WCCI 2010 IEEE World Congress on Computational Intelligence
July, 18-23, 2010 - CCIB, Barcelona, Spain CEC IEEE

Simplifying the Bacteria Foraging Optimization Algorithm

Mario A. Muñoz, Saman K. Halgamuge, Wilfredo Alfonso, and Eduardo F. Caicedo

***Abstract*— The Bacterial Foraging Optimization Algorithm is a swarm intelligence technique which models the individual and group foraging policies of the E. Coli bacteria as a distributed optimization process. The algorithm is structurally complex due to its nested loop architecture and includes several parameters whose selection deeply influences the result. This paper presents some modifications to the original algorithm that simplifies the algorithm structure, and the inclusion of best member information into the search strategy, which improves the performance. The results on several benchmarks show reasonable performance in most tests and a considerable improvement in some complex functions. Also, with the use of the T–Test we were able to confirm that the performance enhancement is statistically significant.**

I. Introduction

Nature has been a source of inspiration for the design of several algorithms. One main principle behind nature-inspired algorithms is the concept of efficiency, interpreted as the capability of an individual to obtain a sufficient energy source in the least amount of time [1]. This procedure called foraging is crucial in natural selection, since the animals with poor foraging strategies are eliminated, and successful ones tend to propagate. Hence, to survive, an animal or a group of animals must develop an optimal foraging policy [2]. Some of the most successful foragers are bacteria like the E. Coli, which employs chemical sensing organs to detect the concentration of nutritive or noxious substances in its environment. The bacteria then moves through the environment by a series of tumbles and runs, avoiding the noxious substances and getting closer to food patch areas in a process called chemotaxis. Besides, the bacteria can secrete a chemical agent that attracts its peers, resulting in an indirect form of communication [3].

Inspired by the E. Coli foraging strategy, in 2000, K. Passino proposed the Bacteria Foraging Optimization Algorithm (BFOA), as a numerical optimization algorithm [4]. This algorithm, which can be classified as a Swarm Intelligence (SI) technique, is not the only one to use the chemotactical behavior as basis for an optimization procedure. Other work includes Bremermann in 1974 [5], Müller, *et al.* in 2002 [6], Vergassola, *et al.* in 2007 [7] and Nicolau, *et al.* in 2008 [8]. Nevertheless, BFOA has attracted a lot of attention from researchers, and several applications have been reported [9]–[19], other works that analyze each of its components [20]–[29], and some hybrid algorithms have been proposed [30]–[34]. Most of these works agree in that the algorithm uses several parameters, which deeply affect the result of the search. Then, the authors present a solution to this problem by simplify the algorithm without sacrificing the performance.

This paper presents some modifications for the BFOA to improve its computation speed and convergence, while simplifying the algorithm structure. The paper is organized as follow. In Section II we describe the original BFOA, and in Section III we report some works attempting to solve the problems associated with BFOA. In Section IV, we present the proposed modification to the algorithm. To examine its performance, we carry out a simulation study using some common benchmark functions comparing the proposed algorithm with the original and adaptive bacteria algorithms. The results for these tests are shown in Section V. Finally, in Section VI we present some conclusions.

Mario A. Muñoz and Saman K. Halgamuge are with the Department of Mechanical Engineering, University of Melbourne, Melbourne, Victoria, Australia (email: m.munozacosta@pgrad.unimelb.edu.au - saman@unimelb.edu.au).

Wilfredo Alfonso and Eduardo F. Caicedo are with the School of Electrical and Electronics Engineering, Universidad del Valle, Cali, Valle, Colombia (email: walfomor@univalle.edu.co - ecaicedo@univalle.edu.co)

II. The Bacteria Foraging Optimization Algorithm

Suppose that we need to find the minimum of a function $J(\theta), \theta \in R^p$, when we do not have a deterministic description of $J(\theta)$ or its gradient. This problem becomes a non gradient optimization problem, where the ideas from bacteria foraging can be used. Suppose that θ is the position of the bacteria and $J(\theta)$ represents the environment conditions, with $J(\theta) < 0$, $J(\theta) = 0$, and $J(\theta) > 0$ represents that the bacteria location is a nutrient rich, neutral, or noxious environment, respectively. The chemotaxis is a foraging behavior where bacteria attempts to increase the nutrient concentration, avoid noxious substances and search for ways out of neutral media by random walk.

Then, we can define a chemotactic step j as a tumble followed by a tumble or a run, a reproductive step k as the selection of the fittest in the population and its splitting, and an elimination-dispersal event l as the selection of random individuals and its relocation on new random positions. Then, $P(j,k,l) = \theta_i(j,k,l) \| i = 1,2,\ldots S$ are the positions of each member of the S bacteria population at j-th chemotactic step, k-th reproductive step and l-th elimination and dispersion event. Then $J(i,j,k,l)$ is the location cost of the i-th bacteria $\theta_i(j,k,l) \in R^p$, and N_c is the bacteria's life time in chemotactic steps. The bacteria move following (1) where $C(i)$ is the size of the step at the direction $\Delta(i)$. If in $\theta_i(j+1,k,l)$, the value of $J(i,j+1,k,l)$ is less than in $\theta_i(j,k,l)$, then a new step is taken in the same direction until a maximum of N_s, making this cycle a chemotactic step.

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

Algorithm 1: Bacteria Foraging Optimization Algorithm

```
Initialize parameters;
Initialize the location of the population;
for $l = 1$ to $N_{ed}$ do
    for $k = 1$ to $N_{re}$ do
        for $j = 1$ to $N_c$ do
            for $i = 1$ to $S$ do
                Calculate
                $J = J(i,j,k,l) + J_{cc}(\theta, P(j,k,l))$;
                Generate a random direction $\Delta(i)$;
                $J_{last} = J$;
                Move using (1);
                $m = 0$;
                while $m < Ns$ do
                    $m = m + 1$;
                    Calculate a new
                    $J = J(i,j,k,l) + J_{cc}(\theta, P(j,k,l))$;
                    if $Jlast > J$ then
                        Move using (1);
                    else
                        $m = N_s$
        Split the best bacteria, eliminate the worst;
    Disperse at random some bacteria;
```

During its movement, the bacteria communicate among one another using chemical substances known as attractants and repellents, which deform the search space, making those locations where more individuals are located more attractive, but at the same time, avoids that bacteria get on top of one another. To calculate this effect (2) is used, where m_{ar} is the magnitude of the attractant/repellent, w_{at} and w_{re} are the width of the attractant and repellent respectively.

$$J_{cc}(\theta, P) = \sum_{i=1}^{S} m_{ar} \left[e^{-w_{re} \sum_{m=1}^{P} \left(\theta_m - \theta_m^i\right)^2} - e^{-w_{at} \sum_{m=1}^{P} \left(\theta_m - \theta_m^i\right)^2} \right] \tag{2}$$

After N_c chemotactic steps, a reproduction step is taken. For the reproduction, the healthiest bacteria are split and the others are eliminated, maintaining a constant population. The individuals to be reproduced are selected by using a health metric which is the cumulative sum of the cost value on each position visited by the bacterium. After N_{re} reproduction steps, a dispersion and elimination event is made, where each bacterium is subject to relocation with a probability p_{ed}. After N_{ed} dispersion and elimination, the algorithm ends. The population size S is restricted to an even number, so the population can be easily kept constant. The structure is shown the Algorithm 1.

III. Related works on BFOA parameters

Although some works have been carried out to identify the effect of each parameter of BFOA, still there is not a profound knowledge of the interaction of all them. At the moment, it is acknowledged that the most critical parameter is the step size C, because of its strong influence in the algorithm stability and convergence. Since the inception of BFOA, the step size remained fixed number, but its adaptation was implied. Dasgupta, et al. [29] show that for the algorithm to converge it is necessary to modify its value on the run. For that, there are several choices possible. Mishra [9] suggest using a Fuzzy Logic Controller (FLC) to adapt this parameter. Nevertheless this requires the tunning of a complete FLC, which implies the selection of several more parameters. Other choices include using the individual performance to adapt its step size. This method, presented in [28], [29], proposes an analytical solution for the step adaptation. If the cost function has its minimum value equal to 0, the authors suggest that (3) is used, but if the minimum is not equal to 0, then (4) is used. These two equations were developed assuming that the step size is small and the location of the bacterium is close to the optimal. It also use a new parameter λ, with its proposed value of $\lambda = 1/16$ which is only valid if the previous assumptions hold. Then it is necessary to select a value for this new parameter each time the algorithm is applied, for which there is not any selection rules just systematic trial–and–error tunning.

$$C(i) = \frac{1}{1 + \frac{\lambda}{|J(\theta)|}} \tag{3}$$

$$C(i) = \frac{1}{1 + \frac{\lambda}{|J(\theta) - J_{best}|}} \tag{4}$$

Other solutions to the problem of the step size was proposed by Korani in 2008 [34], where the random direction was replaced by a particle swarm optimization (PSO) based movement rule using (5) and (6), where w is the inertia weight, c_1 and c_2 are the acceleration coefficients, R_1 and R_2 are two random numbers from an uniform distribution, P_{best} is the best known position of the particle and G_{best} is the best known global position. While the idea to add a simpler mean of information exchange that is thoroughly proven, the proposal adds the following inconvenience. It ignores the use of ϕ as a unit direction vector, and its multiplication by C could create a possible explosion of the bacteria in the same way it occurs in PSO. The result is a PSO algorithm with cell–to–cell communication based in attractants/repellents.

$$\phi(j+1) = w\phi(j) + C_1 R_1 (P_{best} - \theta) + C_2 R_2 (G_{best} - \theta) \tag{5}$$

$$\theta(j+1,k,l) = \theta(j,k,l) + C(i)\phi(j+1) \tag{6}$$

The reproduction and the elimination/dispersal mechanisms also produce noticeable effects. The reproduction scheme is a major exploitation mechanism, because it eliminates the “worst performing” individuals and replaces them with copies of the “best performing.” Because of this, diversity in the population is lost if the reproduction steps are very frequent. Besides, this mechanism heavily depends on the health metric used, which is usually the sum of the cost of each position the bacteria have found. This can have some deceiving results, since a bacterium stagnated in a local minimum could have a better health measurement than a bacterium located nearby the global optimum. The elimination/dispersal event is a major exploration scheme, since it takes any bacteria and relocates it in a new position. This event can disrupt in a deep way the search process, since we could take our best performing bacteria and locate it in an inadequate environment, losing the possibility to get a better result from that individual.

Other issue is the calculation of the attractants/repellents. The original formulation of this mechanism is time consuming, since it requires the measurement of the distance between each individual. While the mechanism adds communication between the individuals, it also adds three more parameters and a complex calculation to be obtained. In some works, this mechanism is removed altogether, leaving the system without any cooperative information source. Also, it seems appropriate to make stronger signals when the bacterium is located in a best position. To improve this, Liu and Passino [2] suggested that (7) is used, to relate the cost function to the cell–to–cell communication. The overall cost function $J_{ar}(\theta)$ of the location is now a composition of the attractant/repellent function J_{cc} and the environment cost function $J(\theta)$. Nevertheless it adds a new parameter M, for which there is no clear rule for tuning.

$$J_{ar}(\theta) = e^{M-J(\theta)} J_{cc}(\theta, P(j,k,l)) \tag{7}$$

One final problem that can be identified is that the general structure of the algorithm is quite complex. Since the evaluation relies on nested loop architecture, it is a little difficult to know *a priori* how many function evaluations will be carried out by the algorithm. If we assume $N_s = 4$, $N_c = 100$, $N_{re} = 4$, and $N_{ed} = 2$, we could have up to 4000 function evaluations per individual, but this number will depends on how many times does the bacteria move on a chemotactic step. We can conclude that, while the solutions proposed by these authors are useful and give insight about the working of the algorithm, they add new problems to be solved when implemented. Therefore, an integral solution is needed. In their 2009 work, Mezura-Montes and Hernández-Ocaña [35] propose the elimination of the nested loop architecture, a step size selection based on (8), where R is a scaling factor, U_i and L_i are the upper and lower limits of the search space, and n is the number of decision variables; a reproduction and elimination/dispersal events for a single individual each iteration, and a swarming mechanism based on the best individual information. Nevertheless, still does not consider a variable step size, which eventually would render the algorithm unstable. Therefore, we propose a solution which tackles some of the issues mentioned, while simplify the algorithm structure and the parameter use.

$$C(i, j+1) = R\frac{U_i - L_i}{\sqrt{n}} \tag{8}$$

IV. Proposed changes to Bacteria Foraging Algorithm

We aim to simplify the algorithm while maintaining its core elements. This include the simplification of the algorithm architecture, the elimination of the N_s parameter, a clear adaptation rule for the step size $C(i)$, the use of an uniform distribution the position initialization, the inclusion of the best individual information in the movement equation, and the removal of the cell–to–cell communication.

The first issue to resolve is the algorithm structure. Since the reproduction and the elimination/dispersal events occur after the chemotactic steps are exhausted, we can replace the loops for iteration counters which trigger the event when a number of iterations have been made. In the original case, when $N_s = 4$, $N_c = 100$, $N_{re} = 4$, and $N_{ed} = 2$, we could replace the counter for N_{re} for an event every 500 iterations, and N_{ed} for an event every 1000 iterations. Since we are eliminating the loops, the bacteria will either tumble or run once on each iteration. We also remove the step counter N_s and the bacteria can swim in one direction while it is a good direction.

The second issue is to develop an adaptation scheme for $C(i)$ which could improve the search and convergence. First, we must remember that the step size is multiplied by an unit direction vector. We thoroughly tested several step sizes using the test reported in Section V and we conclude that $C(i,0) = 0.01d$, where d is the diagonal of the search space, is an appropriated value. As adaptation technique, we use a modification the $1/5$-th rule extracted from the Evolution Strategies (ES) [36]. We assume that $C(i, j+1) = C(i, j) + \sigma_i(j) \cdot R$, where σ_i is the mutation strength for the bacteria i, and R is a random number from a Gaussian distribution. We carried out the test reported in Section V with different values for this parameter and we concluded that $\sigma_i(0) = 0.0001d$ gives good results. For the adaptation of the step size we include a rule to control the size of $\sigma_i(j)$ depending on the cost value found by the bacteria. If the cost value has decreased or has sustained, then $\sigma_i(j) = 0.80\sigma_i(j-1)$, otherwise $\sigma_i(j) = 1.20\sigma_i(j-1)$. The chosen values for the adaptation are based on the works of ES in the change of mutation strength [36], since we can compare the movement of the bacteria to the mutation in ES. The results show that they work in an acceptable way.

The next modification proposed is the use of a different metric for the health measurement. For us it is more interesting the bacteria that were able to descend more in the gradient, than those that got stagnated. For this, we replace the health measurement from the sum of the cost functions to the sum of the change between two steps, ΔJ. If a bacterium has not moved, then $\Delta J = 0$, and if it has descended the gradient then $\Delta J < 0$.

From Korani's work [34], we suggest the use of the best known individual to guide the search. First, we assume that

Algorithm 2: Revised Bacteria Foraging Algorithm

```
Initialize parameters;
Initialize the location of the population by Hammersley method;
Calculate $J_{last} = J(\theta)$;
Obtain the $\theta^{gbest}$;
for j=1 to epochs do
    for i=1 to S do
        Move using (9);
        Calculate $J\left(\theta^{i}\right)$;
        if $J_{last} \geq J(\theta)$ then
            Update position, cost and health;
            $\sigma_i(j) = 0.80 \cdot \sigma_i(j-1)$;
        else
            Keep previous position, cost and health;
            Generate a random direction $\Delta(i)$;
            $\sigma_i(j) = 1.20 \cdot \sigma_i(j-1)$;
        $C(i, j+1) = C(i, j) + \sigma_i(j) \cdot R$;
    Update the $\theta^{gbest}$;
    if Reproduction Step then
        Split the best bacteria, eliminate the worst;
    if Elimination/Dispersal Event then
        Disperse at random some bacteria;
```

the bacteria will not move from its location if the next step will take it to a worst position, then the bacteria are *always* at their personal best. Then, to add the best known individual information to the movement of the bacteria, we suggest replacing (1) by (9), where K is calculated by (10) and θ^{gbest} is the best position found so far by the swarm. Because the bacteria are always at their personal best, then there is going to be a bacterium that is located in the global best, then in this case, when we calculate the unit vector for K in (9), this term becomes ∞. For this bacterium we replace this value with 0.

$$\theta^{i}(j) = \theta^{i}(j) + C(i, j)\left(\frac{\Delta(i)}{\sqrt{\Delta^{T}(i) \cdot \Delta(i)}} + \frac{K(i)}{\sqrt{K^{T}(i) \cdot K(i)}}\right) \tag{9}$$

$$K(i) = \theta^{gbest}(j) - \theta^{i}(j) \tag{10}$$

Finally, following some suggestions found for PSO [37], we encountered that position initialization plays a major role in the algorithm results. The usual method for position initialization is the use of an uniform random distribution, which creates noise filled initial positions. Therefore, regular pseudo random distributions are suggested for the initialization of PSO, which among the most popular are Halton, Hammersley, and Centroidal Voronoi Tessellations (CVT). We carried out the test reported in Section V with each of these distribution algorithms and the best results were obtained with the Hammersley distribution.

TABLE I
PARAMETERS USED FOR THE BENCHMARK

Algorithm	Parameters
NBFO	$S = 50,\ N_{re} = 100,\ N_{ed} = 500,\ p_{ed} = 0.25$ $C = 0.01d,\ \sigma = 0.0001d$
BFOA	$S = 50,\ N_c = 125,\ N_s = 4,\ N_{re} = 2,\ N_{ed} = 2$ $p_{ed} = 0.25,\ C = 0.1,\ m_{ar} = 0.1,\ w_{at} = 0.2\ w_{re} = 10$
ABFOA	$S = 50,\ N_c = 125,\ N_s = 4,\ N_{re} = 2,\ N_{ed} = 2$ $p_{ed} = 0.25,\ C_r = 0.01d \cdot C(i),\ \lambda = \max(J(1,1,1))$ $m_{ar} = 0.1,\ w_{at} = 0.2\ w_{re} = 10$

While it is necessary to obtain a better cell–to–cell communication scheme, for the moment we discard the original one to reduce the number of parameters to be tuned. In overall, the resulting structure is presented in the Algorithm 2.

V. BENCHMARK

To test the modifications to the algorithm, we carried on a benchmark study using 18 well known test functions. These are the Sphere (F1 / $[-100,\ 100]$), Rosenbrock (F2 / $[-100,\ 100]$), Ackley (F3 / $[-30,\ 30]$), Griewank (F4 / $[-600,\ 600]$), Rastrigin (F5 / $[-5.12,\ 5.12]$), Non–continuous Rastrigin (F6 / $[-5.12,\ 5.12]$), Schewefel (F7 / $[-500,\ 500]$), the rotated versions of functions three to six (F8–F11), Schaffer (F12 / $[-100,\ 100]$), and the six Composition functions from [38] (F13–F18 / $[-5,\ 5]$), all of them in 30 dimensions. We compare our results with the Original BFOA and the Adaptive BFOA [29], using (4). The parameters used for each algorithm are shown in Table I, where NBFO is the proposed algorithm and for it N_{re} and N_{ed} represent the number of epochs necessary to perform a step. Each algorithm was run 30 times, each with a total of 50000 function evaluations. We calculate the minimum, the mean, and the standard deviation for the data, which are shown in Table II.

The results show that there is an improvement over complex functions over the original and adaptive algorithms (F7, F13, F14 and F17), and good results within an acceptable range of tolerance in other less complex functions (F1, F3, F4, F8, F9, F12, F16 and F18). In the remaining functions the results are disappointing. The proposed solution still presents some stagnation problems, since it does not achieve the global minimum with the given number of function evaluations limit and catastrophically fails in some functions.

For an one–on–one performance comparison, we used the t–test with a 95% confidence range. The t–test allows the verification of the statistical validity of the result and at the same time if the algorithm under test can be considered statistically better than the control algorithm. The t–test, considered to be a signal to noise ratio, calculates a difference between the two groups. If we have $n = 30$ for both groups, the lower limit for a 95% confidence range is 1.66055. According to t value it is possible to conclude about the performance of the test algorithm, if $t \leq -1.697261$ the performance is better than the control algorithm, if $-1.697261 < t <$

TABLE II

Minimum and Mean Values, and Standard Deviation for the benchmark function test for functions F1 to F18

		Min	Mean	Std
F1	NBFO	**0.00**	10.06	6.29
	BFOA	0.00	0.00	0.00
	ABFOA	0.00	**0.00**	**0.00**
F2	NBFO	**26.19**	13308.75	24120.75
	BFOA	29.00	29.00	**0.00**
	ABFOA	28.96	**28.98**	0.01
F3	NBFO	0.02	2.13	0.68
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F4	NBFO	0.02	0.85	0.45
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F5	NBFO	79.69	224.86	42.44
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F6	NBFO	169.41	222.51	22.43
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F7	NBFO	**-7452.32**	**-5349.66**	886.35
	BFOA	414.42	415.20	0.45
	ABFOA	-5059.80	-3258.11	**878.63**
F8	NBFO	0.04	2.25	0.65
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F9	NBFO	0.01	0.79	0.51
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F10	NBFO	140.33	223.84	31.84
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F11	NBFO	174.86	240.82	21.82
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F12	NBFO	0.49	0.50	0.00
	BFOA	0.00	0.00	0.00
	ABFOA	**0.00**	**0.00**	**0.00**
F13	NBFO	**0.02**	**61.10**	**93.08**
	BFOA	883.42	888.59	1.85
	ABFOA	862.35	894.47	6.39
F14	NBFO	**114.13**	**399.67**	**60.03**
	BFOA	900.00	900.00	0.00
	ABFOA	739.51	892.69	30.86
F15	NBFO	**300.34**	**675.45**	**199.10**
	BFOA	900.00	900.00	0.00
	ABFOA	900.00	900.00	0.00
F16	NBFO	**900.00**	923.31	41.34
	BFOA	900.00	900.00	0.00
	ABFOA	900.00	**900.00**	**0.00**
F17	NBFO	**21.51**	**202.49**	193.30
	BFOA	900.00	900.00	0.00
	ABFOA	900.00	900.00	**0.00**
F18	NBFO	900.01	900.37	0.20
	BFOA	900.00	900.00	0.00
	ABFOA	**900.00**	**900.00**	**0.00**

TABLE III

Results of the T–Test: Values of t for a 95% confidence range

	BFOA	ABFOA		BFOA	ABFOA
F1	8.77	8.77	F10	60.93	60.93
F2	3.02	3.02	F11	63.63	63.63
F3	17.15	17.15	F12	5283.92	5283.92
F4	10.32	10.32	F13	**-94.51**	**-94.46**
F5	29.02	29.02	F14	**-48.53**	**-41.69**
F6	62.85	62.85	F15	**-6.18**	**-6.18**
F7	**-115.25**	**-11.07**	F16	2.96	2.96
F8	18.96	18.96	F17	**-29.63**	**-29.63**
F9	8.51	8.51	F18	9.46	9.46

1.697261 the performance is equal, and the performance is lower if $1.697261 \leq t$. We boldfaced the better and equally performing situations for the new algorithm.

VI. Conclusions

This paper has presented a modification for the Bacteria Foraging Algorithm proposed by Passino [3], [4]. We aimed to simplify the algorithm while maintaining its core elements. The most important modifications are the elimination of the N_s parameter, the simplification of the algorithm architecture by removing the nested loops, a clear adaptation rule for the step size $C(i)$, the use of an uniform distribution for the position initialization, the inclusion of the best individual information in the movement equation, and the removal of the cell–to–cell communication. The results obtained from the benchmark tests suggest that a reasonable performance is obtained under the same test conditions in most tests, with a reduced complexity compared with the original and adaptive bacteria algorithm, and in different type of benchmark functions. Still, the algorithm suffers of premature convergence and in several tests did not acquire the global minimum in the function evaluation limit previously set. This still puts the bacteria algorithm in disadvantage against some other proven metaheuristics, which achieve better reported results than those reported in this paper in several of the benchmark functions tested [39]. A possible reason for this behavior is the lack of "elasticity" of the step size.

Nevertheless, more work must be carried on. The first issue is to develop a simple cell–to–cell communication scheme. Even if there is work that supports its stability and performance [23], it is a time consuming effort, which adds several parameters, and for simplicity is commonly removed from the general practice, as it was in this case. The next issue to solve is how to simplify the reproduction and elimination/dispersal mechanisms even further, so they do not depend on any parameters. In our case the number of iterations is required for one of these events to happen. Finally, it would be convenient to make stability and convergence tests for the proposed solution, which would show, if the algorithm performs comparable to those reported in [39].

Acknowledgments

This work was possible thanks to the funding of the University of Melbourne and the Ministry of Information and Communication Technologies of Colombia, through a graduate research scholarship awarded to M.A. Muñoz.

REFERENCES

[1] K. Steer, A. Wirth, and S. Halgamuge, "The rationale behind seeking inspiration from nature," in *Nature-Inspired Algorithms for Optimisation*, ser. Studies in Computational Intelligence, R. Chiong, Ed. Springer, 2009, vol. 193, pp. 51–76.

[2] Y. Liu and K. Passino, "Biomimicry of social foraging bacteria for distributed optimization: Models, principles and emergent behaviors," *Journal of Optimization Theory and Applications*, vol. 115, no. 3, pp. 603–628, 2002.

[3] K. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.

[4] ——, "Distributed optimization and control using only a germ of intelligence," in *Intelligent Control, 2000. Proceedings of the 2000 IEEE International Symposium on*, 2000, pp. P5–13.

[5] H. Bremermann, "Chemotaxis and optimization," *Journal of the Franklin Institute*, vol. 297, no. 5, pp. 397–404, 1974.

[6] S. Müller, J. Marchetto, S. Airaghi, and P. Koumoutsakos, "Optimization based on bacterial chemotaxis," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 16–29, 2002.

[7] M. Vergassola, E. Villermaux, and B. Shraiman, "'infotaxis' as a strategy for searching without gradients," *Nature*, vol. 445, no. 7126, pp. 406–409, 2007, cited By (since 1996) 31. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-33846553228&partnerID=40

[8] D. Nicolau, Jr., K. Burrage, D. Nicolau, P. Maini, D. V. N. Jr., K. Burrage, D. V. Nicolau, and P. K. Maini, "'Extremotaxis': Computing with a bacterial-inspired algorithm," *BioSystems*, vol. 94, pp. 47–54, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/B6T2K-4ST3YNH-C/2/8378efc668ecd663d73e82b2ed5e6fea

[9] S. Mishra, "A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation," *Evolutionary Computation, IEEE Transactions on*, vol. 9, no. 1, pp. 61–73, Feb. 2005.

[10] D. Kim and J. Cho, "Adaptive tuning of PID controller for multivariable system using bacterial foraging based optimization," in *Advances in Web Intelligence*, ser. Lecture Notes in Computer Science. Springer, 2005, vol. 3528, pp. 231–235.

[11] M. Tripathy, S. Mishra, L. Lai, and Q. Zhang, "Transmission loss reduction based on facts and bacteria foraging algorithm," in *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference, Reykjavik, Iceland, September 9-13, 2006, Procedings*, ser. Lecture Notes in Computer Science, T. P. Runarsson, H.-G. Beyer, E. K. Burke, J. J. M. Guervós, L. D. Whitley, and X. Yao, Eds., vol. 4193. Springer, 2006, pp. 222–231.

[12] D. Acharya, G. Panda, S. Mishra, and Y. Lakshmi, "Bacteria foraging based independent component analysis," in *ICCIMA '07: Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 527–531.

[13] W. Alfonso, "Regulación de temperatura en la plataforma UV–PTM01 basada en agentes cooperativos para la asignación dinámica de recursos," Universidad del Valle, Tech. Rep., 2007.

[14] M.S. Li, W.J. Tang, W.H. Tang, and Q. J.R. Saunders, "Bacterial foraging algorithm with varying population for optimal power flow," in *EVOWORKSHOPS*, ser. Lecture Notes in Computer Science, vol. 4448. Springer Berlin / Heidelberg, 2007, pp. 267–261.

[15] M. Muñoz, J. López, and E. Caicedo, "Bacteria swarm foraging optimization for dynamical resource allocation in a multizone temperature experimentation platform," in *12th International Fuzzy Systems Association Congress (IFSA 2007)*, 2007.

[16] S. Mishra and C. Bhende, "Bacterial foraging technique-based optimized active power filter for load compensation," *Power Delivery, IEEE Transactions on*, vol. 22, no. 1, pp. 457–465, Jan. 2007.

[17] M. Tripathy and S. Mishra, "Bacteria foraging-based solution to optimize both real power loss and voltage stability limit," *Power Systems, IEEE Transactions on*, vol. 22, no. 1, pp. 240–248, Feb. 2007.

[18] C. Wu, N. Zhang, J. Jiang, J. Yang, and Y. Liang, "Improved bacterial foraging algorithms and their applications to job shop scheduling problems," in *8th International Conference ICANNGA 2007*, ser. Lecture Notes in Computer Science, vol. 4431. Springer, 2007, pp. 562–569.

[19] T. Datta, I. Misra, B. Mangaraj, and S. Imtiaj, "Improved adaptive bacteria foraging algorithm in optimization of antenna array for faster convergence," *Progress In Electromagnetics Research C*, vol. 1, pp. 14–157, 2008.

[20] V. Gazi and K. Passino, "Stability analysis of swarms in an environment with an attractant/repellent profile," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 3, 2002, pp. 1819–1824 vol.3.

[21] ——, "Stability analysis of swarms," *IEEE Transactions on Automatic Control*, vol. 48, no. 4, pp. 692–697, 2003.

[22] ——, "Stability analysis of social foraging swarms," *IEEE Transactions of Systems, Man and Cybernetics - Part B*, vol. 34, no. 1, pp. 539–557, 2004.

[23] ——, "A class of attractions/repulsion functions for stable swarm aggregations," *Int. J. Control*, vol. 77, no. 18, pp. 1567–1579, 2004.

[24] ——, "Stability of a one-dimensional discrete-time asynchronous swarm," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 35, no. 4, pp. 834–841, Aug. 2005.

[25] A. Abraham, A. Biswas, S. Dasgupta, and S. Das, "Analysis of reproduction operator in bacterial foraging optimization algorithm," in *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, June 2008, pp. 1476–1483.

[26] A. Biswas, S. Das, S. Dasgupta, and A. Abraham, "Stability analysis of the reproduction operator in bacterial foraging optimization," in *CSTST '08: Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology*. New York, NY, USA: ACM, 2008, pp. 564–571.

[27] S. Das, S. Dasgupta, A. Biswas, A. Abraham, and A. Konar, "On stability of the chemotactic dynamics in bacterial-foraging optimization algorithm," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 3, pp. 670–679, May 2009.

[28] S. Dasgupta, A. Biswas, A. Abraham, and S. Das, "Adaptive computational chemotaxis in bacterial foraging algorithm," in *Complex, Intelligent and Software Intensive Systems, 2008. CISIS 2008. International Conference on*, March 2008, pp. 64–71.

[29] S. Dasgupta, S. Das, A. Abraham, and A. Biswas, "Adaptive computational chemotaxis in bacterial foraging optimization: An analysis," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 919–941, August 2009.

[30] W. Tang, Q. Wu, and J. Saunders, "Bacterial foraging algorithm for dynamic environments," in *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, 0-0 2006, pp. 1324–1330.

[31] A. Biswas, S. Dasgupta, S. Das, and A. Abraham, "A synergy of differential evolution and bacterial foraging optimization for global optimization," *Neural Network World*, vol. 17, no. 6, pp. 607–626, 2007. [Online]. Available: http://www.softcomputing.net/nnworld2.pdf

[32] ——, "Synergy of PSO and bacterial foraging optimization - a comparative study on numerical benchmarks," in *Innovations in Hybrid Intelligent Systems*, ser. Advances in Soft Computing, E. Corchado, J. Corchado, and A. Abraham, Eds., vol. 44. Springer, 2008, pp. 255–263.

[33] D. Kim, A. Abraham, and J. H. Cho, "A hybrid genetic algorithm and bacterial foraging approach for global optimization," *Information Sciences*, vol. 177, pp. 3918–3937, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/B6V0C-4NG3TD4-2/2/44e1ed3d3aad9f8471aa167d43e3ee6e

[34] W. Korani, "Bacterial foraging oriented by particle swarm optimization strategy for pid tuning," in *GECCO '08: Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2008, pp. 1823–1826.

[35] E. Mezura-Montes and B. Hernández-Ocaña, "Modified bacterial foraging optimization for engineering design," in *Proceedings of the Artificial Neural Networks in Enginnering Conference (ANNIE'2009)*, ser. Intelligent Engineering Systems Through Artificial Neural Networks, C. Dagli, K. Bryden, M. Gen, S. Corns, G. Suer, and K. Tumer, Eds., vol. 19. ASME Press Series, November 2009. [Online]. Available: http://www.lania.mx/~emezura/util/files/Betania-ANNIE09-CR.pdf

[36] H. Beyer, *The Theory of Evolution Strategies*, G. Rozenberg, T. Bäck, A. Eiben, J. Kok, and H. Spanik, Eds. Springer, 2001.

[37] M. Richards and D. Ventura, "Choosing a starting configuration for particle swarm optimization," in *Neural Networks, 2004. Proceedings.*

2004 IEEE International Joint Conference on, vol. 3, July 2004, pp. 2309–2312 vol.3.

[38] J. Liang, P. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, June 2005, pp. 68–75.

[39] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, December 2009.