Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 76 (2015) 443 – 448

Procedia Computer Science

2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS 2015)

African Buffalo Optimization: A Swarm-Intelligence Technique

Julius Beneoluchi Odili*, Mohd Nizam Mohmad Kahar, Shahid Anwar

Faculty of Computer Systems and Software Engineering, Univesity Malaysia Pahang, Gambang, Kuantan, Malaysia

Abstract

This paper proposes a new optimization technique called the African Buffalo Optimization. The African Buffalo Optimization (ABO) draws its inspiration from the behavior of African buffalos in the vast African forests and savannahs. African buffalos are a wild species of domestic cattle and are always mobile tracking the rainy seasons in different parts of Africa in search of lush green pastures to satisfy their large appetites. Our interest is in their organizational ability through two basic sounds in search of solutions. Experiments carried out using this novel algorithm in solving some benchmark Travelling Salesman's Problem when compared with the results from some popular optimization algorithms show that the ABO was not only able to obtain better solutions but at a faster speed.

Keywords: African Buffalo Optimization, African buffalos, Optimization, Nature-inspired, Bio-inspired, Travelling Salesman's Problem.

1. Introduction

The need for cost reduction, in a quest for profit maximization, has led researchers to investigate the possibility of drawing inspiration from nature to attempt solutions to myriads of problems in science, engineering, technology and industrial processes. These efforts have yielded several dividends leading to the proposal of a number of bio-inspired algorithms such as the Ant Colony Optimization (ACO) [1, 2], Honey Bee Mating Optimization (HBMO) [3], Particle Swarm Optimization (PSO) [4] Genetic Algorithm (GA) [5], Improved Genetic Algorithms (IGA) [6] and so on. These algorithms have been applied to solve a number of optimization problems such as Travelling Salesman's Problem [1], Vehicle Routing [4], Networking and other logistic problems [2], among others, with amazing results. In general, bio-inspired algorithms exploit the intelligent behavior of plants, animals and other natural elements in the ecosystem in a competitive or cooperative manner in arriving at solutions. These algorithms exploit the imprecision, haphazard and stochastic attitudes of these biological elements in arriving at marvelous solutions. In spite of the great achievements of these scientific efforts, it has been discovered that most of these

* Corresponding author. Tel.: +60102384948

E-mail address: odili_julestyahoo.com

Peer-review under responsibility of organizing committee of the 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS 2015)
doi:10.1016/j.procs.2015.12.291

algorithms require further refinements to make them faster and achieve better results as they are prone to premature convergence, delay in arriving at solutions, inability to explore and exploit the search space, the use of several parameters and so on [7] This is the motivation for this research.

African Buffalo Optimization (ABO) is an attempt to develop a user-friendly, robust, effective, efficient, yet, simple-to-implement algorithm that will demonstrate exceptional capacity in the exploitation and exploration of the search space. ABO attempts to solve the problem of pre-mature convergence or stagnation by ensuring that the location of each buffalo is regularly updated in relation to the particular buffalo's best previous location and the present location of the best buffalo in the herd. In a situation, for instance, where the leading (best) buffalo's location is not improved in a number of iterations, the entire herd is re-initialized. Tracking the best buffalo ensures adequate exploration of the search space and tapping into the experience of other buffalos as well as that of the best buffalo enables the ABO to achieve adequate exploitation. Similarly ABO ensures fast convergence with its use of very few parameters, primarily the learning parameters **lp1** and **lp2**. These parameters enable the movement of the animals towards greater exploitation or exploration depending on the focus of the algorithm at a given iteration.

The African Buffalo Optimization models the three characteristic behaviors of the African buffalos that enable their search for pastures. First is their extensive memory capacity. This enables the buffalos to keep track of their routes through thousands of kilometers in the African landscape. Moreover, buffalos are very cooperative. They are about the only animal species that dare to risk their lives in order to defend one of their own in dangers, hence the second attribute of the buffalos is their cooperative cum communicative ability whether in good or bad times. This they do through their *waaa* vocalizations with which they ask the herd to keep moving because the present location is unfavorable, lacks pasture or is dangerous. In other instances, the same '*waaa*' sound is used to invite other buffalos to come to the aid of other animals in danger. Basically the *waaa* is an alarm call. On the other hand, the *maaa* vocalizations are used to signal to the buffalo herd to stay on to exploit the present location as it holds promise of good grazing pastures and is safe. The third attribute of the buffalos is their democratic nature borne out of extreme intelligence. In cases where there are opposing calls by members of the herd, the buffalos have a way of doing an 'election' where the majority decision determines the next line of action. These three characteristics mark out African buffalos as one of the most organized and successful herbivores of all times [8].

This rest of this paper is organized this way: the second section discusses the African Buffalo Optimization (ABO) algorithm; the third section is concerned with experiment and discussion of results; the fourth section draws conclusions on the study and the fifth, acknowledges the support for the study.

2 African Buffalo Optimization

African Buffalo Optimization algorithm, basically models the three principal aforementioned characteristics of the African Buffalo. The '*maaa'* sound of buffalo k $(\mathrm{k} = 1, 2, 3 \ldots \mathrm{n})$ is represented by $\mathbf{m.k}$ and the '*waaa'* sound is represented by $\mathrm{w.k.}$ Mathematically, the democratic Equation (1) below determines the movement of the buffalos.

1. Objective function *f(x)* $x = (x1, x2, \ldots xn)T$
2. Initialization: randomly place buffalos to nodes at the solution space;
3. Update the buffalos fitness values using Eq. (1)

$$\mathrm{m.k} + 1 = \mathrm{m.k} + \mathrm{lp1}(\mathrm{bgmax} - \mathrm{w.k}) + \mathrm{lp2}(\mathrm{bpmax.k} - \mathrm{w.k}) \quad (1)$$

where $\mathrm{w.k}$ and $\mathrm{m.k}$ represents the exploration and exploitation moves respectively of the k^{th} buffalo (k=1,2,……..N) ; lp1 and lp2 are learning factors; bgmax is the herd's best fitness and bpmax.k the individual buffalo's best

4. Update the location of buffalo k (bpmax.k and bgmax) using (2)

$$\mathrm{w.k} + 1 = \frac{(\mathrm{w.k} + \mathrm{m.k})}{\pm 0.5} \quad (2)$$

5. Is bg_{max} updating. Yes, go to 6. No, go to 2
6. If the stopping criteria is not met, go back to algorithm step 3, else go to step 7
7. Output the best solution.

Figure 1: ABO algorithm

Equation (1) has three main parts, namely, the memory part $(m.k + 1)$ which is an indication that the animals are aware that they have relocated from their former positions *(*$m.k$) to a new one. This is an indication of their extensive memory capacity that is a vital tool in their migrant lifestyle. The second part represents the cooperative attributes of the animals $lp1(bgmax - w.k)$. The buffalos are excellent communicators and are able to track the location of the best buffalo in each iteration. The last part of this equation $lp2(bpmax.k - w.k)$ brings out the exceptional intelligence of these animals. They are able to tell their previous best productive location in comparison to their present position. This enables them take informed decisions in their search for solutions. Eq. 2, basically, propels the buffalos to a new location following the outcome of Eq.1.

2.1 The movement of the buffalos

Two main equations control the movement of the buffalos within the solution space. These are Equations (1) and (2) (refer to Figure 1). The democratic Equation (1) provides the template for the movement or otherwise of the animals. The *waaa* update (move on to explore, eq. 2) provides for the actual adjustment of the herd movement given the two competing forces *(waaa and maaa* calls). The result is a new location for the animals.

The first equation has two major parameters, namely, the global maximum ($bgmax$) and the personal maximum ($bpmax.k$) positions: each exercising its influence over the animal's choices. The algorithm subtracts the *waaa* element ($w.k$) asking the animal to explore the search space from the maximum vector (($bgmax$ and $bpmax.k$) and then multiplies this by the learning parameters (**lp1, lp2**) usually set to between 0.1 to 0.6. Using the learning parameters 0.1 to 0.6 has so far proved effective in obtaining fast convergence. The sum of these products is then added to the *maaa* ($m.k$) elements (asking the animals to stay on to exploit the area) for the given dimension. The output here is fed into Eq. 2, resulting in the movement or otherwise of the buffalos in a particular iteration.

3. Experiments and discussion of results

The experiments were performed using a desktop computer running Windows 7, 64-bit Operating System, Intel Core [TM] , i7-3770 CPU@ 3.4GHz, 3.4GHz, 4GB RAM. In the experiments on the benchmark global optimization functions (Section 3.5), the benchmark function equations were coded in MATLAB programming language and were run using MATLAB 2012b tool. The data obtained from the experiment with the African Buffalo Optimization (ABO) was compared with results obtained from similar experiments using the Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Ant Colony Optimization and the Genetic Algorithm (GA) [9, 10] in investigating the Symmetric Traveling Salesman's Problem. These algorithms chosen to compare with the ABO in this study have posted some of the best results in literature.

3.1 Parameter setting

The parameters used in these experiments posted very good results. Further investigations are ongoing, however, to discover the parameter settings that will yield better results. For the experiments involving the Particle Swarm Optimization, the experimental parameters are: population size 200; iteration (T_{max}) 1000; intertia weight 0.85; C_1 2; C_2 2; rand1 (0,1); rand2(0,1).

Also, it is important to note the following vital equations for ASA-GS:

$$tcool = \alpha \times (N0.5 - 1.0))(\alpha \times N0.5) \quad (3)$$

$$\rho = e - (F(cbest) - F(cold)/tcurrent \times (10.0 \times N/OPT) \quad (4)$$

D* represents the number of nodes.

Table 1: Experimental parameters setting

ABO		**ACO**		**ASA- GS**		**HBMO**		**GA**	
Parameter	Values	*Parameters*	*Values*	Parameters	*Values*	Parameters	Values	Parameter	Values
Population	40	*ants*	*D**	Population	*No of nodes*	Queen	1	Generation	100
m.k	*1.0*	*β*	*5.0*	$t_{initial}$	*1000*	*Drones*	200	*β*	2.0
bgmax/ bpmax	0.6	*ρ*	*0.65*	*ρ*	*see Eq. 3*	Spermateca	50	*ρ*	0.1
lp1 / lp2	0.5	*α*	*1.0*	*Opt*	Tour length	Mating drones	50	Ro	0.33
w.K	1.0	□	*200*	*tv*	N/10	Brood	50	Crossover rate	1.0
N/A	-	*qo*	0.9	$t_{current}$	$t_{current}$*t_{cool}	*α*	*0.9*	*qo*	0.9
N/A	-	N/A	-	t_{cool}	See Eq.3	Mating flights	1000	□r	0.3
N/A	-	N/A	-	t_{end}	*0.005*	w1	3	□ *ρ*	0.2
N/A	-	N/A	-	t_{greedy}	β*N	w2	4	τ_{min}	τ_{max}/20
N/A	-	N/A	-	N/A	-	□	10^{-10}	τ_{max}	1-(1- *ρ*)
Total no of runs	50		50		50		50		50

3.2 Experimental data

To investigate the capacity of the ABO to solve combinatorial optimization problem, 13 benchmark Traveling Salesman's problem from the TSPLIB [11] ranging from 52 to 14051 cities were used. The choice of the TSP datasets is made in such a way as to test the performance of ABO in searching routes of less than 100 cities (Berlin52, St70, Pr76 and Rat99) to searching TSP problems of less than 200 cities (Pr107, Pr124, Ch130, Pr152 and U159) and finally to problems that run to some hundreds of cities (Tsp225, Rat575 and Brd14051). The stopping criterion is when there is no more improvement in the best result obtained by the algorithm.

3.3 Discussions of results

In evaluating the performance of the ABO, the authors compared the results obtained from using the ABO to find solutions to the listed benchmark TSP cases with the results obtained by using some other popular algorithms in literature, namely, Particle Swarm Optimization, Ant Colony Optimization and the Genetic Algorithm [9, 10]. The results are shown in Table 2.

Table 2: Comparative experimental results

TSPLIB Problem	**TSPLIB Values**	**ABO**		**PSO**		**ACO**		**GA**	
		Best	Rel. Error %	Best	Rel. Error %	Best	Rel. Error %	Best	Rel. Error %
Berlin52	7542	7542	0	7542	0	7549	0.09	7542	0
St70	675	676	0.15	717.53	6.3	696.62	3.2	710.49	5.19
Pr76	108159	108167	0.01	118028	9.13	110917	2.55	115329	6.63
Rat99	1211	1211	0	1278	5.53	1236	2.06	1269	4.79
Pr107	44303	44407	0.01	44436	0.3	44354	0.12	44417	0.26
Pr124	59030	59058	0.05	59283	0.43	59113	0.14	59247	0.37
Ch130	6110	6111	0.02	6181.4	1.15	6141	0.51	6158.3	0.79
Pr136	96772	96784	0.01	-	-	96785	0.01	-	-
Pr152	73682	73730	0.07	73898	0.29	73835	0.21	73872	0.26
U159	42080	42107	0.06	-		42080	0	-	-
Tsp225	3916	3917	0.03	-	-	4112.4	5.01	-	-
Rat575	6773	6777	0.06	6910	2.02	6876	1.52	6897	1.83
Brd14051	469385	469835	0.1	477346	1.7	476949	1.61	477304	1.69

Rel. Error % = Relative Error percentage; Best=Best result obtained by an algorithm

In Table 2, Relative Error % values were obtained with the formula:

$$Rel.Error = ((\text{Best} - \text{TSPLIB value})\ /\text{TSPLIB value}) * 100 \quad (5)$$

As can be seen in Table 2, the ABO outperformed the other algorithms (PSO, ACO and GA) in realizing the closest solution in all the test cases under investigation. Aside from getting the closest results to the optimal result, the ABO obtained optimal results in Berlin52 and Rat99, and the ACO in u159, the PSO and the GA could only obtain optimal result in Berlin52. Further analyses show that the cumulative relative error percentage of ABO is a mere 0.57% compared to PSO's 26.85%, ACO's 17.03% and GA's 21.81%. The outstanding performance of ABO gets rather more glaring when one considers that the ABO and ACO solved all the problems under investigation whereas the GA and PSO only attempted ten test cases each.

3.4. Performance cost consideration

Furthermore, the time needed to get optimal or near-optimal solution is very vital as time correlates with cost in business and production engineering. An efficient algorithm, therefore, has to be one that obtains good solutions at a reasonable time [12]. To achieve this, a number of experiments were done to examine the cost implication of ABO in terms of time taken to arrive at optimal or near-optimal solutions to the benchmarked Travelling Salesman's Problems. The results of the best CPU time spent were compared with those obtained by Genetic Algorithm (GA) [9], Honey Bee Mating Optimization (HBMO) [3], Ant Colony Optimization (ACO) [10], Simulated Annealing (SA) [13] and Adaptive Simulated Annealing with Greedy Search (ASA-GS) [12]. The experiments were carried out using benchmark TSP cities ranging from 16 to 14,051 cities. The results are presented in Table 3.

Table 3: Comparative CPU time

TSPLIB	ABO	GA	HBMO	ACO	SA	ASA-GS
Problem	(secs)	(secs)	(secs)	(secs)	(secs)	(secs)
Ulysses16	0.03	0.16	-	-	0.02	-
Eil51	0.04	1.16	0.17	112.1	3.77	3.91
Berlin52	0	2.77	0.19	116.7	3.07	3.83
St70	0.08	-	-	226.1	3.35	5.15
Pr76	0.08	6.73	0.53	272.4	2.63	5.49
KroA100	0.03	16.5	0.62	615.1	3.32	7.14
Tsp225	0.09	16.2	5.38	4039	-	-
Ch130	0.08	14.4	1.28	-	5.29	8
Rat99	0.09	26.4	0.58	-	4.83	-
Pr107	0.11	18.2	1.01	-	2	7.78
Pr124	0.07	20.5	1.08	-	1.93	9.01
Pr136	0.08	30.5	1.35	-	7.08	9.86
Pr152	0.09	31.1	2.21	-	3.69	10.85
Rat783	0.05	26.4	71.1	0.63	3.45	78.9
Brd14051	78.9	-	902	-	-	2081

Table 3 shows the capacity of ABO to obtain solutions at incredibly fast rate. ABO obtained optimal solutions faster than every other algorithm under review here in all the test cases; the only exception being in Ulysses 16 where the SA marginally obtained a better result of 0.02 to ABO 0.03 second. The ABO's exceptional performance becomes more glaring when one considers that the other algorithms are among the best in literature and were published only recently [10]. On algorithm-by-algorithm analysis, it took ABO a cumulative time of 79.7 seconds to solve all the

15 TSP instances under investigation (including the most time-consuming Brd14051) to GA's 210.81 seconds to solve 13 instances excluding Brd14051; HBMO's 987.87 seconds; ACO's 5381.69 seconds for just six TSP instances; SA's 44.43 for 13 instances excluding the time-consuming Brd14051 and ASA-GS's 2,229.92 seconds for 12 TSP instances. From the foregoing analysis, it is safe to say that the ABO is at least 12 times faster than the HMBO and 27 faster than ASA-GS. In the same vein, juxtaposing the TSP instances attempted by the other algorithms with the results obtained by ABO in those same instances, the ABO is 59.24 faster than SA; 277.38 faster than the GA and over 14,545 faster than ACO.

4. Conclusion

In this study, a novel optimization algorithm, the African Buffalo Optimization was proposed and validated using a number of benchmark Traveling Salesman Problems from the TSPLIB. The investigations show that the ABO has immense capacity to solve different kinds of optimization problems, many times obtaining the optimal solutions much faster than the popular optimization algorithms like the PSO, GA and ACO. We can safely conclude, therefore, that the ABO is a worthy addition to Swarm Intelligence techniques and recommend the use of ABO to solve knapsack problems, PID tuning of parameters etc.

5. Acknowledgement

The authors appreciate the contributions of the anonymous reviewers for their useful input to this study as well as the Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang, Kuantan, Malaysia. for the funding of this study through Grant PGRS 1403118.

References:

[1] G. Di Caro and M. Dorigo, "AntNet: Distributed stigmergetic control for communications networks," *Journal of Artificial Intelligence Research,* pp. 317-365, 1998.

[2] J. B. Odili, H. Deng, C. Yang, Y. Sun, M. C. d. E. Santo Ramos, M. C. G. Ramos*, et al.*, "Application of Ant Colony Optimization to Solving the Traveling Salesman's Problem," *Science Journal of Electrical & Electronic Engineering,* vol. 2013, 2013.

[3] Y. Marinakis, M. Marinaki, and G. Dounias, "Honey bees mating optimization algorithm for the Euclidean traveling salesman problem," *Information Sciences,* vol. 181, pp. 4684-4698, 2011.

[4] F. Hui-liana and L. Xian-lib, "Discrete particle swarm optimization for TSP based on neighborhood [J]," *Application Research of Computers,* vol. 2, p. 030, 2011.

[5] G. Dong, X. Fu, and H. Xue, "An Ant System-Assisted Genetic Algorithm For Solving The Traveling Salesman Problem," *International Journal of Advancements in Computing Technology,* vol. 4, 2012.

[6] R. T. Zheng, N. Ngo, P. Shum, S. Tjin, and L. Binh, "A staged continuous tabu search algorithm for the global optimization and its applications to the design of fiber bragg gratings," *Computational Optimization and Applications,* vol. 30, pp. 319-335, 2005.

[7] N. Kumbharana and G. M. Pandey, "A Comparative Study of ACO, GA and SA for Solving Travelling Salesman Problem," *International Journal of Societal Applications of Computer Science,* vol. 2, pp. 224-228, 2013.

[8] D. S. Wilson, "Altruism and organism: Disentangling the themes of multilevel selection theory," *The American Naturalist,* vol. 150, pp. s122-S134, 1997.

[9] F. Liu, "A dual population parallel ant colony optimization algorithm for solving the traveling salesman problem," *Journal of Convergence Information Technology,* vol. 7, 2012.

[10] M. G. UZ, M. S. Kiran, and E. ÖZCEYLAN, "A hierarchic approach based on swarm intelligence to solve the traveling salesman problem," *Turkish Journal of Electrical Engineering & Computer Sciences,* vol. 23, pp. 103-117, 2015.

[11] G. Reinelt, "TSPLIB—A traveling salesman problem library," *ORSA journal on computing,* vol. 3, pp. 376-384, 1991.

[12] B. Baritompa and E. M. Hendrix, "On the investigation of stochastic global optimization algorithms," *Journal of global optimization,* vol. 31, pp. 567-578, 2005.

[13] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search," *Applied Soft Computing,* vol. 11, pp. 3680-3689, 2011.