Journal of Theoretical and Applied Information Technology
31st August 2022. Vol.100. No 16
© 2022 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN IMPROVED SARDINE FEAST METAHEURISTIC OPTIMIZATION BASED ON LÉVY FLIGHT

MOHAMMAD FAIDZUL NASRUDIN [1], DWI YANUAR PANJI TRESNA [2], SALWANI ABDULLAH [3], HAFIZ MOHD SARIM [4], SARINA SULAIMAN [5]

[1,2,3,4] Centre for Artificial Intelligence Technology (CAIT), Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia

[5] School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor Bahru, Johor, Malaysia

E-mail: [1] mfn@ukm.edu.my, [2] tresna.panji@gmail.com, [3] salwani@ukm.edu.my, [4] hms@ukm.edu.my, [5] sarina@utm.my

ABSTRACT

The recently proposed Sardine Feast Metaheuristic Optimization (SFMO) is a population-based meta-heuristic optimization algorithm inspired by the commensal behavior of various predators while preying on sardines at sea, which is commonly known as a sardine feast. This algorithm is a behavior imitation of dolphins and sea birds (blue-footed boobies and brown pelican) preying on schools of sardines. SFMO suffers from premature convergence since it relies on the normal random function to calculate predators' movement or step size during exploration and exploitation. The proposed improved SFMO (SFMO-Lévy) aims to enhance the ability of predators to explore divergent areas using Lévy flight in the step size calculation. The performance of the SFMO-Lévy is investigated using several predefined benchmark functions for global optimization problems. The outcomes of the tests are then compared with those generated by the standard SFMO algorithm. The SFMO-Lévy outperforms the SFMO by providing an average of 23.44% fewer function evaluations. The results reveal that the proposed algorithm can solve the benchmark functions better than the standard SFMO algorithm.

Keywords: *Sardine Feast Metaheuristic Optimization, Lévy Flight, Metaheuristics, Global Optimization*

1. INTRODUCTION

Metaheuristic algorithms have been widely used in developing systems and solving problems [1]. Methods inspired from natural processes have been developed to solve many problems, including benchmark function [2], engineering [3], robotic [4], web caching [5], scheduling [6], and many more. The efficiency of bio-inspired algorithms is due to their ability to imitate the best features in nature. One of the main categories of the nature-inspired metaheuristic algorithms, called biologically inspired algorithms, is based on the survival of the fittest in biological systems, which causes evolution by natural selection over millions of years [7, 8].

The recently proposed SFMO is a population-based optimization algorithm inspired by the behavior of predators that prey on schools of sardines at sea during an event called the sardine feast [9]. SFMO is a biologically inspired algorithm that effectively employs several predator movements to find the optimal solution. The algorithm consists of 2 stages. Firstly, dolphins explore the ocean space for a school of sardines. Secondly, sea birds exploit a ball of sardines by diving and scooping while dolphins encircle it until the algorithm ends. SFMO successfully outperforms other modern meta-heuristic techniques in solving several benchmark functions for global optimization problems.

SFMO seems to use the two-phase random search method as suggested by [10], where one phase is a global search phase, and the other is a local search phase. The global phase can be viewed as an exploration phase to explore the entire feasible region. In contrast, the local phase can be viewed as an intensification phase that exploits

local information (e.g., gradient or nearest neighbor information).

The SFMO, however, may suffer from premature convergence because it fails to execute exploration and exploitation efficiently. Exploration is the ability to search space globally to avoid local optima. On the other hand, exploitation is the ability to search local space to improve optimal efficiency [11]. One of the premature convergence reasons is its dependency on simple randomization in its predators' step size calculation. The model gets trapped within the local minima due to the inability to explore the search space efficiently.

This paper proposes an improved SFMO by enhancing the predators' step size randomization. Randomization is an integral function of any meta-heuristic optimization technique that plays a vital role in optimizing a value. A random search pattern found in nature, such as the Lévy Flight. The contribution of this paper is the inclusion Lévy Flight to the existing SFMO algorithm.

The search pattern of Lévy Flight mimics the foraging behavior of predators, in our case, as in the sardine feast. [12] found that predators employ Lévy behavior when preys are sparse, while Brownian movement is used where preys are abundant and not sparsely distributed. The integration of random walks using Lévy flight increases the exploration potential of the SFMO and converges the model into the optimal solution.

This research paper is further organized as follows: A brief introduction of SFMO and Lévy Flight is discussed in Section 2. It also elaborates the proposed SFMO-Lévy. Section 3 deals with the implementation of the proposed algorithm and the benchmarking. Section 4 presents the result of the experiments and compares both algorithms' performance on the various benchmark functions. The conclusion and future scope of the research are presented in Section 5.

2. RELATED WORKS

2.1 Sardine Feast Metaheuristic Optimization (SFMO) Algorithm

Generally, the SFMO algorithm is conceptualized based on the commensalism foraging behavior of the dolphins, boobies, and pelicans in catching a bait ball of sardines during the sardine feast. It was first introduced in [13] as a two-stage searching algorithm for solving global optimization problems.

In the first stage, dolphins explore the ocean space for a school of sardines. They will stop searching once a member finds a good school, defined by a pre-set threshold value. In the second stage, sea birds: boobies, and pelicans start diving and scooping to exploit the sardines while the dolphins encircle the bait ball until the algorithm ends. Fig 1 illustrates both stages in SFMO.

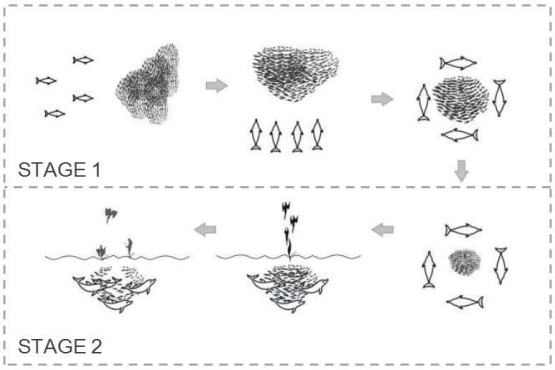

Figure 1. Two-stage SFMO algorithm

Formally, let $f: \mathbb{R}^n \to \mathbb{R}$ be the cost function which must be minimized. The function takes a candidate solution as an argument in a vector of real numbers. It produces a real number as output which indicates the objective function value of the given candidate solution. The gradient of f is unknown. The objective is to find a solution a for which $f(a) \leq f(b)$ for all b in the search space, which would mean a is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

Let D, B, and P be the number of dolphins, boobies, and pelican in the feast, respectively, each having a position $x_i \in \mathbb{R}^n$ in the search space. Let N be the number of all predators. In stage 1, the dolphins search randomly for a school of sardines. Once a dolphin finds a good school defined by a threshold value, let as bss (best sardine shoal) they will stop searching and start stage 2 which is encircling sardines to create a bait ball.

Let bbs be the size of the bait ball. Let g be the position of ball bait center. Based on g are real numbers of maximum distances of each predator types will prey. Let dss be the maximum distance of boobies will plunge dive. Let sss be the maximum distance of pelicans will scoop. **Fig 2** explains the definistion of *bbs*, *dss*, and *sss*.

Journal of Theoretical and Applied Information Technology
31st August 2022. Vol.100. No 16
© 2022 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

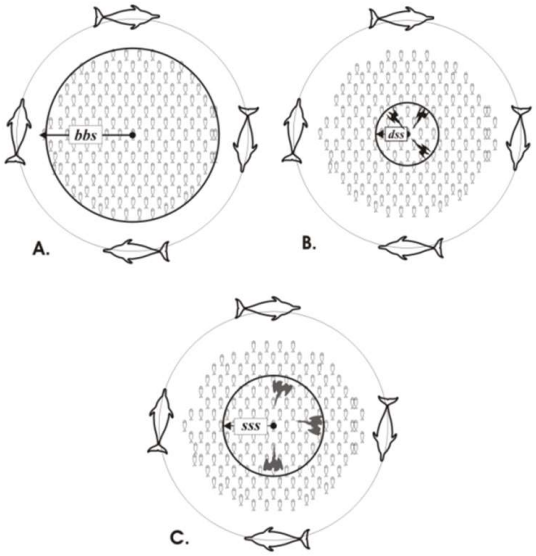

Figure 2. A. Bait ball size by dolphins (bbs), B. Diving search spot by boobies (dss), C. Scooping search spot by pelicans (sss)

The parameters *bds*, *dss*, and *sss* are selected according to the size of the search space where the general rule *bbs* < *sss* < *dss*. Basically, the *dss* and *sss* limit the step size of the predators. When generating new solutions (t +1) for, say, predator i, a new position is calculated based on:

$$x_i^{(t+1)} = x_{best}^t + r \times d \times rand \quad (1)$$

where x_{best}^t is the predator with the best solution at the center of bait ball at time t, r is the range of search spot (*dss* or *sss*), d is the direction (1 or -1), and *rand* is a random function;

The basic SFMO algorithm is then:

Pseudocode: SFMO

Initialize the population N that consists of number of each type of predators: D (dolphins), B (boobies), and P (pelicans).

Set the number of problem dimension, dim.

Repeat the Stage 1 until termination criterion is met (use a pre-defined threshold value):

– For each dolphin i = 1... D do:

– Assign the dolphin's position x at dimension d with a uniformly distributed random position: $x_{i,d} \sim U(b_l, b_u)$, where l and u, are the lower and upper boundaries of the search space.

– Calculate objective function and sort $f(x_i)$

– If the best $f(x_i) < f(g)$ update g the center position of the bait ball: $g \leftarrow x_i$

Repeat the stage 2 until termination criterion is met (use a maximum number of iteration):

– For each pelican i = 1, ..., P do:

– For each dimension d = 1, ..., dim do:

– Calculate direction, dir. If a random number is greater than 0.5 then direction is positive else negative: dir ← ±1

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (sss * dir * rand) based on Equation 1

– For each boobie i = 1, ..., B do:

– For each dimension d = 1, ..., dim do:

– Calculate direction, dir. If a random number is greater than 0.5 then direction is positive else negative: dir ← ±1

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (dss * dir * rand) based on Equation 1

– For each dolphin i = 1, ..., D do:

– For each dimension d = 1, ..., dim do:

– While dolphin inside the bait ball bbs do:

– Assign the dolphin's position at dimension d with a uniformly distributed random position: $x_{i,d} \sim U(b_l, b_u)$

– For each predator i = 1, ..., N do:

– Calculate objective function and sort $f(x_i)$

– If the best $f(x_i) < f(g)$ update the center position of the bait ball: $g \leftarrow x_i$

2.2 Lévy Flight

A Lévy Flight (LF) is a random walk in which the step lengths have a Lévy distribution, a heavy-tailed probability distribution. It represents the animal movement as a distribution of discrete straight-line movements modeled by the Lévy distribution, according to [7], often in terms of a simple power-law formula:

$$L(s) \sim |s|^{-1-\beta} \quad (0 < \beta \leq 2) \quad (2)$$

which has an infinite variance with an infinite mean. According to [14], for $\beta \geq 3$, the search path corresponds to Brownian motion under the central limit theorem, whereas for $\beta < 1$, it emerges as distributions that cannot be normalized.

Humans, birds, and other animals follow paths that have been modeled using LF when searching for food. It gives a smaller probability of returning to a previously visited site, making it convenient for

scattered and randomly distributed target sites. [15] concludes that Lévy flights' movement is determined and based on the distribution of the object of interest, which is why it is flexible and adaptive.

In this paper, the Mantegna method taken from [16] is used for generating step size, *s*, based on Lévy distribution as follows:

$$s = \frac{u}{|v|^{\frac{1}{\beta}}} \tag{3}$$

where u and v are drawn from a normal distributions. That is

$$u \sim N(0, \sigma_u^2), \qquad v \sim N(0, \sigma_v^2) \tag{4}$$

where

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

Using this step size, the new positions generated by Lévy walk around the best solution obtained so far will speed up the local search. Simultaneously, the new positions generated by far-field randomization and whose locations should be far enough from the current best solution will ensure the searching will not be trapped in a local optimum.

3. SFMO-LÉVY ALGORITHM

This section elaborates the incorporating of LF in the proposed algorithm.

3.1 Stage 1

In this stage, dolphins will start using the LF step size in their movement. Firstly, each dolphin will be given a uniformly distributed random position, to begin with. Then, the benchmark function is evaluated and sorted. The best dolphin is marked as the bait ball. Until the Stage 1 termination criterion is met, the new dolphins' positions will be based on the LF step size calculated using Eq (3).

3.2 Stage 2

In stage 2, boobies and pelicans will start using LF movement. Until the Stage 2 termination criterion is met, the boobies' and pelicans' new positions will be calculated based on their *sss* and *dss* values, respectively, and the LF step size value from Eq (3). The dolphins will continue foraging outside the bait ball using LF movement.

3.3 The SFMO-Lévy Algorithm

Based on the changes above, the proposed algorithm is then:

Pseudocode: SFMO-Lévy

Initialize the number of each type of predators: D, B, and P

Assign the dolphin's position with a uniformly distributed random position: $x_i \sim U(b_l, b_u)$, where l and u, are the lower and upper boundaries of the search space

Until the Stage 1 termination criterion is met (use a pre-defined threshold value), repeat:

– For each dolphin f = 1... D do:

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (factor * random() * step). Step is based on Eq. (3)

– Calculate and sort f(x)

– If the best $f(x_i) < f(g)$ update the center position of the bait ball: $g_i \leftarrow x_i$

Until the stage 2 termination criterion is met (use a maximum number of iteration), repeat:

– For each pelican i = 1, ..., P do:

– For each dimension d = 1, ..., n do:

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (factor * random() * sss * step). Step is based on Eq. (3)

– For each boobie i = 1, ..., B do:

– For each dimension d = 1, ..., n do:

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (factor * random() * dss * step). Step is based on Eq. (3)

– For each dolphin i = 1, ..., D do:

– For each dimension d = 1, ..., n do:

– Until $abs(g_d — (U(b_l, b_u))) > bbs$ (outside the bait ball) do:

– Assign the pelican's position: $x_{i,d} \leftarrow g_d$ + (factor * random() * step). Step is based on Eq. (3)

– For each predator i = 1, ..., N do:

– Calculate and sort $f(x_i)$

– If the best $f(x_i) < f(g)$ update the center position of the bait ball: $g \leftarrow x_i$

4. IMPLEMENTATION AND BENCH-MARKING

Both SFMO and SFMO- Lévy algorithms have been implemented in a Matlab program on an Intel Core™ i3 2.27 GHz computer. To demonstrate their improvement, both algorithms were tested to solve several standard global benchmark functions from the literature [17, 18].

We selected twelve unimodal and multimodal benchmark functions: the Shubert, Booth, Branin, De Jong, Dixon-Price, Lévy, Matyas, Michalewicz, Rastrigin Rosenbrock, Sphere, and Sum Squares functions. The details are presented in Table 1. Output from the algorithm is validated with the analytical or known solutions.

After implementing the algorithms using Matlab, we performed extensive simulations. Each benchmark function was run 100 times to perform meaningful statistical analyses. The algorithms stop when the variations of the function values are less than the tolerance of 10^{-5}. There are two common ways to compare algorithm performance. The first method compares the number of function evaluations for a given tolerance. The second method compares the accuracies for a fixed number of function evaluations. In this study, we used the first method.

In our experiments, we use different predators for different functions. Besides that, other parameters such as bbs, sss, dss, bss, and step size factor for LF are selected differently for each function, as shown in Table 2. We selected those parameters using a trial-and-error approach.

Table 1. Benchmark functions

Function	Dim	Definition	Bounds	Global Min
Shubert	2	$f(x) = (1.5 - x_1 + x_2x_2)^2 + (2.25 - x_1 + x_1x_1^2)^2 + (2.625 - x_1 + x_1x_1^3)^2$	$x_i \in [-4.5, 4.5]$, for all $i = 1, 2$	0
Booth	2	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$x_i \in [-10, 10]$, for all $i = 1, 2$	0
Branin	2	$f(x) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1 - t)\cos(x_1) + s$	$x_1 \in [-5, 10]$, $x_2 \in [0, 15]$	0.397 887
De Jong	20	$f(x) = \sum_{i=1}^{d} x_i^2$	$x_i \in [-5.12, 5.12]$, for all $i = 1, \ldots, d$	0
Dixon-Price	2	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{d} i(2x_i^2 - x_{i-q})^2$	$x_i \in [-10, 10]$, for all $i = 1, 2$	0
Lévy	2	$f(x) = \sin^2(\pi\omega_1) + \sum_{i-1}^{d-1}(\omega_i - 1)^2[1 + 10\sin^2(\pi\omega_i + 1)] + (\omega_d - 1)^2[1 + \sin^2(2\pi\omega_d)]$, where $\omega_i = 1 + \frac{x_i - 1}{4}$	$x_i \in [-10, 10]$, for all $i = 1, 2$	0
Matyas	2	$f(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	$x_i \in [-10, 10]$, for all $i = 1, 2$	0
Michalewicz	2	$(x) = -\sum_{i=1}^{d} \sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$, where $m = 10$	$x_i \in [0, \pi]$, for all $i = 1, 2$	-1.8013
Rastrigin	2	$f(x) = 10d + \sum_{i=1}^{d}[x_i^2 - 10\cos(2\pi x_i)]$	$x_i \in [-5.12, 5.12]$, for all $i = 1, 2$	0
Rosenbrock	2	$f(x) = \sum_{i-1}^{d-1}[(1 - x_i)^2 + 100(x_{i+1} - x_1^2)^2$	$x_i \in [-2.048, 2.048]$, for all $i = 1, 2$	0
Sphere	20	$f(x) = \sum_{i=1}^{d} x_i^2$	$x_i \in [-5.12, 5.12]$, for all $i = 1, \ldots, d$	0
Sum Squares	10	$f(x) = \sum_{i=1}^{d} ix_i^2$	$x_i \in [-10, 10]$, for all $i = 1, \ldots, d$	0

Table 2. SFMO and SFMO-Lévy parameters and their values

Function	SFMO							SFMO-Lévy									
	D	B	P	bbs	sss	dss	bss	D	B	P	bbs	sss	dss	bss	Step size factor*		
															D (S1)	B&P (S2)	D (S2)
Shubert	3	2	2	1.0	0.01	0.01	10^{-5}	3	2	2	2.0	1.0	0.5	10^{-5}	1.2	0.1	1.0
Booth	3	2	2	1.0	0.1	0.01	10^{-5}	3	2	2	1.0	1.0	0.1	10^{-5}	1.2	0.1	1.0
Branin	3	2	2	1.0	0.1	0.01	10^{-5}	3	2	2	1.0	1.0	0.1	10^{-5}	1.2	0.1	1.0
De Jong	2	3	2	0.3	0.1	0.001	10^{-5}	3	2	3	3.5	2.5	0.05	10^{-5}	1.0	0.1	1.0
Dixon-Price	3	2	2	1.0	0.3	0.003	10^{-5}	3	2	2	1.0	1.0	0.1	10^{-5}	1.0	0.1	1.0
Lévy	5	3	2	0.1	0.03	0.001	10^{-5}	3	2	2	0.1	0.1	0.001	10^{-5}	1.2	0.1	1.0
Matyas	3	2	2	1.0	1.0	0.1	10^{-5}	3	2	2	1.0	1.0	0.1	10^{-5}	1.2	0.1	1.0
Michale-wicz	3	3	3	0.3	0.3	0.003	10^{-5}	3	2	2	1.0	1.0	0.1	10^{-5}	1.0	0.1	1.0
Rastrigin	3	5	5	1.7	1.3	0.001	10^{-5}	5	3	2	0.1	0.1	0.01	10^{-5}	1.2	0.1	1.0
Rosen-brock	2	3	3	1.25	0.1	0.003	10^{-5}	3	3	3	2.0	1.2	0.2	10^{-5}	2.2	0.1	1.0
Sphere	2	2	2	0.3	0.1	0.001	10^{-5}	3	2	3	3.5	2.5	0.05	10^{-5}	1.0	0.1	1.0
Sum Squares	2	3	2	1.5	0.07	0.0005	10^{-5}	3	2	3	1.0	1.0	0.1	10^{-5}	1.2	0.1	1.0

* D (S1) = Dolphin in stage 1; B&P (S2) = Boobie and Pelican in stage 2; D (S2) = Dolphin in stage 2

5. RESULTS

The experiment outcomes are summarized in Table 3 where results from both the SFMO and SFMO-Lévy are presented. Besides that, the percentage changes of the results obtained using both algorithms are also calculated.

Table 3. Experiment Results

Function	SFMO	SFMO-Lévy	Change %
Shubert	1290 ± 544	1193 ± 540	-7.52%
Booth	544 ± 209	409 ± 130	-24.82%
Branin	389 ± 155	329 ± 102	-15.42%
De Jong	5853 ± 773	5658 ± 470	-3.33%
Dixon-Price	677 ± 206	412 ± 133	-39.14%
Lévy	4158 ± 4634	953 ± 459	-77.08%
Matyas	581 ± 462	406 ± 162	-30.12%
Michalewicz	487 ± 208	446 ± 245	-8.42%
Rastrigin	1869 ± 890	1703 ± 1474	-8.88%
Rosenbrock	3430 ± 2087	3224 ± 1541	-6.01%
Sphere	6618 ± 553	5754 ± 379	-13.06%
Sum Squares	8495 ± 1300	4459 ± 826	-47.51%

We can see that the SFMO-Lévy can find the global optima in all benchmark functions with fewer function evaluations. The Lévy function obtained the highest function evaluation reduction. On the other hand, the De Jong function recorded the lowest reduction. The average reduction of function evaluation by the SFMO-Lévy is 23.44%. Generally, for all the test functions, the SFMO-Lévy algorithm outperformed the SFMO.

6. DISCUSSION

The interesting observation is that the highest reduction recorded by the SFMO-Lévy is in the Lévy function that is known for its high number of local minima. The landscape of the Lévy function is shown in Figure 3. The predators, especially dolphins, can be easily be trapped inside a local minimum with little help from the normal randomization. This observation justify that the large step size from LF can help SFMO to avoid those local minima and escape premature convergence.

The results also show reductions of standard deviation values in ten out of twelve functions. It means that the results generated by the SFMO-Lévy are more stable since it can avoid local minima in the early stage of its execution. We believe the two

functions that recorded higher standard deviation values can be further reduced by allowing sufficient time for fine-tuning their parameters.

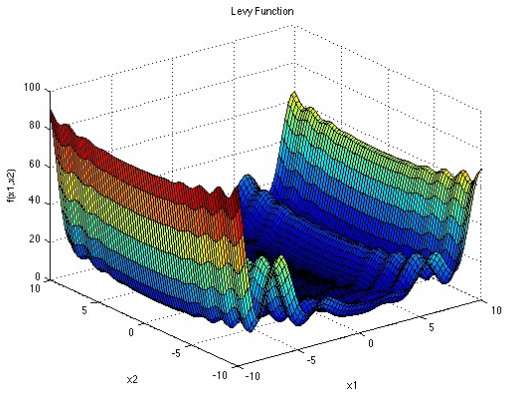

Figure 3.Lévy Function

Unimodal functions with high dimensions such as De Jong, Shere and Sum Squares also recorded a reduction of function evaluations. It only can be done by improving local space searching. It shows that LF leads to efficient exploitation of local search space by the predators (boobie and pelican) in stage 2 of the SFMO-Lévy algorithm.

7. CONCLUSION

In this study, we have embedded the LF in the SFMO algorithm. Experiments and result comparisons show that the proposed algorithm is superior to the original SFMO algorithm for all objective functions. The results are positive because LF allows predators in the algorithm to make nature-like diversification and intensification. Our major finding is that it leads to a reduction of function evaluations.

However, the SFMO-Lévy will add another three parameters to be tuned in terms of limitations. This drawback means that we have to fine-tune these parameters for a specific function. Another drawback is that the execution time of the SFMO-Lévy algorithm is generally higher than the SFMO algorithm because of the additional line of codes for LF. However, execution time is not reported in this paper. We believe the positive effects have outnumbered the negative ones.

The SFMO algorithm can easily be improved by embedding other random walk algorithms. One of them is the Brownian movement. The same concept can be applied as in the SFMO-Lévy. It is worth investigating since, in nature, predators employ Lévy behavior when preys are sparse, while Brownian movement is used where preys are abundant and not sparsely distributed. The Brownian movement might work well in stage 2 of the SFMO algorithm.

Acknowledgments

The authors would like to acknowledge the Department of Higher Education of Malaysia for financial support under research grant GUP-2020-065.

Conflict of Interest

The author(s) declare(s) that there is no conflict of interest.

REFRENCES:

[1] Rajabioun, R (2011). Cuckoo Optimization Algorithm, Applied Soft Computing, 11: 5508–5518.

[2] Beheshti, Z., Shamsuddin, S. M., & Sulaiman, S. (2014). Fusion global-local-topology particle swarm optimization for global optimization problems. Mathematical Problems in Engineering, 2014, 1–19.

[3] Nahlah Shatnawi, Shahnorbanun Sahran and Mohammad Faidzul (2013). A Memory-based Bees Algorithm: An Enhancement, Journal of Applied Sciences, 13(3): 497-502.

[4] Bahareh Nakisa, Mohammad Naim Rastgoo Mohammad, Faidzul Nasrudin, Mohd Zakree Ahmad Nazri (2015). A Multi-Swarm Particle Swarm Optimization with Local Search on Multi-Robot Search System, Journal of Theoretical and Applied Information Technology, 71(1): 129-136.

[5] Sulaiman, S., Shamsuddin, S. M., Forkan, F. and Abraham, A. (2008). Intelligent Web Caching Using Neurocomputing and Particle Swarm Optimization Algorithm, 2008 Second Asia International Conference on Modelling & Simulation (AMS), pp. 642-647.

[6] Cao P., Zhao, H. and Jiang, G. (2017) Integrated scheduling optimization of Yard Crane and Yard Truck in ship-loading operation, 2017 4th International Conference on Transportation Information and Safety (ICTIS), pp. 595-599.

[7] Gandomi, A. H. and Alavi, A. H. (2012). KrillHerd: A new bio-inspired optimization

algorithm, Communications in Nonlinear Science and Numerical Simulation, 17(12): 4831–4845.

[8] Xin-She Yang (2010). Nature Inspired Metaheuristic Algorithms, Second Edition, Luniver press.

[9] Marcos R. Rossi-Santos, Paulo A.C. Flores. (2009) 'Commensalism between Guiana Dolphins Sotalia guianensis and Sea Birds in the North Bay of Santa Catarina, Southern Brazil', The Open Marine Biology Journal, 3: 77-82.

[10] Schoen, F. (2002). Two-phase Methods for Global Optimization, in Handbook of Global Optimization, Volume 2, edited by P. M. Pardalos, and H. E. Romeijn, Kluwer Academic Publishers, Netherlands, 151-177.

[11] Khaire UM, Dhanalakshmi R (2020). High-dimensional microarray dataset classification using an improved adam optimizer (iAdam). J Ambient Intell Humaniz Comput 11:5187–5204.

[12] Humphries, N., Queiroz, N., Dyer, J. et al. (2010). Environmental context explains Lévy and Brownian movement patterns of marine predators. Nature 465: 1066–1069.

[13] Mohammad Faidzul Nasrudin, Fitranto Kusumo, Dwi Yanuar Panji Tresna et. al. (2021). Sardine Feast Metaheuristic Optimization: An Algorithm Based on Sardine Feeding Frenzy, Journal of Theoretical and Applied Informtion Technology, 99(17): 4349-4357.

[14] Viswanathan, G. M., S. V. Buldyrev, S. Havlin, M. G. E. da Luz, E. P. Raposo, and H. E. Stanley (1999). Optimising the success of random searches. Nature 401:911–914.

[15] Reynolds, A. M., & Rhodes, C. J. (2009). The Lévy Flight Paradigm: Random Search Patterns and Mechanisms. Ecology, 90(4): 877–887.

[16] Tarkhaneh, O., & Shen, H. (2019). Training of feedforward neural networks for data classification using hybrid particle swarm optimization, Mantegna Lévy flight and neighborhood search. Heliyon, 5(4): e01275.

[17] Jamil, M. and Yang, X. (2013). A literature survey of benchmark functions for global optimisation problems, International Journal of Mathematical Modelling and Numerical Optimisation, 4(2): 150.

[18] Surjanovic, S. & Bingham, D. (2014). Optimization Test Problems. http://www.sfu.ca/~ssurjano/optimization.html. Simon Fraser University.