Research Article | **Open Access**

A Novel Strategy Adaptation Based Bacterial Foraging Algorithm for Numerical Optimization

Chin-Ling Lee[1] and Cheng-Jian Lin[2]*

[1]Department of International Business, National Taichung University of Science and Technology, Taichung City 404, Taiwan
[2]Department of Computer Science and Information Engineering National Chin-Yi University of Technology, Taichung City 41170, Taiwan

Abstract

In this paper, a strategy-adaptation-based bacterial foraging optimization (SABFO) algorithm is proposed to solve the optimization of complex problems. The proposed SABFO algorithm adopts the strategic approach into chmotaxis step of traditional bacterial foraging optimization (BFO). The proposed method makes each bacterium swim on different run-lengths, and increases bacterial diversity as well. Five optimization problems of nonlinear benchmark functions are used to verify the performance of SABFO. Simulation results show that the SABFO obtains better global optimal solutions than other methods.

Keywords: Bacterial foraging optimization; Strategy evolution; Numerical optimization

Introduction

Recently, soft computing methods [1-3], such as fuzzy logic, artificial neural networks, and evolutionary computation, have been capable of handling and tackling a wide range of real-world application problems in society and nature. The evolutionary algorithms are computational models of evolutionary computation, which simulate the natural evolution processes based on Darwinian principle. Genetic algorithm (GA) [4,5] and differential evolution (DE) [6] are two popular evolutionary algorithms that optimize a problem by iteratively improving candidate solutions in terms of a given measurement of objective function. These algorithms can efficiently explore a large global solution space, but they may trap into local minimums and not guarantee an optimal solution being ever found [7]. Therefore, some researchers [8-10] have proposed various improved-GAs to solve global optimization problems.

In artificial intelligence, swam intelligence (SI), a subset of evolutionary computation, is the collective behavior in a decentralized and self-organized system, which is made up by a population of simple individuals interacting locally with one another and with their environments. Inspired by the social behavior of animals, such as bird flocking, fish schooling and swarm theory, Kennedy and Eberhart [10] developed a new optimization algorithm called particle swarm optimization (PSO) in 1995, which has come to be widely used as a problem solving method in engineering and computer science. PSO is easy to understand and implement, and it requires less computational bookkeeping. However, one major drawback of the PSO is its slow convergence speed. Other algorithms inspired by biological systems and natural phenomena were also proposed, such as ant colony optimization (ACO) [11], bacterial foraging optimization (BFO) [12], artificial immune system (AIS), fish schooling search (FSS), fireworks algorithms (FWA).

A number of studies have been aimed to improve the evolutionary algorithm. Some researchers combined two different evolutionary algorithms to create new means of solving the engineering problems. For example, the combination of DE based mutation operator and bacterial chemotaxis was made to solve the nonlinear benchmark functions [13], the conjunction of the foraging behavior of *E. coli* bacteria and PSO was utilized to tune PID controller [8], and the improved method of introducing a PSO-like swarming mechanism into chemotaxis step of BFO was implemented in numerical optimization [7].

BFO is inspired by the foraging behavior of *E. coli* found in the intestines, and the advantage of this algorithm is the parallelizable searching ability of swarm intelligence, which tends to jump out of local minimum values. In complex optimization problems, however, BFO is hard to converge to the optimal solution and its performance heavily depends on the chemotaxis step length. Thus, Yan et al. [9] proposed social cooperation and adaptive step size strategies, which guided the bacteria tumbling towards better directions. Majhi et al. [14] adaptively adjusted the run-length of bacterial and applied it into two new forecasting models. Chen et al. [15,16] improved the convergence rate of BFO by verifying the influence of the size of bacteria run-length on the execution results. Besides, studies on mathematical analyses of the chemotaxis step [17] and reproduction step [18] have been proposed.

The run-length of original BFO is fixed, which makes it easy to fall into local optimal solution in complex problem. In this paper, we propose an improved algorithm, strategy-adaptation-based bacterial foraging optimization (SABFO), which adopts the strategy adaptation method in chemotaxis step. This adaptation method assesses three successive fitness values of each bacterium in the searching direction, and adjusts the run-length parameter, based on our newly-designed mathematical formulas.

The rest of this paper is organized as follows. In section 2, we introduce the traditional Bacterial Foraging Optimization. Section 3 is devoted to the proposed SABFO, and section 4 to its performance testing on a set of benchmark functions. Finally, discussion and conclusion are drawn in Sections 5 and 6, respectively.

Review of Traditional Bacterial Foraging Optimization

The *E. coli* bacterium, the most famous bacteria in the intestinal tract

of humans and animals, has two movement patterns, swim and tumble, achieved by the rotation of a set of tensile flagella. The movements of bacterium can help it avoid noxious substances and push its body towards abundant nutrient region while foraging. According to the studies of modern biology, *E. coli* bacterium grows very fast in a suitable environment, which can carry the reproduction out in every twenty minutes. By simulating the foraging process of bacteria, Passino [12] proposed the Bacterium Foraging Optimization (BFO) algorithm (Figure 1).

BFO employs three dominant mechanisms, chemotaxis, reproduction and elimination-dispersal, to solve the optimization problems. The followings describe these mechanisms in detail and the complete flowchart is shown in Figure 1.

Chemotaxis

Chemotaxis is the behavior of bacteria gathering to nutrient concentration in environment, and it consists of a tumble and several swims. The unit step of the movement in any direction is defined as the tumble. When bacteria complete a tumble, if the fitness value of the new position has improved, it will continue to move several steps in the same direction. This process is defined as the swim, and it continues until the fitness value gets worse or the number of movement reaches the predetermined threshold.

The location of a bacterium is the solution of optimization problem, which can be expressed as a D-dimensional vector, $\theta^i = \left[\theta_1^i, \theta_2^i, \dots \theta_D^i \right], i = 1, 2, \dots, S$, where S represents the population size of bacteria. The chemotaxis step formula is expressed as follows:

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

where C_i is the run-length of the bacteria tumble movement, $\theta^i(j,k,l)$ represents the location of *i-th* bacterium at *j-th* chemotaxis, *k-th* reproduction and *l-th* elimination-dispersal steps, and $\Delta(i)$ is the randomly selected direction, whose elements lies in [-1, 1].

The population of bacteria can be written as $P(j,k,l) = \left\{ \theta^i(j,k,l) / i = 1, 2, \dots S \right\}$, and each cluster message signaling among bacteria is represented as the following equation:

$$J_{cc}(\theta, P(j,k,l)) = \sum J_{cc}(\theta, \theta(j,k,l)) = \quad (2)$$
$$\sum_{i=1}^{s} \left[-d_{attractant} \exp\left(-W_{attractant} \sum_{n=1}^{D} (\theta_n - \theta_n^i)^2 \right) \right] + \sum_{i=1}^{S} \left[h_{repellent} \sum_{n=1}^{D} (\theta_n - \theta_n^i)^2 \right]$$

Where four parameters, $d_{attractant}, w_{attractant}, h_{repellent}$ and $w_{repellent}$ and, are chosen appropriately to describe the attractant and repellent behavior of bacteria, and J_{cc} is the updated value of fitness function. The new bacteria fitness function is as follows:

$$J(i, J+1, k, l) = J(i, j, k, l) + J_{cc}\left(\theta^i(J+1, k, l), P(J+1, k, l) \right) \quad (3)$$

If the fitness value of bacterium gets improved and the maximum number of swim length (*Ns*) has not been reached, a new run will take place; otherwise, the next bacterium will be dealt with.

Reproduction

At the end of the above process, the times of chemotaxis reached the predefined threshold, so that the bacteria would conduct the reproduction process. In this process, poor half of the bacteria will die, and the others will survive and each will split into two sub-bacteria of equal part. The sub-bacteria inherit the parent-bacteria's characteristics of biological, including the same position and run-length. Moreover, the population size of bacteria remains constant after the reproduction process. The health of bacteria is defined as the following:

$$J_{health}^i = \sum_{J=1}^{N_C+1} J(i, j, k, l) \quad (4)$$

where N_c is the number of chemotactic steps and J_{health}^i cumulate the bacterial fitness value for each chemotaxis step. Thus, the higher the J_{health}^i expressed the lower degree of health; on the contrary, the higher the health degree.
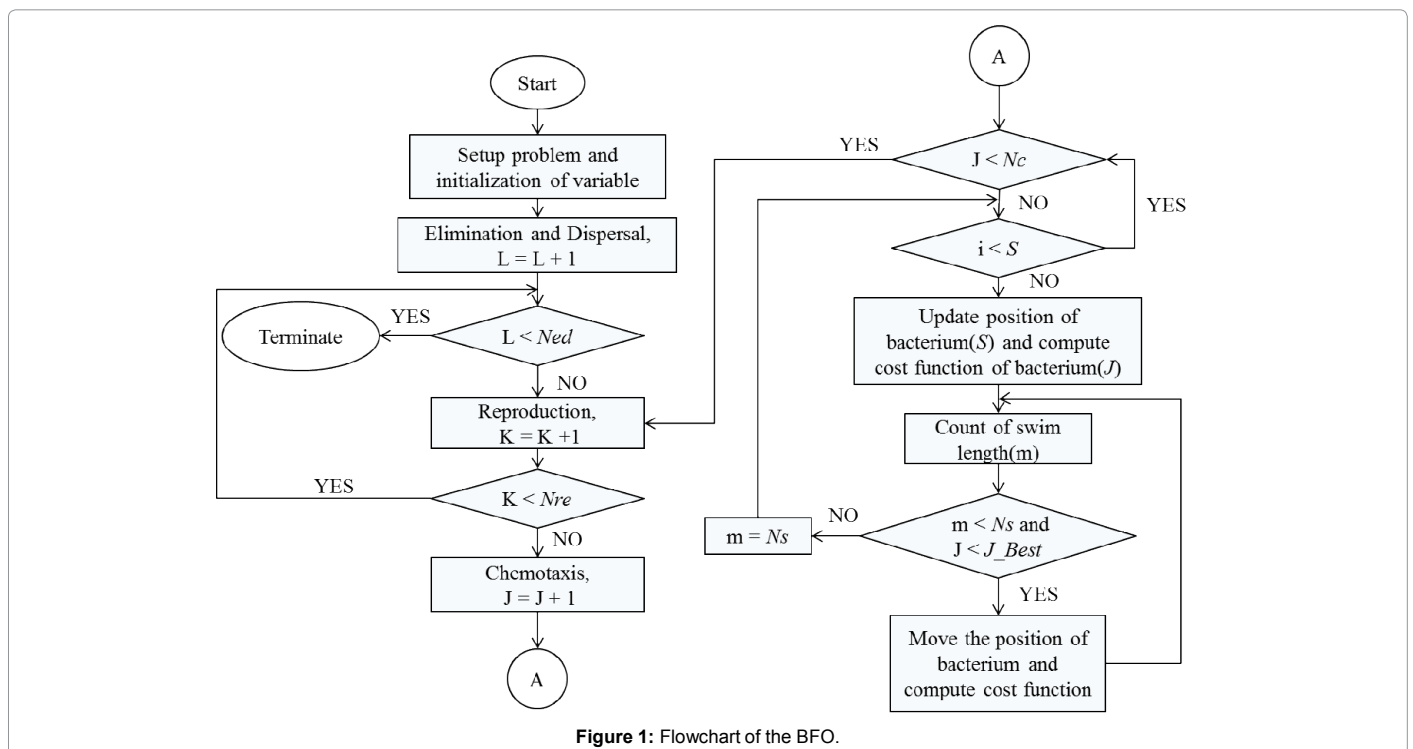

Figure 1: Flowchart of the BFO.

Elimination dispersal

The chemotaxis process ensures the local search ability of bacteria, and the reproduction process can accelerate their search speed. However, when dealing with complex optimization problems, the chemotaxis process is unable to avoid bacteria trapping into local minimum. Therefore, elimination-dispersal process strengthens the ability of global optimization of the BFO algorithm. The dispersal-elimination process will conduct after every N_{re} times of reproduction steps. If the random number, generated between 0 and 1 for each bacterium, is less than a predetermined threshold, the bacterium will be eliminated and a new bacterium is generated in arbitrary position within the search space. Consequently, the chemo taxis step can leap from the local optimum to find the global optimum solution.

The Proposed Strategy-adaptation-based Bacterial Foraging Optimization

In this study, a Strategy-adaptation-based Bacterial Foraging Optimization (SABFO) is proposed. SABFO adopts the strategy adaptation method in the chemotaxis step of traditional BFO algorithm to solve the problems of prolonged execution time in multi-dimensional problem and trap into local optimal solution problem. The strategy concept was derived from Cooren et al. which used independent Gassians and pivot strategy to improve the performance of PSO [19,20]. Besides, it has been researched that the bacterial run-length size will affect the solution found in the optimization problem [21]. When bacteria are farther away from the best global optimal solution, the run-length size needs to be set larger; otherwise, the size needs to be set smaller. Therefore, we update the run-length for each bacterium corresponding to its current position, and it will be used on next moving action. The proposed strategy adaptation method records the previous and the current values of the fitness function for each bacterium, and compares these two values for conducting an assessment. If the current value (see Eq. (3)) is better than the previous one, a representative sign is marked as "+", and if the current value is worse than the previous one, the sign is marked as "−"; otherwise, the sign is "=". Derived from three consecutive fitness values, a gathered adaptability status in the strategy is a compound of two consecutive signs.

For example, the status (= +) means that the health J_{j-2}^i and J_{j-1}^i of bacteria i are the same and J_j^i is better than J_{j-1}^i. All the possible statuses are tabulated in Table 1, where *Gbest* is the best of global optimal solution of each generalization and the *Pbest* is the best of local optimal solution in each bacterium of each generalization.

As shown in Table 1, the status is divided into three different cases, and we adjust the position and calculate the next run-length (C_i) of bacterium i according to the corresponding evolutionary strategies. In case 1, statuses (= +) and (+ +) represent that bacterium has always found out a better region approaching the optimal solution, so we move this bacterium toward the global best direction

(θ_{Gbest}) of the current generation. Next, statuses (+ =) and (− +) in case 2 represent the bacterium cannot continually find out a better region, but it also does not fall into a worse area, so that we direct the bacteria to the location of the local best (θ_{pbest}^i) and of the global best

(θ_{Gbest}). Finally, in case 3, we make the bacterium swims along its direction.

We adopt these evolution strategies in the chemotaxis step of BFO algorithm, and the designed formulae of the bacterial position and run-length for different strategies are shown in Eqs. (5) - (10).

Case 1

$$\theta^i = \theta^i + C_i \times \theta_{Gbest} - \theta^i \tag{5}$$

Where0

$$C_i = \frac{J_{Gbest}}{\left(J_j^i + J_{Gbest} + J_{pbest}^i\right)} \tag{6}$$

Case 2

$$\theta^i = \theta^i + C_i \times \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{7}$$

where

$$C_i = \frac{J_j^i}{\left(J_j^i + J_{Gbest} + J_{pbest}^i\right)} \tag{8}$$

Case 3

$$\theta^i = \theta^i + C_i \times \left(\theta_{pbest}^i - \theta^i\right) + C_i \times \left(\theta_{Gbest} - \theta^i\right) \tag{9}$$

$$C_i = \frac{J_{pbest}}{\left(J_j^i + J_{Gbest} + J_{pbest}^i\right)} \tag{10}$$

Where θ^i represents the current position of bacterium i, C_i – the current bacterium's run-length, - the fitness value of bacterium i for chemotaxis step j, J_{pbest}^i and

J_{pbest}^i - the best location and the best fitness value of current bacterium, and θ_{Gbest} and J_{Gbest} - the best location and the best global optimal solution among the bacteria. In short, the adjustment of the bacterium's run-length in next movement is based on its current location.

The formulae of the run-length and position adjustment are different in each case, where the sum of three fitness values, J_j^i 、 J_{Gbest} and J_{pbest}^i, is regarded as the denominator, and each parameter is also held as the molecular. In case 1, the bacterium finds a better region, so it moves toward to θ_{Gbest} and uses the J_{Gbest} as molecular in Eq.6, which makes the run-length C_i smaller.In the following, we show the details of the proposed strategy-adaptation-based bacterial foraging optimization (SABFO) in the form of pseudo codes.

Proposed algorithm: Strategy-adaptation-based Bacterial Foraging Optimization

(SABFO)

Let N_s is the number of swim length of the bacteria in chemotaxis step, N_c is the number of chemotaxis step, N_{re} is the number of reproduction step, N_{ed} is the number of elimination-dispersal step, C_i is the run-length of each bacterium, J_j^i is the fitness value N_{ed} of each bacterium, J_{Gbest} is the best fitness value among the bacteria in each iteration, and J_{pbest}^i is the better fitness value of each bacterium.

Case	Gathered adaptability statuses	Evolutionary Strategy
1	(= +), (+ +)	move to the *Gbest* direction
2	(+ =), (− +)	move to the *Pbest* direction move
3	(− −), (= −), (+ −), (− =), (= =)	around itself

Table 1: Gathered adaptability statuses and corresponding evolutionary strategies.

[Step 1] Initialize parameters, S, CI, N_s, N_{re}, N_{ed} and p_{ed} then randomly generate the positions of bacteria in solution space, and store this position in ($\quad_{pbest}$) For each bacterium. Next, calculate the fitness values (J^i_{pbest}), and store the smallest one and its position in (J_{Gbest}) and (θ_{Gbest}), respectively. Initially, C_i is set 0.1 to make each bacterium move at the same run-length, and parameter SA_i, gathered adaptability status, is marked as "+" indicating a better searching direction.

[Step 2] Elimination-dispersal loop: $l = l + 1$;

[Step 3] Reproduction loop: $k = k + 1$;

[Step 4] Chemotaxis loop: $j = j + 1$

For bacteria i, $i = 1, 2, ..S$, take a chemotaxis step.

For swim length $m = 1, 2, ..N_s$,

Conduct strategic assessment according to the definitions of Cases 1-3, calculate the fitness value and the bacterium's new location, and update C_i and SA_i for the next run.

[Step 5] If $j < N_c$, go to [step 4]. Continue the chemotaxis step since the life of the bacteria is not over.

[Step 6] Reproduction: calculate the health of bacteria by Eq. (4). After the fitness values are sorted, the bacteria of size S_r with the higher fitness values will die, and the other bacteria with the smaller fitness value will split into two sub-bacteria of equal part. In here, S_r is the half the number of population size.

[Step 7] If $k < N_{re}$, go to [step 3]. In this case, the specified reproduction step has not yet reached, so it starts the next generation of chemotaxis loop.

[Step 8] Elimination-dispersal: If a generated random number is less than Probability p, the bacterium is randomly assigned a new position in the solution space and the parameter SA'_i is signed as "+".

If $l < p_{ed}$, then go to [step 2]; otherwise end.

There are some modifications of original BFO algorithm. In the unusual chemotaxis step, the bacteria stop swimming when the fitness value gets worse. In contract, the proposed method makes each bacterium conduct swimming operation no matter whether the fitness value worsens or not. In the elimination-dispersal step, if the bacterium is eliminated and randomized dispersed, the adaptation status SA_i will be updated as "+", exactly the same as the initialization step does. The flowchart of the proposed method is illustrated in Figure 2.

Experimental Results

Five minimizations of nonlinear benchmark function are measured to evaluate the performance of the Strategy-adaptation-based Bacterial Foraging Optimization (SABFO). These well-known benchmark functions are commonly used in the literatures evaluating the global optimization algorithm [22], and their global optimum and search range are shown in Table 2. The parameter of D represents the dimension and f is the function value. In fact, all these nonlinear benchmark functions have an optimum of zero. Each benchmark function for dimension of 30, 45 and 60 is independently executed 50 times, and their mean and standard deviation of the best-of-run solution are calculated. In order to evaluate the convergence rate of each algorithm, the number of Function Evaluations (FEs) is used, and its maximum is set differently according to the complexity of the problem. In addition, when the objective function value is less than 0.001, the stop criterion is executed. In addition, we compare the experimental results of SABFO with that of BFO and the recently proposed three algorithms, ABFOA1 [17], ABFOA2 [17] and ABFOF [23].

In these experiments, the initial run-length value λ in the ABFOA1 and ABFOA2 formula is 400, the values Ψ, a real number ranging from 0 to 1, and λ in ABFOF are 0.8 and 400, and initial run-length in SABFO and traditional BFO is 0.1. The related parameters set for these algorithms are listed in Table 3.

Tables 4-8 show the experimental results. Based on the obtained results shown in Tables 4, 5 and 7, we can clearly demonstrate that our method is superior than other compared algorithms in Sphere (f_1), Rosenbrock (f_2) and Griewank (f_4) functions. Table 9 expresses the actual execution FEs of the proposed algorithm in the five functions. Although the mean best values of our algorithm in Rastrigin (f_3)
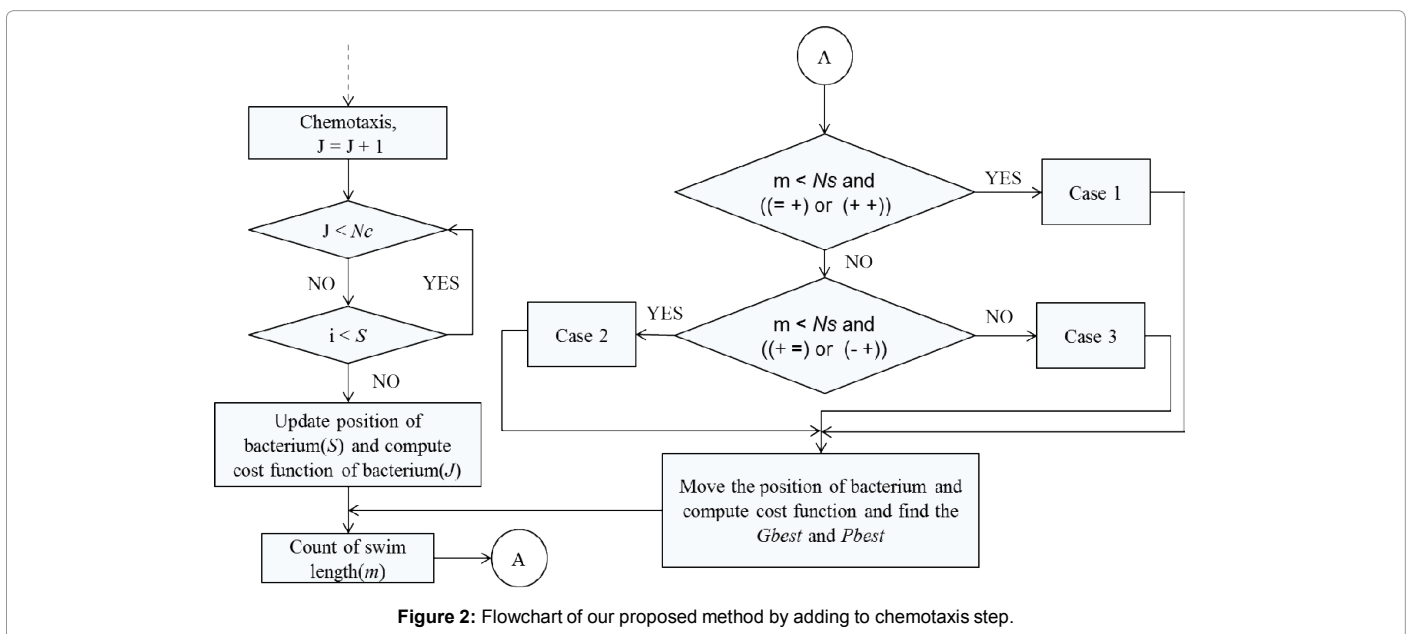

Figure 2: Flowchart of our proposed method by adding to chemotaxis step.

Functions	Mathematical Representation	Global optimum	Range of search
Sphere	$f_1(\vec{X}) = \sum_{i=1}^{D} x_i^2$	$f_1(\vec{0}) = 0$	$(-100,100)^P$
Rosenbrock	$f_2(\vec{X}) = \sum_{i=1}^{D-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	$f_2(\vec{1}) = 0$	$(-100,100)^P$
Rastrigin	$f_3(\vec{X}) = \sum_{i=1}^{D}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$f_3(\vec{0}) = 0$	$(-10,10)^P$
Griewank	$f_4(\vec{X}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$f_4(\vec{0}) = 0$	$(-600,600)^P$
Ackley	$f_5(\vec{X}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$	$f_5(\vec{0}) = 0$	$(-32,32)^P$

Table 2: Numerical benchmark functions.

and Ackley (f_5) functions are slightly larger than those of ABFOF, our proposed algorithm is much simpler to execute and takes less computation time.

Discussion

In Tables 4, 5 and 7, experimental results show that the performance of the proposed method is better than other algorithms. Moreover, our

S	N_s	N_c	N_{re}	N_{ed}	p_{re}	$d_{attractant}$	$w_{attractant}$	$h_{repellent}$	$w_{repellent}$
100	12	100	16	4	0.25	0.1	0.2	10	0.1

Table 3: Initial parameters setting of each algorithm.

Dim	Max FEs	Mean best value (Standard deviation)				
		BFO	ABFOA1	ABFOA2	ABFOF	SABFO
30	1×10^5	0.084 (0.003)	0.022 (0.0063)	0.044 (0.0721)	0 (0)	**0 (0)**
45	5×10^5	0.776 (0.156)	0.208 (0.0664)	0.419 (0.2096)	0.0014 (0.0008)	**0 (0)**
60	1×10^5	1.728 (0.213)	0.427 (0.1472)	0.632 (0.5747)	0.0038 (0.0047)	**0 (0)**

Table 4: Average of the best-of-run solution on sphere function.

Dim	Max FEs	Mean best value (Standard deviation)				
		BFO	ABFOA1	ABFOA2	ABFOF	SABFO
30	1×10^5	58.216 (14.33)	4.572 (3.0631)	6.748 (2.6625)	0.1599 (0.0773)	**0 (0)**
45	5×10^5	96.873 (26.14)	24.663 (10.864)	39.736 (30.626)	0.3116 (0.034)	**0 (0)**
60	1×10^6	154.71 (40.16)	91.257 (32.628)	84.6473 (53.273)	0.4915 (0.0451)	**0 (0)**

Table 5: Average of the best-of-run solution on Rosenbrock function.

Dim	Max FEs	Mean best value (Standard deviation)				
		BFO	ABFOA1	ABFOA2	ABFOF	SABFO
30	1×10^5	17.525 (9.896)	2.5372 (0.382)	2.9823 (0.572)	0.0084 (0.001)	0.0428 (0.0203)
45	5×10^5	32.952 (10.01)	6.0236 (1.454)	8.1121 (4.363)	0.0344 (0.0191)	0.0727 (0.0186)
60	1×10^6	41.482 (17.66)	8.3343 (0.292)	9.4637 (6.792)	0.0923 (0.041)	0.4469 (0.048)

Table 6: Average of the best-of-run solution on Rastrigin function.

algorithm can achieve 0 for objective function value in a short time in f_1, f_2, f_4 functions. According to the mean and standard deviation of the best-of-run solution in the Tables 4-8, the proposed method obtains a

Dim	Max FEs	Mean best value (Standard deviation)				
		BFO	ABFOA1	ABFOA2	ABFOF	SABFO
30	1×10^5	0.3729 (0.035)	0.1914 (0.012)	0.2028 (0.153)	0.0142 (0.017)	**0 (0)**
45	5×10^5	0.6351 (0.052)	0.3069 (0.053)	0.3065 (0.092)	0.0096 (0.010)	**0 (0)**
60	1×10^6	0.8324 (0.075)	0.5638 (0.345)	0.6074 (0.573)	0.0055 (0.005)	**0 (0)**

Table 7: Average of the best-of-run solution on Griewank function.

Dim	Max FEs	Mean best value (Standard deviation)				
		BFO	ABFOA1	ABFOA2	ABFOF	SABFO
30	1×10^5	2.3243 (1.883)	0.5038 (0.551)	0.7316 (0.675)	0.0063 (0.007)	0.0141 (0.010)
45	5×10^5	3.4564 (3.439)	1.5532 (0.195)	1.3672 (0.462)	0.0067 (0.005)	0.0187 (0.018)
60	1×10^6	4.3247 (1.561)	1.7832 (0.458)	1.9272 (0.7734)	0.0069 (0.0024)	0.0196 (0.018)

Table 8: Average of the best-of-run solution on Ackley function.

Functions	Dim	Max FEs	Actual Execution FEs of our Method
Sphere	30	1×10^5	5
	45	5×10^5	5
	60	1×10^6	5
Rosenbrock	30	1×10^5	2028
	45	5×10^5	780
	60	1×10^6	805
Rastrigin	30	1×10^5	1×10^5
	45	5×10^5	5×10^5
	60	1×10^6	1×10^6
Griewank	30	1×10^5	484
	45	5×10^5	409
	60	1×10^6	19
Ackley	30	1×10^5	1×10^5
	45	5×10^5	5×10^5
	50	1×10^6	1×10^6

Table 9: Actual execution FEs of the proposed method in five functions.

smaller the variation of standard deviation. Therefore, we can find that the proposed method is better convergent stability than other methods. In Table 9, the proposed method only requires less number of the actual execution FEs to obtain the optimal solution of the objective function.

Conclusion

In this study, an efficient Strategy-adaptation-based Bacterial Foraging Optimization (SABFO) algorithm is proposed for numerical optimization. The advantages of the proposed SABFO are to use the strategy adaptation method in the chemotaxis step for improving the exploratory capability of each bacterium in the search space. The status of the proposed strategy adaptation method in the chemotaxis step is divided into three different cases. The position and the next run-length of a bacterium are adjusted and calculated according to the corresponding evolutionary strategies. Experimental results demonstrate that the proposed SABFO has superior performance than other methods in most cases.

References

1. Vasant PM (2012), Meta-Heuristics Optimization Algorithms in Engineering, Business, Economics, and Finance, IGI Global, September.

2. Vasant PM (2013) Handbook of Research on Novel Soft Computing Intelligent Algorithms: Theory and Practical Applications (2 Volumes), IGI Global, August.

3. Vasant PM (2014) Handbook of Research on Artificial Intelligence Techniques and Algorithms (2 Volumes), IGI Global, November.

4. Holland JH (1975) Adaptation in Natural and Artificial Systems, University of Michigan Press.

5. Goldberg DE (1989), Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley.

6. Storn R, Price K (1995) "Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces," Technical Report TR-95-012, Berkeley, USA.

7. Chatzis SP, Koukas S (2011) "Numerical optimization using synergetic swarms of foraging bacterial populations," Expert Systems with Applications 38: 15332-15343.

8. Korani WM, Dorrah HT, Emara HM (2009) "Bacterial foraging oriented by particle swarm optimization strategy for PID tuning," IEEE Computational Intelligence in Robotics and Automation (CIRA) 445-450.

9. Yan X, Zhu Y, Chen H, Zhang H (2012) "Improved bacterial foraging optimization with social cooperation and adaptive step size," Intelligent Computing Technology, Lecture Notes in Computer Science 7389: 634-640.

10. Eberhart R, Kennedy J (1995) "A new optimizer using particles swarm theory," Proc. of IEEE Int. Conf. on Neural Networks 1942-1948.

11. Dorigo M, Caro GD (1999) "Ant colony optimization: A new meta-heuristic," Proceedings of the 1999 Congress on Evolutionary Computation, CEC 99 2: 1470-1477.

12. Passino KM (2002) "Biomimicry of bacterial foraging for distributed optimization and control," IEEE Control Systems Magazine 22: 52-67.

13. Biswas Dasgupta S, Das S, Abraham A (2007) "A synergy of differential evolution and bacterial foraging algorithm for global optimization," Neural Network World 607-626.

14. Majhi R, Panda G, Majhi B, Sahoo G (2009) "Efficient prediction of stock market indices using adaptive bacterial foraging optimization (ABFO) and BFO based techniques," Expert Systems with Applications 36: 10097-10104.

15. Chen H, Zhu Y, Hu K (2008) "Self-adaptation in bacterial foraging optimization algorithm," 3rd International Conference on Intelligent System and Knowledge Engineering 1: 1026-1031.

16. Chen H, Zhu Y, Hu K (2011) "Adaptive bacterial foraging optimization," Hindawi Publishing Corporation Abstract and Applied Analysis.

17. Dasgupta S, Das S, Abraham A, Biswas A (2009) "Adaptive computational chemotaxis in bacterial foraging optimization: An analysis," IEEE Transactions on Evolutionary Computation 13: 919-941.

18. Biswas Das S, Abraham A, Dasgupta S (2010) "Analysis of the reproduction operator in an artificial bacterial foraging system," Applied Mathematics and Computation 215: 3343-3355.

19. Serra P, Stanton AF, Kais S (1997) "Pivot method for global optimization," Physical Review E 55:1162-1165.

20. Cooren Y, Clerc M, Siarry P (2009) "Performance evaluation of TRIBES, An adaptive particle swarm optimization algorithm," Swarm Intelligence 3: 149-178.

21. Halgamuge SK, Alfonso W, Caicedo EF (2010) "Simplifying the bacteria foraging optimization algorithm," IEEE Congress on Evolutionary Computation (CEC) 1-7.

22. Yao X, Liu Y, Lin G (1999) "Evolutionary programming made faster," IEEE Transactions Evolutionary Computation 3: 82-102.

23. Xu X, Liu YH, Wang AM, Wang G, Chen HL (2012) "A new adaptive bacterial foraging optimizer based on field," 8th International Conference on Natural Computation 986-990.