later stages.

3.2 Novel Step Evolution Strategy

In the original BFO algorithm, chemotaxis step size is set fixed for each bacterium. Fixed step size saps bacterium's strength to exploit, leading to premature convergence easily. To address the limitation, a novel step-size evolution strategy is applied here. After bacteria being ranked and reproduced, step lengths of bacteria will be updated by the following equation:

$$C^{''}(i, k+1) = C(i, k+1)\boldsymbol{b} \tag{6}$$

$$C(i, k+1) = [C^{''}(i, k+1) - C(i, k+1)] \cdot (1 - \frac{1}{lk+1}) \tag{7}$$

where $\boldsymbol{b}$ denotes a $p \times 1$ vector whose elements are all 1, p is the dimension of bacteria. This novel step-size evolution strategy is introduced to make bacteria move adaptively and efficiently. As moving step sizes significantly impact the ability of bacteria to search glocal solution, careful moving can dramatically improve the odds of finding the best solution. The novel step-size evolution strategy allows bacteria to move with bigger step sizes at the beginning, and allow them to fine-tune solutions at later stage with smaller step size. Combined with Levy-flight step length strategy, the adaptive moving strategy can efficiently mitigate the problem of premature convergence.

3.3 Conjugation Strategy

Conjugation strategy proposed in BFO-CC algorithm by Cuicui Yang et al. in 2016 [23] is a novel strategy simulating the process of exchanging genetic makeup among creatures. The process of conjugation is illustrated in Fig. 1. The proposed BFO-CC algorithm introduces conjugation mechanism to bacteria, providing a channel for bacteria to communicate among each other, so that the diversity of bacteria can be improved. In this way, conjugation strategy enhances bacteria's capability of searching for global best. In BFO-CC algorithm, each bacterium chooses a bacterium $\theta^{i'}$ and a conjugation point pt randomly to perform conjugation. The equation for conjugation mechanism is given as follows:

$$\theta^i_{new} = \theta^i(j,k,l) + \boldsymbol{a} \cdot [\theta^{i'}(j,k,l) - \theta^i(j,k,l)] \tag{8}$$

where $\boldsymbol{a}$ is a p-dimension vector in which the values of the elements from pt to $pt + L - 1$ are random numbers uniformly distributed from 0 to 1, with the rest of elements being 0. L indicates the length of exchange of dimensions. In this way, learning process is made more flexible and adaptable, for bacterium can not only learn from other bacterium, but also learn with varying strength on varying dimensions. By taking the conjugation strategy, the bacterium can take advantage of solutions found by each other, whereby avoid premature convergence.

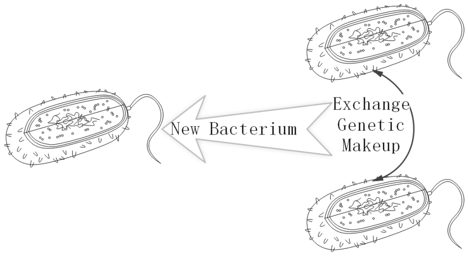

Fig. 1. Conjugation of bacteria

4 Experiments and Analysis

4.1 Experiment Setting

To prove the efficiency of the proposed CNS-BFO, it is to be tested along with other sorts of meta-heuristics: GA [7] and PSO [8]. Besides, as a variant of BFO, CNS-BFO is also to be compared with standard BFO [19] and several popular BFO variants: BFO-LDC [14], BFO-NDC [14], and SRBFO [15] algorithms.

To make sure that the comparisons are fair, algorithms are set to initialize at the same initial positions. The population sizes for the algorithms are set to 50.

Algorithm 1. Pseudocode for CNS-BFO Algorithm

```
1: Initialize parameters
2: for $l$ = 1,2,...,$N_{ed}$ do
3:   for $k$ = 1,2,...,$N_{re}$ do
4:     for $j$ = 1,2,...,$N_c$ do
5:       for $i$ = 1,2,...,$S$ do
6:         Compute $J^i$, and set $J_{last} = J^i$
7:         Tumble by equation (4), then compute $J^i$
8:         while $m < N_s$ do
9:           if $J^i < J_{last}$ then
10:            Update $\theta^i$
11:          else
12:            Stop
13:          end if
14:        end while
15:        if $rand < p_{con}$ then
16:          Compute $\theta_{new}$ by equation (8)
17:          if $J^i < J_{last}$ then
18:            $\theta^i = \theta_{new}$
19:          end if
20:        end if
21:      end for
22:    end for
23:    Compute the health status $J^i$ and sort bacteria by $J^i$
24:    for $j$ =1 to $Sr$ do
25:      split
26:      for $e$ = 1 to $N_n$ do
27:        Update $C_{i,k+1}$ by equation (6) and (7)
28:      end for
29:    end for
30:  end for
31:  for m = 1,2,...,S do
32:    Eliminate each bacterium if $rand < p_{ed}$
33:  end for
34: end for
```

The total number of FEs is set to 10,000. All the algorithms are tested for 30 times independently on all the test functions. Furthermore, parameters of PSO, GA, BFO, BFO-LDC, BFO-NDC and SRBFO are set as recommended at their source papers, as referred to [7,8,14,14,15,19] respectively. Parameters of CNS-BFO algorithms are set as follows: N_c = 1,000, N_{re} = 10, N_{ed} = 2, P_{ed} = 0.25, $Sr = \frac{S}{2}$, n = 30, s = 2, and g = 2, α = 1,000, β = 1.5, p_{con} = 0.2, N_n = 30, L = 2. To comprehensively test the efficiency of CNS-BFO, 24 popular test functions are used in the experiment. Test functions consist of 10 unimodal function ($f_1 - f_{10}$), 10 multi-modal function ($f_{11} - f_{20}$), and 4 2-dimensional multi-modal function ($f_{21} - f_{24}$). Generally multi-modal test functions are considered more difficult than unimodal functions to solve, since algorithms are easier to be trapped in multiple local optima in the case of multi-modal functions. Benchmark functions included in the experiment are shown in Table 1.

Table 1. Test functions: name, search range, and global optimum

Fun	Name	Search range	f_{opt}	Fun	Name	Search Range	f_{opt}
f_1	Sphere	$[-100, 100]^D$	0	f_{13}	Alpine N. 1	$[0, 10]^D$	0
f_2	Sum Squares	$[-10, 10]^D$	0	f_{14}	Xin-She Yang	$[-5, 5]^D$	0
f_3	Powell Sum	$[-1, 1]^D$	0	f_{15}	Xin-She Yang N. 2	$[-2\pi, 2\pi]^D$	0
f_4	Schwefel 2.20	$[-100, 100]^D$	0	f_{16}	Levy	$[-10, 10]^D$	0
f_5	Schwefel 2.21	$[-100, 100]^D$	0	f_{17}	Periodic	$[-10, 10]^D$	0.9
f_6	Schwefel 2.22	$[-100, 100]^D$	0	f_{18}	Quartic	$[-1.28, 1.28]^D$	0
f_7	Schwefel 2.23	$[-10, 10]^D$	0	f_{19}	Salomon	$[-100, 100]^D$	0
f_8	Griewank	$[-600, 600]^D$	0	f_{20}	Powell	$[-4, 5]^D$	0
f_9	Zakharov	$[-5, 10]^D$	0	f_{21}	Schaffer N. 1	$[-100, 100]^D$	0
f_{10}	Dixon-Price	$[-10, 10]^D$	0	f_{22}	Bohachevsky N. 1	$[-100, 100]^D$	0
f_{11}	Ackley	$[-32, 32]^D$	0	f_{23}	Bohachevsky N. 2	$[-100, 100]^D$	0
f_{12}	Rastrigin	$[-5.12, 5.12]^D$	0	f_{24}	Eggcrate	$[-5, 5]^D$	0

4.2 Experiment Results and Analysis

Means and standard deviations of the optima attained by algorithms are showcased in Table 2. Note that the best results have been marked with boldface, and that all of the actual fitness values have been taken the log of to contrast performance of different algorithms. Hence when certain algorithm reaches the optimal value 0 before maximum FEs, its convergence curve seems to "disappear" for the remaining iterations. Also, Fig. 2 reveals convergence graphs on a part of test functions. Only convergence graphs for f_1, f_8, f_{10}, f_{16} are displayed, for the convergence graphs for remaining functions are considered quite similar to the four selected graphs: graph for f_1 represents those for function 1, 2, 3, 4, 5, 6, 7, 11, 13, 14, 18, 19, 20, 24; graph for f_8 represents those for function 12, 21, 22, 23; graph for f_{10} represents those for function 15, 17; graph for f_{16} represents that for function 19.

Clearly, the proposed CNS-BFO outperforms other meta-heuristics in most cases. Table 2 reveals dramatic advantages of CNS-BFO as compared to other algorithms. To begin with, in 23 out of 24 cases, means of best results obtained by CNS-BFO are superior to those by other algorithms. Take f_1 for example, the mean of best result obtained by CNS-BFO is $1.24E^{-50}$, while the best results of other algorithms at most reach $3.56E^{-02}$, a level far subordinate than the result attained by CNS-BFO. Even in cases where CNS-BFO do not attain sufficient advantage (such as f_{17}), it still has the edge over the its counterparts. Besides, it is noticeable that CNS-BFO are exceptionally good at low-dimension functions

Table 2. Means and standard deviations on f_1-f_{24} over 30 runs

Func	PSO		GA	BFO	BFO-LDC	BFO-NDC	SRBFO	CNS-BFO
f_1	Mean	3.56E−02	3.08E+01	8.42E−02	7.48E+02	6.31E−01	7.51E−02	**1.24E−50**
	Std.	6.17E−02	2.12E+01	1.33E−02	1.70E+02	1.22E−01	1.42E−02	**3.71E−50**
f_2	Mean	2.82E+00	2.94E+00	7.32E−01	7.69E−03	5.27E+00	5.34E−01	**2.69E−47**
	Std.	6.17E+00	1.99E+00	1.30E−01	1.47E−03	9.01E−01	8.18E−02	**1.45E−46**
f_3	Mean	2.84E−03	9.05E−07	2.47E−04	8.44E−07	3.49E−01	1.17E−04	**2.67E−54**
	Std.	4.50E−03	1.34E−06	8.86E−05	3.68E−07	2.72E−01	5.11E−05	**1.46E−53**
f_4	Mean	3.45E+01	9.80E+00	1.12E+00	1.34E+02	3.00E+00	1.01E+00	**9.46E−25**
	Std.	8.72E+01	4.42E+00	9.86E−02	1.93E+01	2.99E−01	8.95E−02	**4.42E−24**
f_5	Mean	8.42E−01	9.73E+00	1.64E+00	3.07E+01	6.84E−01	2.04E−01	**8.90E−20**
	Std.	4.35E−01	3.44E+00	1.11E+00	1.34E+00	1.05E−01	2.57E−02	**2.48E−19**
f_6	Mean	1.55E+12	2.33E+02	3.33E+08	3.67E+09	8.07E+04	1.37E+04	**1.20E−24**
	Std.	2.23E+12	7.06E+01	8.82E+08	1.16E+10	1.40E+05	2.55E+04	**6.49E−24**
f_7	Mean	3.54E−05	5.61E+00	7.39E−08	1.37E−17	3.00E−03	2.27E−08	**2.21E−218**
	Std.	1.15E−04	1.55E+01	5.08E−08	1.02E−17	3.91E−03	2.10E−08	**0**
f_8	Mean	6.95E+00	9.19E−01	1.48E−02	1.31E+00	7.83E−02	2.80E+00	**0**
	Std.	4.52E−15	2.33E−01	7.95E−03	1.08E+00	1.36E−02	7.83E−01	**0**
f_9	Mean	3.45E+00	1.33E+01	1.53E+00	2.51E+02	8.06E+01	8.02E−01	**1.26E−04**
	Std.	7.33E+00	5.76E+00	3.59E−01	8.51E+02	2.67E+01	1.86E−01	**4.19E−04**
f_{10}	Mean	1.66E+01	1.09E+05	1.40E+00	**6.52E−01**	1.11E+01	1.22E+00	6.67E−01
	Std.	3.70E+01	2.82E+05	1.51E−01	**1.12E−01**	3.42E+00	9.64E−02	2.38E−16
f_{11}	Mean	2.62E+00	1.03E+00	7.92E−01	9.37E−01	2.83E+00	5.56E−01	**2.66E−15**
	Std.	6.32E−01	6.49E−01	1.23E−01	9.46E−01	2.40E−01	7.52E−02	**0**
f_{12}	Mean	4.54E+01	2.42E+01	5.37E+01	6.18E+01	1.29E+02	2.45E+01	**0**
	Std.	1.56E+01	8.39E+00	7.31E+00	1.16E+01	1.20E+01	2.38E+00	**0**
f_{13}	Mean	3.67E+00	9.70E−01	3.44E+00	2.78E+00	9.60E+00	1.69E+00	**4.76E−27**
	Std.	1.86E+00	6.25E−01	6.08E−01	1.98E+00	1.92E+00	4.14E−01	**1.81E−26**
f_{14}	Mean	3.81E+02	1.25E−01	8.49E+00	2.55E+02	1.32E+02	4.88E−03	**2.17E−25**
	Std.	2.06E+03	2.51E−01	7.04E+00	2.71E+02	9.36E+01	2.02E−03	**8.93E−25**
f_{15}	Mean	8.30E−04	2.82E−05	6.26E−05	1.17E−05	3.53E−03	8.93E−06	**5.73E−06**
	Std.	6.01E−04	7.51E−07	1.66E−05	1.22E−06	2.20E−03	8.82E−07	**2.65E−09**
f_{16}	Mean	6.55E+00	5.81E+01	1.68E+01	1.65E+01	2.92E+00	1.49E+01	**4.37E−06**
	Std.	4.73E+00	9.53E+01	3.81E+00	3.46E+00	1.21E+00	3.29E+00	**1.40E−05**
f_{17}	Mean	1.00E+00	1.01E+00	1.08E+00	1.00E+00	1.50E+00	1.07E+00	**9.13E−01**
	Std.	4.13E−03	1.02E−02	1.15E−02	1.40E−04	8.80E−02	1.62E−02	**3.46E−02**
f_{18}	Mean	2.08E−02	1.76E+00	1.23E−02	1.48E−06	5.85E−01	7.19E−03	**5.81E−89**
	Std.	5.02E−02	1.74E+00	4.32E−03	4.29E−07	2.10E−01	2.23E−03	**2.28E−88**
f_{19}	Mean	9.59E−01	1.21E+00	1.54E+01	1.52E+01	1.15E+01	1.37E+01	**9.99E−02**
	Std.	2.01E−01	3.46E−01	4.66E−02	5.24E−01	7.04E−01	1.23E+00	**2.62E−02**

(*continued*)

Table 2. (*continued*)

Func	PSO		GA	BFO	BFO-LDC	BFO-NDC	SRBFO	CNS-BFO
f_{20}	Mean	1.83E−02	1.89E+03	5.37E−01	5.82E−03	8.84E+00	3.34E−01	**8.73E−27**
	Std.	3.28E−02	2.07E+03	1.33E−01	1.38E−03	2.82E+00	8.04E−02	**4.78E−26**
f_{21}	Mean	2.26E−04	2.34E−01	1.25E−01	9.00E−02	9.03E−02	6.69E−03	**0**
	Std.	4.90E−04	2.02E−01	4.33E−02	4.53E−02	4.05E−02	2.02E−02	**0**
f_{22}	Mean	3.56E−01	2.07E+03	6.04E−06	8.33E−08	1.92E−04	3.75E−06	**0**
	Std.	2.12E−01	2.78E+03	6.64E−06	7.33E−08	2.20E−04	3.15E−06	**0**
f_{23}	Mean	2.48E−01	1.41E+03	5.72E−06	4.83E−08	1.98E−04	3.25E−06	**0**
	Std.	8.89E−02	2.40E+03	5.10E−06	4.17E−08	2.85E−04	2.60E−06	**0**
f_{24}	Mean	1.50E−03	2.14E+01	3.16E+00	2.21E+00	1.44E−04	5.08E−06	**8.02E−54**
	Std.	1.19E−03	1.91E+01	4.55E+00	4.08E+00	1.61E−04	5.49E−06	**3.37E−53**

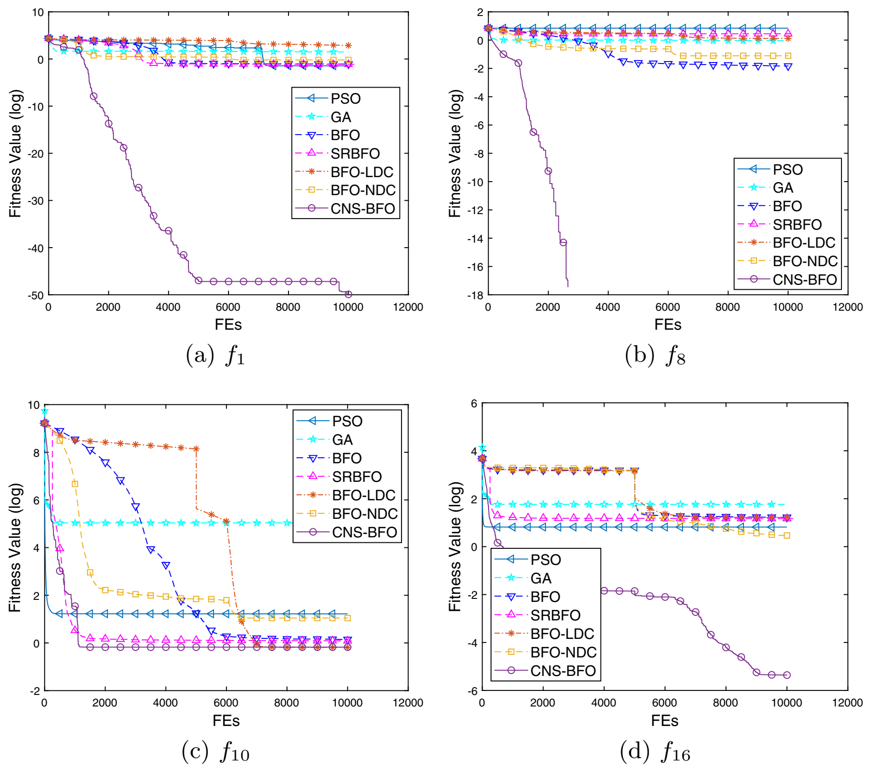

Fig. 2. Convergence graphs on f_1, f_8, f_{10}, f_{16} over 30 runs

(from f_1 to f_{24}), in which CNS-BFO obtained the optimal solution in three out of four functions. The same conclusion can be drawn by looking at Fig. 2. Convergence graph for f_8 suggests that CNS-BFO can rapidly converge to best solution in some cases. In graph for f_1 and f_{16}, even though CNS-BFO fails to achieve best results over the 10000 iterations, it can effectively approach far better solutions than its counterparts. Both the figure and table show that CNS-BFO can effectively obtain good solutions in both unimodal and multi-modal problems.

Furthermore, CNS-BFO also achieves satisfying results in the aspect of stability. Table 2 suggests that performance of CNS-BFO comes with very small volatility in most cases, which can be measured by the standard deviations of results over 30 runs. Over the 24 test functions, CNS-BFO has sufficiently smaller standard deviations than other meta-heuristics. Thus, it proves that the proposed CNS-BFO has strong stability in solving complicated problems.

Overall, the proposed CNS-BFO is proved sufficient enough to address both multidimensional problems and unimodal problems. It is also proved to be quite competitive in low-dimension problems. Also, low volatility of the results computed by CNS-BFO shows that it is capable of delivering results with strong stability.

5 Conclusion

Conjugated Novel Step-size BFO algorithm (CNS-BFO) is proposed to improve the performance of BFO algorithm. Improvements lie in the way of constructing a novel step evolution strategy and incorporating both Lévy flight step-size strategy and conjugation strategy. By modifying the step-size strategy and introducing a learning mechanism, CNS-BFO strikes a good balance between exploitation and exploration, whereby significantly mitigates the problem of premature convergence in BFO algorithms. Benchmark functions tests are conducted to compare the performance of the CNS-BFO algorithm with that of other meta-heuristics. The experiment results prove that the proposed CNS-BFO algorithm significantly can deliver solutions with good quality and stability in both unimodal and multi-modal cases, revealing its outstanding capability of solving difficult problems. Future work might focus on the the improvement of the proposed algorithm or its applications in some areas, such as feature selection.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Grant No. 71901152), Natural Science Foundation of Guangdong Province (2018A 030310575), Natural Science Foundation of Shenzhen University (85303/00000155), Project supported by Innovation and Entrepreneurship Research Center of Guangdong University Student (2018A073825), and Research Cultivation Project from Shenzhen Institute of Information Technology (ZY201717).

References

1. Alhenak, L., Hosny, M.: Genetic-frog-leaping algorithm for text document clustering. Comput. Mater. Contin. **61**, 1045–1074 (2019)

2. Biswas, A., Dasgupta, S., Das, S., Abraham, A.: A synergy of differential evolution and bacterial foraging optimization for global optimization. Neural Netw. World **17**(6), 607 (2007)
3. Dasgupta, S., Das, S., Abraham, A., Biswas, A.: Adaptive computational chemotaxis in bacterial foraging optimization: an analysis. IEEE Trans. Evol. Comput. **13**(4), 919–941 (2009)
4. Deepa, S.R., Rekha, D.: Bacterial foraging optimization-based clustering in wireless sensor network by preventing left-out nodes. In: Mandal, J.K., Sinha, D. (eds.) Intelligent Computing Paradigm: Recent Trends. SCI, vol. 784, pp. 43–58. Springer, Singapore (2020). https://doi.org/10.1007/978-981-13-7334-3_4
5. Dorigo, M., Di Caro, G.: Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 2, pp. 1470–1477. IEEE (1999)
6. Hernández-Ocaña, B., Hernández-Torruco, J., Chávez-Bosquez, O., Calva-Yáñez, M.B., Portilla-Flores, E.A.: Bacterial foraging-based algorithm for optimizing the power generation of an isolated microgrid. Appl. Sci. **9**(6), 1261 (2019)
7. Holland, J.H., et al.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge (1992)
8. Kennedy, J.: Particle swarm optimization. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 760–766. Springer, Boston (2010). https://doi.org/10.1007/978-0-387-30164-8_630
9. Kim, D.H., Abraham, A., Cho, J.H.: A hybrid genetic algorithm and bacterial foraging approach for global optimization. Inf. Sci. **177**(18), 3918–3937 (2007)
10. Lin, N., Tang, J., Li, X., Zhao, L.: A novel improved bat algorithm in UAV path planning. J. Comput. Mater. Contin. **61**, 323–344 (2019)
11. Liu, Y., Yang, Z., Yan, X., Liu, G., Hu, B.: A novel multi-hop algorithm for wireless network with unevenly distributed nodes. Comput. Mater. Contin. **58**(1), 79–100 (2019)
12. Majumder, A., Laha, D., Suganthan, P.N.: Bacterial foraging optimization algorithm in robotic cells with sequence-dependent setup times. Knowl.-Based Syst. **172**, 104–122 (2019)
13. Mantegna, R.N.: Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. Phys. Rev. E **49**(5), 4677 (1994)
14. Niu, B., Fan, Y., Wang, H., Li, L., Wang, X.: Novel bacterial foraging optimization with time-varying chemotaxis step. Int. J. Artif. Intell. **7**(A11), 257–273 (2011)
15. Niu, B., Bi, Y., Xie, T.: Structure-redesign-based bacterial foraging optimization for portfolio selection. In: Huang, D.-S., Han, K., Gromiha, M. (eds.) ICIC 2014. LNCS, vol. 8590, pp. 424–430. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09330-7_49
16. Niu, B., Liu, J., Bi, Y., Xie, T., Tan, L.: Improved bacterial foraging optimization algorithm with information communication mechanism. In: 2014 Tenth International Conference on Computational Intelligence and Security, pp. 47–51. IEEE (2014)
17. Niu, B., Liu, J., Wu, T., Chu, X., Wang, Z., Liu, Y.: Coevolutionary structure-redesigned-based bacterial foraging optimization. IEEE/ACM Trans. Comput. Biol. Bioinform. **15**(6), 1865–1876 (2017)
18. Pang, B., Song, Y., Zhang, C., Wang, H., Yang, R.: Bacterial foraging optimization based on improved chemotaxis process and novel swarming strategy. Appl. Intell. **49**(4), 1283–1305 (2018). https://doi.org/10.1007/s10489-018-1317-9

19. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst. Mag. **22**(3), 52–67 (2002)
20. Sahib, M.A., Abdulnabi, A.R., Mohammed, M.A.: Improving bacterial foraging algorithm using non-uniform elimination-dispersal probability distribution. Alexandria Eng. J. **57**(4), 3341–3349 (2018)
21. Tan, L., Lin, F., Wang, H.: Adaptive comprehensive learning bacterial foraging optimization and its application on vehicle routing problem with time windows. Neurocomputing **151**, 1208–1215 (2015)
22. Yan, X., Zhu, Y., Zhang, H., Chen, H., Niu, B.: An adaptive bacterial foraging optimization algorithm with lifecycle and social learning. Discrete Dyn. Nat. Soc. **2012** (2012)
23. Yang, C., Ji, J., Liu, J., Yin, B.: Bacterial foraging optimization using novel chemotaxis and conjugation strategies. Inf. Sci. **363**, 72–95 (2016)
24. Yang, X.S.: Nature-Inspired Optimization Algorithms. Elsevier, Amsterdam (2014)
25. Yang, X.S., Deb, S., Fong, S.: Metaheuristic algorithms: optimal balance of intensification and diversification. Appl. Math. Inf. Sci. **8**(3), 977 (2014)
26. Zhao, W., Wang, L.: An effective bacterial foraging optimizer for global optimization. Inf. Sci. **329**, 719–735 (2016)