International Journal of Software Engineering and Computer Systems (IJSECS)
ISSN: 2289-8522, Volume 3, pp. 138-145, February 2017
©Universiti Malaysia Pahang
DOI: http://dx.doi.org/10.15282/ijsecs.3.2017.10.0032

A GLOBAL AFRICAN BUFFALO OPTIMIZATION

M. A. K Azrag*, T. A. Abdul Kadir, J.B. Odili, M. H. A Essam

Faculty of Computer Systems & Software Engineering,
Universiti Malaysia Pahang, Kuantan 26300, Malaysia
E-mail: mohammed87kunna@gmail.com

ABSTRACT

In this paper, a modified version of the African Buffalo Optimization algorithm with emphasis on global search is proposed. Two different equations with the values of their upper and lower boundaries are selected to be tested. The experimental result illustrates the difference in performance between the modified and the original ABO algorithm toward the optimum solution. At the end of all experimental procedures, it was observed that the Global African Buffalo Optimization (GABO) converges very quickly towards the optimal solution with a few buffalos. However, the experimental result show that GABO convergence still needs improvement as it tends to have unpromising convergence towards the end of a large iteration. In any case, GABO is a promising optimization method.

Keywords: African Buffalo Optimization, Global African Buffalo Optimization, Convergence Speed, Sphere Function, Rosenbrock Function

INTRODUCTION

Within the past few decades, Nature-inspired optimization algorithms have increasingly been gaining popularity in the areas of scientific and engineering research all over the world (Rauff, 2015). This development has thrilled many researchers and they have deduced various reasons for it. Some researchers argue that these Algorithms are successful because they were developed to replicate some of the most successful dynamics in biological, physical and chemical processes which occur naturally (Gandomi & Alavi, 2012; Priami, 2009). With this situation, the issue of choice of algorithm always surfaces (since there exists so many to choose from) whenever the need for optimization arises (Huang & Lam, 2002). There is this general understanding among researchers that the choice of the 'best' algorithm to solve a problem should largely be based on the nature of problem being faced. The No free lunch optimization theorem reinforced this line of thought (Wolpert & Macready, 1997; Xu, Caramanis, & Mannor, 2012). In fact, there is no agreement on the recommended principles guiding the choice of algorithms when faced with large-scale, nonlinear optimization problems (Ellison, Finn, Qin, & Tang, 2015).

The African Buffalo Optimization (ABO) algorithm was proposed by Odili and Kahar with inspiration from the natural behavior of the African buffalos (Odili & Kahar, 2015a). The ABO, though a relatively new algorithm has been successfully applied to solving benchmark travelling salesman's problems (Odili, Kahar, & Noraziah, 2016), numerical function optimization (Odili & Kahar, 2015b) and tuning of PID parameters (Odili & Mohmad Kahar, 2016).

The rest of this paper is organized thus: section two discusses the modified African Buffalo optimization with emphasis on global search (GABO), section three explains the experimental setup and discussion of results while section four draws conclusions from the study.

THE WORKING OF THE GABO

With the ability of the buffalos to recognize the global best position in a standard ABO algorithm at step 3 (Odili, Kahar, & Anwar, 2015), the buffalos were made to perform a re-search of the best position found by the ABO buffalos to ensure that the global best position are the same and as close enough to the global optimum solution as possible. This can be seen in step 4. With this idea, some modifications of the standard ABO algorithm are proposed. The modified algorithm selected the best position which has already been found by the ABO buffalos and then set them as the best position to be tested by considering the lower and the upper values of the herds' best ($\boldsymbol{bg_{max}}$). After this, the algorithm then starts to search towards the optimum global solution. In step 5 if the global optimum solution was found, then ABO move to step 6 if not, it repeats steps 2-5. But the Step 6 in GABO is asking to ensure the global best is reached if not repeat step 2-6 (This is the primary difference between ABO and GABO). Step 7 outputs the optimum values which means the best minimum values. In this way, the buffalos can search for more domains. The Pseudocode of the proposed GABO is presented in Figure 1:

```
1- Randomly initialize the buffalos to nodes within the solution space;
2- Update the buffalo's exploitation using:

       $\boldsymbol{m_k}' = \boldsymbol{m_k} + \boldsymbol{lp1}(\boldsymbol{bg} - \boldsymbol{w_k}) + \boldsymbol{lp2}(\boldsymbol{bp.k} - \boldsymbol{w_k})$

       where $\boldsymbol{m_k}$ and $\boldsymbol{w_k}$ represents the exploitation and exploration moves
       respectively of the kth buffalo (k=1, 2…. N); $\boldsymbol{lp1}$ and $\boldsymbol{lp2}$ are learning
       factors; $\boldsymbol{bg}$ is the herd's best fitness and $\boldsymbol{bp}$, the individual buffalo's best
       location.
3- Update the exploration fitness of the buffalos using:

       $\boldsymbol{w_k}' = \frac{(\boldsymbol{w_k} + \boldsymbol{m_k})}{\boldsymbol{rand}}$

4- Set the best position bp to be the new dimension of the global best position
   $bg_{max}$.
5- Is the global best position found and $bg_{max}$ is updating? Yes, go to 6. If No,
   go to 2-5
6- If the stopping criteria is not met, go back to algorithm step 2, else go to 6
7- Output best solution.
```

Figure 1. Pseudocode of GABO

The performance of GABO was greatly improved with the introduction of the search for the optimum global solution of the best position that has already been found by buffalos into the original version of ABO. This improvement was evident through the experimental study carried out on the benchmark problem of Sphere and Rosenbrock functions. To further illustrate the relevance of this study, the experimental results with the two non-linear tested functions used in by are reported and discussed (Shi & Eberhart, 1999).

EXPERIMENTAL SETUP

For comparison, two nonlinear functions used in are used here. The first function is the Sphere function described by equation (1):

$$\boldsymbol{f}(\boldsymbol{x}) = \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}} \boldsymbol{x}^{\mathbf{2}} \quad (1)$$

Where $x = [x_1, x_2, \dots, x_n]$ is an n-dimensional real-valued vector. The second function is the Rosenbrock function described by equation (2):

$$\boldsymbol{f_1}(\boldsymbol{x}) = \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}} (\mathbf{100}(\boldsymbol{x_i} - \boldsymbol{x_i^2})^{\mathbf{2}} + (\boldsymbol{x_i} - \mathbf{1})^{\mathbf{2}}) \quad (2)$$

Following the suggestion in (Shi & Eberhart, 1999) and for the purpose of comparison, the lower and upper values are selected to be as the original values. Table 1 lists the initialization of the upper and lower values of the two function.

Table 1. The Lower and Upper Values

Function	**Lower and Upper Values**
$\boldsymbol{f}$	[-10 10]
$\boldsymbol{f_1}$	[-5 10]

Utilizing the original parameters of ABO to evaluate the GABO algorithm, different population sizes were used for each function and these population sizes are 10 original; 20 and 30 buffalos. The maximum number of iteration is set to 100 original: 200, and 300 and the dimension is 5. Each algorithm was tested 3 times to get the get the Mean global best position.

EXPERIMENTAL RESULT AND DISCUSSION

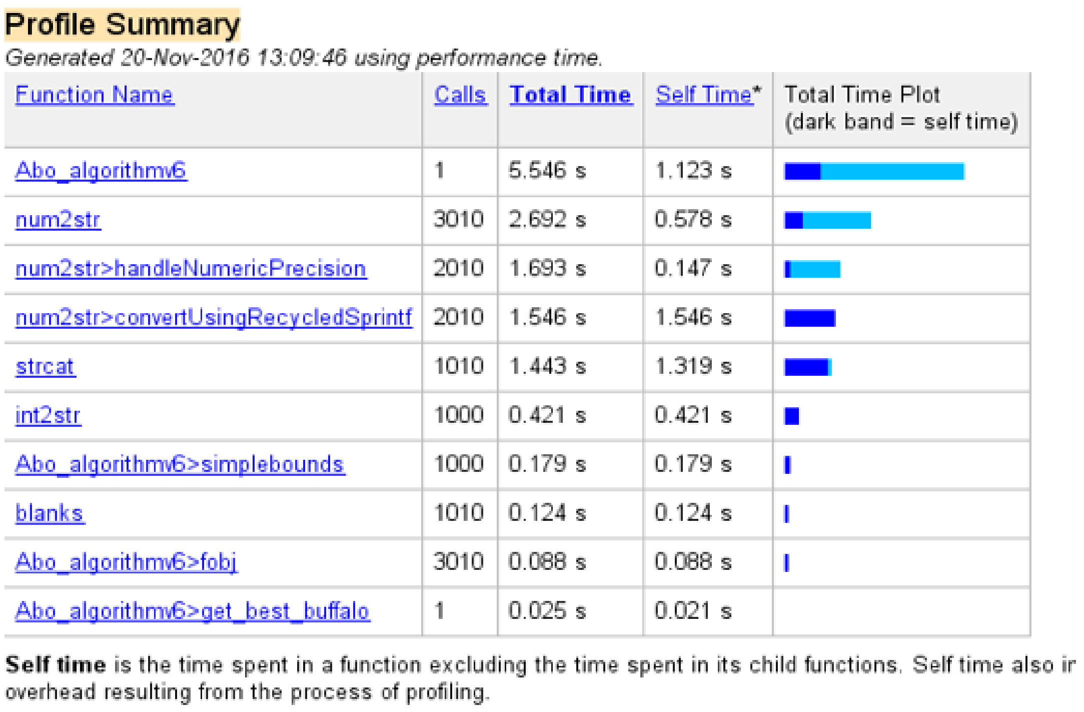
The global best position of GABO presented a very improved result, almost twice as good, when compared with the result of the ABO. Moreover, as shown in Tables 1 and 2, the convergence speed of the GABO towards the optimum values was faster when compared with those of the ABO. It is, however, noticeable that the convergence of the GABO was quick in both functions but slowed down as it approaches the optimal. It took GABO 0.028s to reach the best global position but the ABO reached the global best position in 0.025s. These comparisons are depicted in Tables 3 and 4. Finally, the performance speed of both algorithms is 5.5465s with 100 iterations for the ABO algorithm and 4.9725s for the GABO algorithm. The results in Tables 3 & 4 are obtained when the iteration is 100.

Table 2. ABO & GABO results for Sphere

Buffalo size	**Dimension**	**Iteration**	**Global Best Position GABO**	**Global Best Position ABO**
10	5	100	1.2785e-48	3.8689e-25
		200	2.1366e-67	1.4528e-43
		300	2.3178e-89	1.2441e-58
20	5	100	1.6503e-36	3.8689e-25
		200	6.8418e-67	4.3096e-47
		300	4.4579e-90	7.1516e-63
30	5	100	1.8175e-40	6.8646e-24
		200	1.2698e-67	2.9172e-45
		300	3.2768e-97	2.8229e-66

Looking at Table 2 where Sphere was tested using ABO and GABO on a 5-dimensional space with 10, 20, 30 buffalos and 100, 200, 300 iterations, the global best position of GABO has very good result that is almost twice as good as the result of ABO. Moreover, the convergence speed of GABO toward the optimum values are faster than those of ABO as plotted in Tables 3 and 4. Moreover, it is easy to observe that GABO convergences quickly in both functions but slows its convergence speed down when reaching the optimum. It takes 0.028s to get the best global position while ABO takes 0.025s as can be seen clearly in Tables 3 and 4. Finally, the total performance speed of the both algorithm is 5.546s with 100 iterations for ABO and 4.972s only for GABO. This result in Tables 3 & 4 was taken when the iteration is 100. Similarly, the GABO takes self-time 1.081s while ABO takes self-time 1.123s this may be due to the function simplicity.

Table 3. Different Iterations results: ABO Convergence speed 100 iteration -Sphere



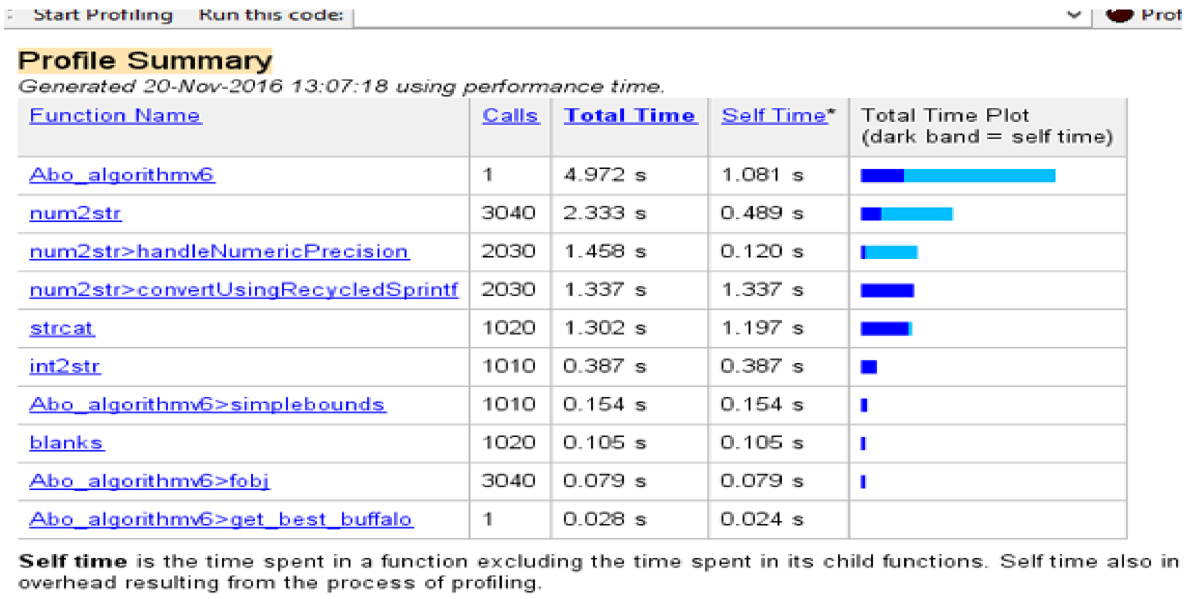
Profile Summary

Generated 20-Nov-2016 13:09:46 using performance time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
Abo_algorithmv6	1	5.546 s	1.123 s	
num2str	3010	2.692 s	0.578 s	
num2str>handleNumericPrecision	2010	1.693 s	0.147 s	
num2str>convertUsingRecycledSprintf	2010	1.546 s	1.546 s	
strcat	1010	1.443 s	1.319 s	
int2str	1000	0.421 s	0.421 s	
Abo_algorithmv6>simplebounds	1000	0.179 s	0.179 s	
blanks	1010	0.124 s	0.124 s	
Abo_algorithmv6>fobj	3010	0.088 s	0.088 s	
Abo_algorithmv6>get_best_buffalo	1	0.025 s	0.021 s	

Self time is the time spent in a function excluding the time spent in its child functions. Self time also ir overhead resulting from the process of profiling.

Table 4. Different Iterations results: GABO Convergence speed 100 iteration -Sphere



Start Profiling Run this code: Prof

Profile Summary

Generated 20-Nov-2016 13:07:18 using performance time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
Abo_algorithmv6	1	4.972 s	1.081 s	
num2str	3040	2.333 s	0.489 s	
num2str>handleNumericPrecision	2030	1.458 s	0.120 s	
num2str>convertUsingRecycledSprintf	2030	1.337 s	1.337 s	
strcat	1020	1.302 s	1.197 s	
int2str	1010	0.387 s	0.387 s	
Abo_algorithmv6>simplebounds	1010	0.154 s	0.154 s	
blanks	1020	0.105 s	0.105 s	
Abo_algorithmv6>fobj	3040	0.079 s	0.079 s	
Abo_algorithmv6>get_best_buffalo	1	0.028 s	0.024 s	

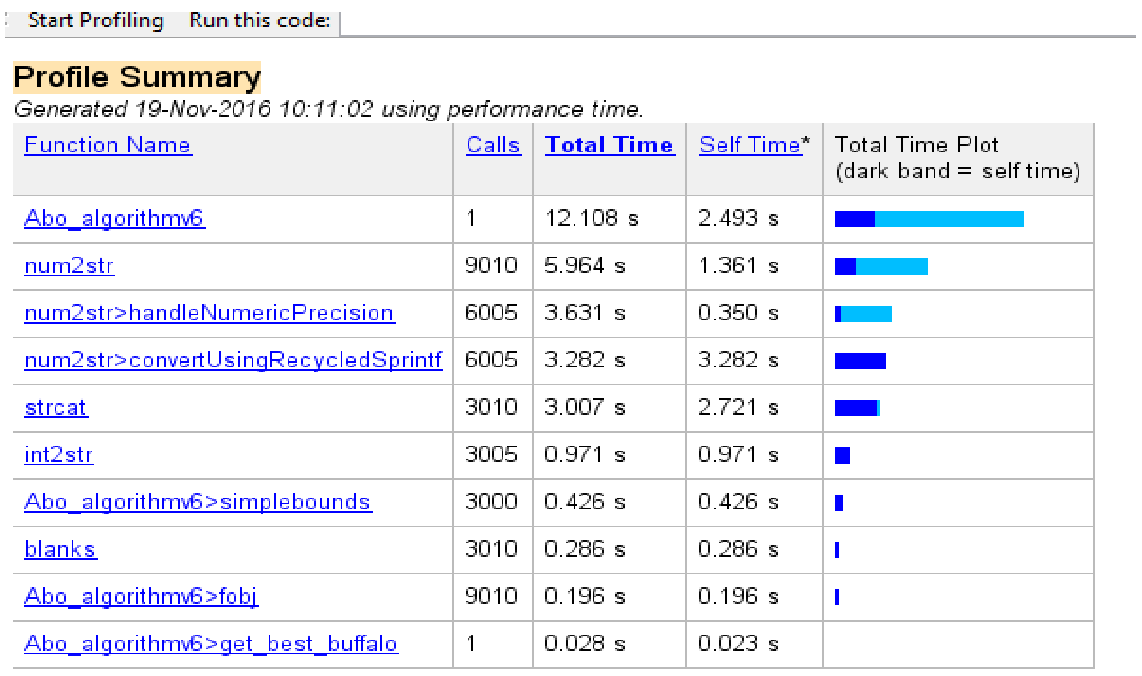
Self time is the time spent in a function excluding the time spent in its child functions. Self time also in overhead resulting from the process of profiling.

In Table 3, Rosenbrock function was tested using ABO and GABO. The parameters used are: 5-dimensional space with 10, 20, 30 buffalos and 100, 200, 300 iterations. It was discovered that the global best position of GABO improved better than the result of ABO as seen below in Table 5. Moreover, the convergence speed of GABO toward the optimum values are faster than ABO as plotted in Tables 6 and 7. However, it was observed that the GABO convergences quickly under both function but will slow its convergence speed down when reaching the optima. It took 0.040s to get the best global position while ABO took 0.028s as can be seen clearly in Tables 3and 4. Finally, the total time performance speed of the both algorithm is 12.1018s with 300 iterations for ABO and 11.254s only for GABO. This result in Figures 6 & 7 was taken when the iteration is 300.

Table 5. Rosenbrock Result: GABO/ABO Convergence speed with different iteration

Buffalo size	**Dimension**	**Iteration**	**Global Best Position GABO**	**Global Best Position ABO**
10	5	100	0.0051	0.0058
		200	0.0046	0.0090
		300	2.4168e-04	3.1955e-04
20	5	100	6.3621e-04	6.0301e-04
		200	3.1493e-04	0.0041
		300	3.6898e-07	1.6222e-04
30	5	100	3.9364e-04	0.0107
		200	1.1100e-04	3.2350e-04
		300	6.7848e-05	4.6605e-04

Table 6. Different Iterations results: GABO Convergence speed 300 iteration Rosenbrock



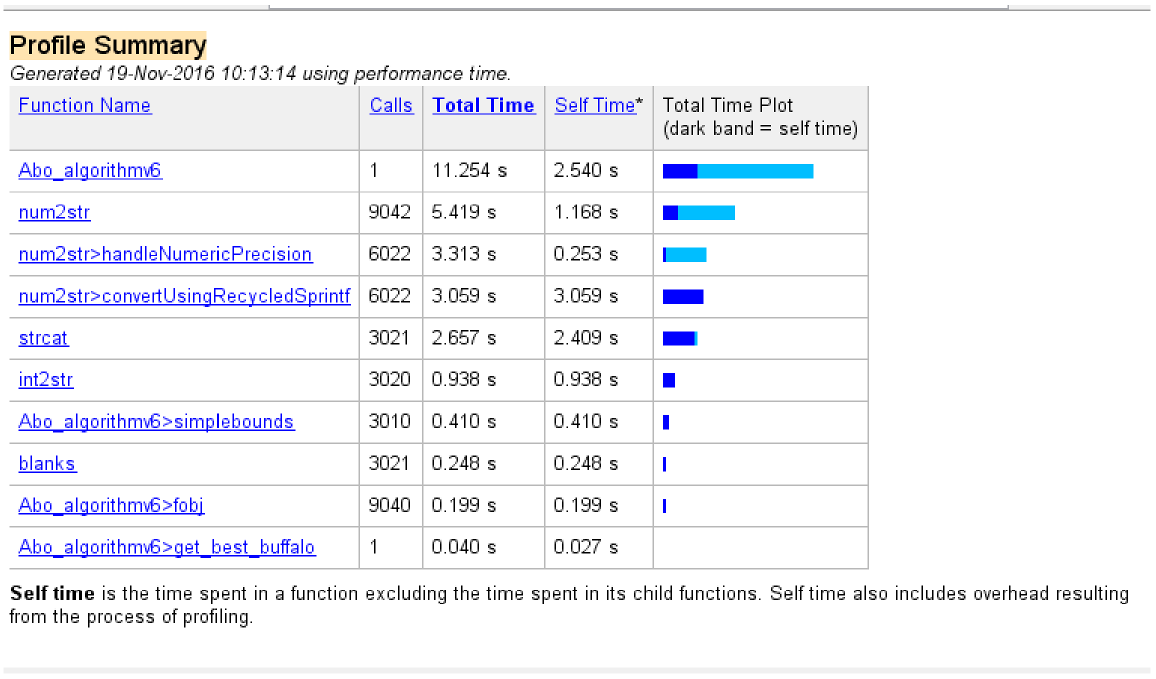
Start Profiling Run this code:

Profile Summary

Generated 19-Nov-2016 10:11:02 using performance time.

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
Abo_algorithmv6	1	12.108 s	2.493 s	
num2str	9010	5.964 s	1.361 s	
num2str>handleNumericPrecision	6005	3.631 s	0.350 s	
num2str>convertUsingRecycledSprintf	6005	3.282 s	3.282 s	
strcat	3010	3.007 s	2.721 s	
int2str	3005	0.971 s	0.971 s	
Abo_algorithmv6>simplebounds	3000	0.426 s	0.426 s	
blanks	3010	0.286 s	0.286 s	
Abo_algorithmv6>fobj	9010	0.196 s	0.196 s	
Abo_algorithmv6>get_best_buffalo	1	0.028 s	0.023 s	

Table 7. Different Iterations results: ABO Convergence speed 300 iteration Rosenbrock



Profile Summary

Generated 19-Nov-2016 10:13:14 using performance time.

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
Abo_algorithmv6	1	11.254 s	2.540 s	
num2str	9042	5.419 s	1.168 s	
num2str>handleNumericPrecision	6022	3.313 s	0.253 s	
num2str>convertUsingRecycledSprintf	6022	3.059 s	3.059 s	
strcat	3021	2.657 s	2.409 s	
int2str	3020	0.938 s	0.938 s	
Abo_algorithmv6>simplebounds	3010	0.410 s	0.410 s	
blanks	3021	0.248 s	0.248 s	
Abo_algorithmv6>fobj	9040	0.199 s	0.199 s	
Abo_algorithmv6>get_best_buffalo	1	0.040 s	0.027 s	

Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling.

CONCLUSION

The GABO algorithm was introduced in this study through the incorporation of the optimum global solution already identified by the buffalos into the original version of the ABO. After this, the best position of ABO was projected as the new search dimension to find the optimum values. To investigate the proposed method, two functions were employed. The results of the experiments showed that the GABO exhibited fast convergence when compared with the ABO. Nevertheless, though it was observed that the GABO algorithm works efficiently, we do not rule out the need for further experiments on more complex multimodal, separable and non-separable functions to further validate the search capacity of GABO. Moreover, there is need to apply GABO to other optimization search landscapes such as urban transportation problems, job scheduling, dynamic modeling, vehicle routing etc.

ACKNOWLEDGEMENT

The authors appreciate the Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang, Kuantan 26300, Malaysia for funding this research study.

REFERENCES

Ellison, C. L., Finn, J., Qin, H., & Tang, W. M. (2015). Development of variational guiding center algorithms for parallel calculations in experimental magnetic equilibria. *Plasma Physics and Controlled Fusion, 57*(5), 054007.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: a new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation, 17*(12), 4831-4845.

Huang, H.-J., & Lam, W. H. (2002). Modeling and solving the dynamic user equilibrium route and departure time choice problem in network with queues. *Transportation Research Part B: Methodological, 36*(3), 253-273.

Odili, J. B., Kahar, M. N., & Noraziah, A. (2016). Solving Traveling Salesman's Problem Using African Buffalo Optimization, Honey Bee Mating Optimization & Lin-Kerninghan Algorithms. *World Applied Sciences Journal, 34*(7), 911-916.

Odili, J. B., & Kahar, M. N. M. (2015a). African Buffalo Optimization (ABO): a New Meta-Heuristic Algorithm. *Journal of Advanced & Applied Sciences, 03*(03), 101-106.

Odili, J. B., & Kahar, M. N. M. (2015b). Numerical Function Optimization Solutions Using the African Buffalo Optimization Algorithm (ABO). *British Journal of Mathematics & Computer Science, 10*(1), 1-12.

Odili, J. B., Kahar, M. N. M., & Anwar, S. (2015). African Buffalo Optimization: A Swarm-Intelligence Technique. *Procedia Computer Science, 76*, 443-448.

Odili, J. B., & Mohmad Kahar, M. N. (2016). African Buffalo Optimization Approach to the Design of PID Controller in Automatic Voltage Regulator System. *National Conference for Postgraduate Research, Universiti Malaysia Pahang, September, 2016*, 641-648.

Priami, C. (2009). Algorithmic systems biology. *Communications of the ACM, 52*(5), 80-88.

Rauff, J. (2015). Nature-Inspired Optimization Algorithms. *Mathematics and Computer Education, 49*(3), 208.

Shi, Y., & Eberhart, R. C. (1999). *Empirical study of particle swarm optimization.* Paper presented at the Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation, 1*(1), 67-82.

Xu, H., Caramanis, C., & Mannor, S. (2012). Sparse algorithms are not stable: A no-free-lunch theorem. *IEEE transactions on pattern analysis and machine intelligence, 34*(1), 187-193.