International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 12 (2020) pp. 093-103
© MIR Labs, www.mirlabs.net/ijcisim/index.html

Received: 12 Jan, 2020; Accepted: 8 May, 2020; Published: 21 May, 2020

Non-linear Simplex Shuffled Frog Leaping Algorithm

Tarun K. Sharma[1] and Ajith Abraham[2]

[1] Department of Computer Science & Engineering, Graphic Era Hill University,
Bell Road, Dehradun, Uttarakhand, India
taruniitr1@gmail.com

[2] Machine Intelligence Research Labs, USA
ajith.abraham@ieee.org

***Abstract*: Shuffled frog leaping algorithm (SFLA) is a recent addition to the family of memetic algorithms that takes its inspiration from the natural foraging behavior of frogs. In SFLA the colony of frogs is divided into memeplexes of equal size. SFLA gathered the interest of research fraternity to solve many real world complex optimization problems. The basic structure of SFLA posses some inherent limitations. In order to overcome the limitation, in this study an enhanced variant of SFLA is proposed and named as NL-SFLA. Generally the initial population is generated using a traditional pseudo-random numbers which may not be much efficient. In NL-SFLA an attempt has been made to initialize the population of frog by integrating the concept of nonlinear simplex method of Nelder and Mead. Later modification is done in the frog distribution scheme in memeplexes to handle continuous optimization problems. Numerical results of NL-SFLA are compared with the state-of-art algorithms over a set of benchmark problems. Also the efficiency of the proposal is investigated on four real world problems. Simulated results signify the efficacy of the proposal.**

Keywords: SFLA, Shuffled frog leaping algorithm, Nelder-Mead, Local Search, Simplex, Nonlinear**.**

I. Introduction

Since decades memetic algorithms gathered the attention of the academicians, researchers as well as industrialists to solve intrinsic complex optimization problems exists in different spheres. Memetic algorithms are basically inspired from the foraging behavior of biological metaphors. A detailed study of such algorithms can be found in [1]. Shuffled frog leaping algorithm (SFLA), pioneered by Eusuff and Lansey in 2003 [2], is a recent addition to the family of memetic algorithms. Since its introduction SFLA gained wide attraction of researchers and academicians. Many recent applications of SFLA and its enhanced variants can be seen in [3]-[12]. SFLA is inspired by socio-cooperative behavior of frogs while searching for better food positions. In SFLA, the population of frogs is divided in equal numbers in each memeplexes. This division is based on the fitness value and frog with best fitness is placed in first memeplex, however frog with worst fitness being placed in last memeplex. Initially SFLA was developed to solve discrete optimization problems [2]. Since then, SFLA has gained popularity in solving multimodal, nonlinear and complex optimization tasks required in various fields of science& engineering, social science and management. A brief overview of basic SFLA and modified variant of SFLA and their applications are discussed in the Section – III.

Like other memetic algorithm SFLA too suffers from inherent limitations of sticking in local optima or premature convergence while searching for global solutions [13]. In this study an attempt is made to handle this inherent limitation. The initial population is generated by amalgamating non-linear simplex method of Nelder and Mead [14] with pseudo random generated by programming method. Secondly, basic SFLA performs better for discrete optimizations problems. A modification is done in the frog distribution scheme in memeplexes to handle continuous optimization problems.

The paper is structured as follows: In Section II, a brief overview and working of basic SFLA is presented. Literature review of SFLA and various variants of SFLA are discussed in Section III. Section IV discusses the motivation and methods used to enhance the efficacy of the basic SFLA. Test bed and real word problems are presented in Section V. Parameter Settings, Results and discussions are presented in Section VI. Finally conclusion drawn from the study is mentioned in Section VII.

II. Brief Overview: SFLA

Main idea behind the working model of SFLA is the combination of group evolution and exchange of information by shuffling at global level. First, the population is initialized, and the members (frogs) are arranged in the descending order of their fitness. Then the frogs are divided into memeplexes having equal number of frogs in each of it. Then, each memeplex is allowed to evolve for a predefined number of times. Every time, the worst frog moves towards the best frog in the memeplex. This modified position is kept only in case it improves the fitness of the worst frog; otherwise the worst frog is moved towards the global best frog. Still, if the position of the frog is not improved then the worst frog is discarded from the population and a new random frog replaces it. This elimination of a non-improving frog is known as censorship. After repeating this local search process for a fixed number of rounds for each memeplex, all the memeplexes are merged and the frogs are arranged in the descending order of their objective function values. These iterations continue till the termination criterion is not satisfied.

Local search within a memeplex encourages exploitation of the search space. As the evolution within a memeplex progresses, the frogs tend to converge near the best frog of the memeplex. Shuffling towards, the end of each evolutionary round depicts exploration. As the execution progresses, all the frogs converge towards the global best frog. The whole process of SFLA has been summarized in four steps below:

A. .Initialization Process

The population of frogs is initialized randomly between lower and upper bounds of the feasible search space in each dimension. This step is very similar to the process adopted in most of the metaheuristic algorithms. Set of all frogs is denoted by $XF_i = (xf_{i1}, xf_{i2,\dots,}xf_{iD})$. Individual frogs are generated using Eq. (1).

$$xf_{ij} = lb_j + rand(0,1) \times (ub_j - lb_j)$$

(1)

where $i = 1$ to F, $j = 1$ to D, lb_j is the lower bound for dimension and ub_j is the upper bound for dimension j

B. Sorting and Division Process

All the members are evaluated for the objective function and sorted in decreasing order of their fitness values. Then an ordered distribution of these frogs is done between m subsets called memeplexes. Number of frogs in each memeplex is n so that $F = m$ x n. This distribution is done in a specific order. Frogs 1 to m goes to memeplex1 to m. Frogs m+1 to 2m goes again to memeplex1 to m in the serial order. After this arrangement, the best frog X_b and the worst frog X_w in each memeplex are identified**.**

C. Local Search Process

This step performs intensification component of the search process. Worst frog is modified using Eq. (2) & (3).

$$DS_i = rand(0,1) \times (X_b - X_w)$$

(2)

$$X_{w,new} = X_w + DS_i; -DS_{\max} \le DS_i \le DS_{\max}$$

(3)

where $i = 1$ to D denotes the dimension. DS_{max} is the maximum allowed step size to avoid violation of the feasible space. $X_{w,\text{new}}$ is evaluated for its fitness. In case of improvement X_w is replaced by $X_{w,new}$. If the position is not improved than the same process is repeated by replacing X_b by the global best frog X_g in Eq. (2). Still if not improved then the X_w is replaced by a random new frog. This process of eliminating non-improving frog is called censorship.

This process of moving of worst frog is called leaping. Leaping is repeated for N_{gen} times in each memeplex. Process of leaping is represented in Fig. 1.

D. Shuffling Process

After repeating the local search for the m memeplexes, all frogs are combined and arranged in the descending order of their fitness again. This shuffling facilitates the exchange of information at global level.

Steps *(b)*, *(c)* and *(d)* are repeated until satisfaction of termination criterion.

Fig. 2, 3 and 4 illustrate the position of frogs in the search space after initialization, after evolution in memeplexes and towards convergence at the end, respectively. Fig .5 illustrates all the steps of the algorithm.

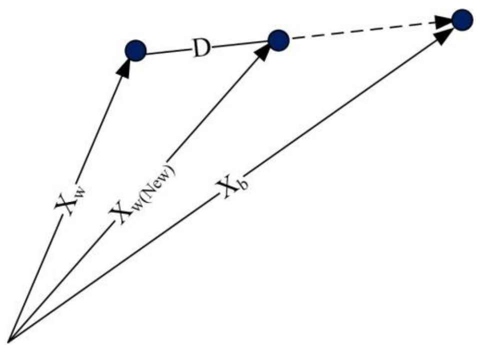


Figure 1. Leaping process of the worst frog during local iteration

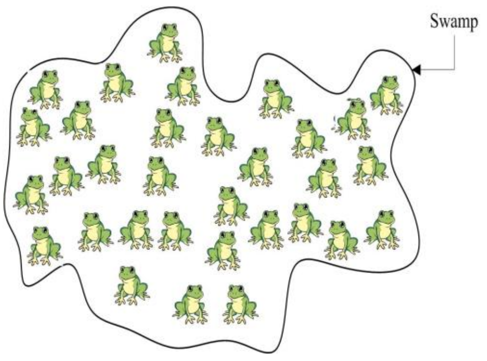


Figure 2. Illustration of randomly distributed frogs after initialization in SFLA

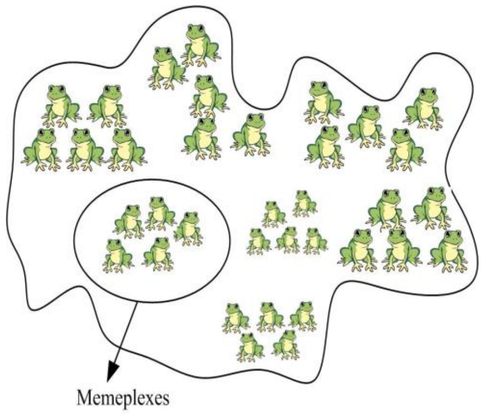


Figure 3. Illustration of convergence of frogs in respective memeplexes

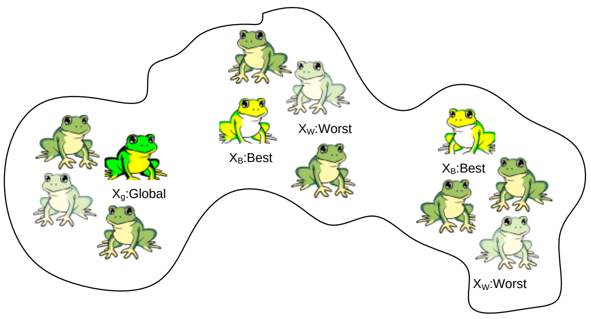


Figure 4. Illustration of convergence of frogs towards global best frog after the execution

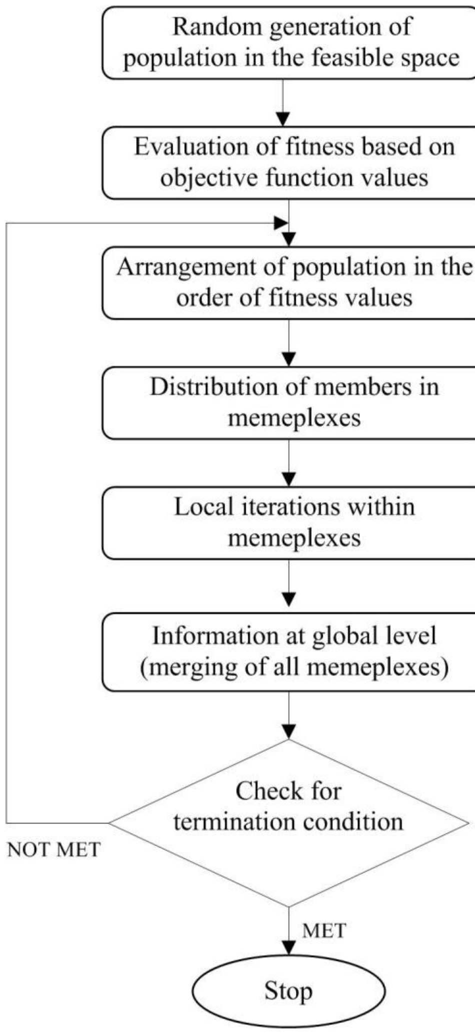

Random generation of population in the feasible space
Evaluation of fitness based on objective function values
Arrangement of population in the order of fitness values
Distribution of members in memeplexes
Local iterations within memeplexes
Information at global level (merging of all memeplexes)
Check for termination condition
NOT MET
MET
Stop

Figure 5. Flowchart of SFLA

III. Literature Review – SFLA

Moayedi et.al, in 2020 [8] embedded SFLA and WDO with ANN in order to magnify its prediction aptitude to simulate the shear strength of Soil. Tang et al.(2020) [9] proposed discrete SFLA (DSFLA) that embeds deterministic and random walk strategies to enhance the local search strategy in each memeplexes. This hybrid DSFLA is employed to maximize the strength of influential nodes in social networks. Sharma et al. (2020) [6] introduced local search strategy in SFLA and employed to solve a human resource problem. Li and Yan (2019) [10] introduced a hybrid model based on SFLA and bacterial foraging algorithm (BFA) in which levy flight is embedded to enhance the global search while maintain the diversity of the population using random grouping strategy. Further local search is improved using migration operation method to handle the optimization problem of redundancy allocation and system reliability. Huang and Song (2019) [11] modified local search in SFLA using the multi agent model to optimize the land use. Elattar (2019) [12], proposed a variant of SFLA that includes movement inertia concept of PSO and crossover and mutation operators of GA to improve local and global searching mechanism respectively. Later the variant is applied to solve heat, emission and economic dispatch problem. Janani et al. (2018) [15], applied SFLA to detect protein complexes in protein-protein interaction. Rajamohana and Umamaheswari (2018) [16] proposed a binary SFLA to address the problem of feature selection to assist customers with reliable reviews. Dash in 2018 [17], developed an improved variant of SFLA that embeds functional link artificial neural network (FLANN) to predict the currency conversion rate. Kaur and Mehta (2017) [18] proposed augmented SFLA to solve the workflow scheduling of resources in the cloud environment. Sharma and Pant (2017) [19] introduced the opposition based learning method to initialize the virtual population of frogs and applied to solve unimodal and multimodal benchmark problems. Tang et al. (2016) [20] introduced a new framework of SFLA using lévy flight. The local search mechanism is modified with lévy flight method whereas global search is enhanced with interaction learning rule. Dalavi et al. (2016) [21] designed a modified SFLA to improve premature convergence and implemented to determine the operations of hole making sequences optimally. Tripathy et al. (2015) [22] employed SFLA to solve the problem of multiprocessor scheduling. Jadidoleslam and Ebrahimi (2015) [23] proposed modified SFLA that encompasses a modified frog leaping strategy in each memeplexes to improve exploitation process. Further author(s) enhance the efficiency by introducing mapping procedure, penalty factor and integer encoding. Later this modified SFLA is applied to solve a generation expansion planning (GEP), a critical problem in power systems. Sharma et al. (2015) [24] embedded geometric centroid mutation, a probability based operator in SFLA to enhance the convergence rate. Bhattacharjee and Sarmah (2014) [25] introduced discrete SFLA that embeds local search mechanism of PSO and shuffled complex evolution. The proposal is implemented to solve 0/1 knapsack problem. Kumar and Kumar (2014) [26] applied SFLA to handle bidding strategy problem. Sarkheyli et al. (2014) [27] presented a brief literature review of SFLA.

IV. Motivation behind the Study

Two antagonists factor i.e. exploration (diversification) and exploitation (intensification) plays a significant role in the success of any memetic, nature/bio inspired and evolutionary algorithms. These two factors need to be balanced or justified to avoid trapping in local optima or slow/premature convergence. Population initialization plays an important role in any memetic algorithm. Generally random population of solutions is generated using defined pseudo random generator with the defined upper and lower bounds of feasible region. This population does not have any idea about the optimum locations in the search space that may help in expediting the search of global solution. In order to enhance the diversity as well as acceleration rate non-linear simplex method is embedded in the structure of SFLA. Secondly, the basic SFLA performs well on discrete optimization problems [2], so a modification is embedded while distributing frogs in different memeplexes to solve continuous optimization problems efficiently. These two modifications are introduced in SFLA and the resulting variant is termed as Nelder-Mead Shuffled Frog Leaping Algorithm (NL-SFLA).

Nelder-Mead and formation of memeplexes to handle the continuous problems are discussed below:

A. Nelder Mead

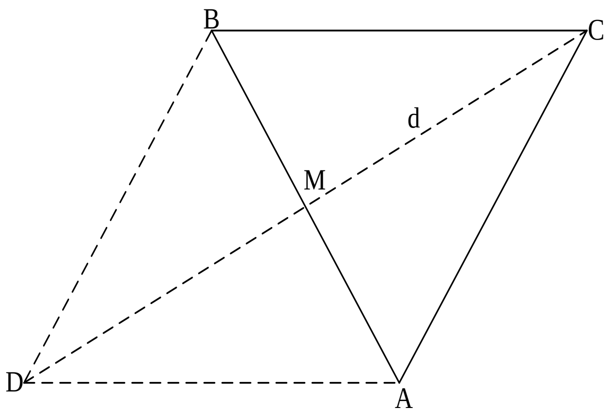
Non-Linear Simplex method (NLSM) a derivate free technique, introduced in 1965 by Nelder and Mead [14]. NLSM being the local direct search technique is best suited to solve unconstrained optimization problems especially the cases of minimization. In general there are four NLSM geometric transformations namely expansion, reduction, contraction and reflection. This transformation helps simplex in self improving as well as in converging to optimum. The objective fitness function value at the vertex of the simplex is used to select the suitable transformation. The worst vertex function value is updated with the better one in each transformation.

Brief Overview of geometric transformations

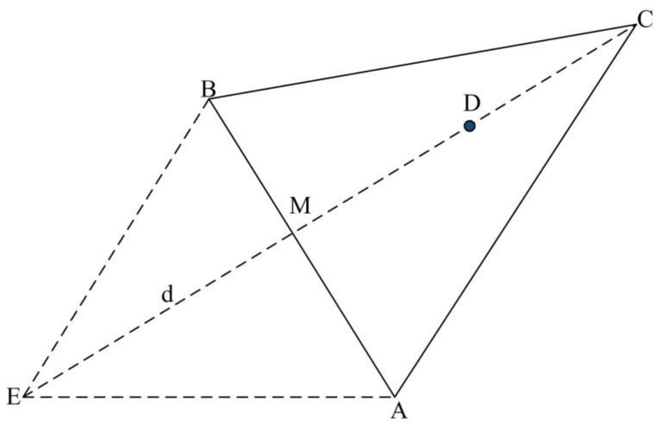
Considering the minimization case, initially only the vertex of the simplex having worst objective fitness value (worst point) is moved and an adjacent image of the worst point is generated. This is termed as reflection. If the reflected image of the point has the better fitness value in comparison to all other points, the simplex is expanded in that direction. Otherwise reflection is again performed with other worst fitness value. If the worst fitness value is comparatively good as the reflected one then contraction is performed. If the point generated through contraction is worse than the worst point, reduction transformation is initiated.
The transformation sequences are plotted in Fig 7(a – d).

B. Formation of Memeplexes

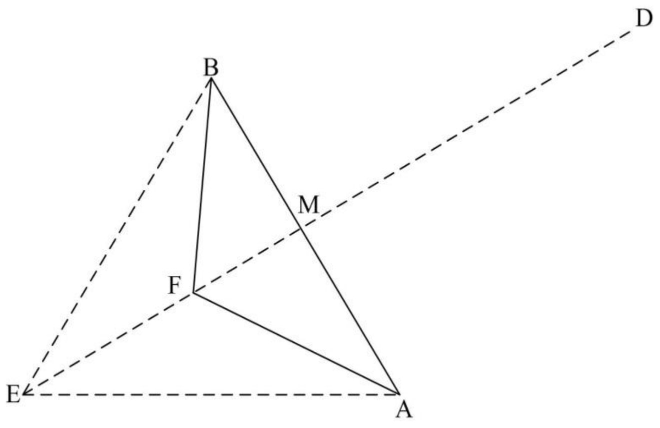
The memeplexes in SFL algorithm are formed by dividing a set of frogs based on their fitness value i.e. frog with best fitness value will be the part of 1st memeplex and the 2nd frog would be part of 2nd memeplex and so on (Fig. 6)



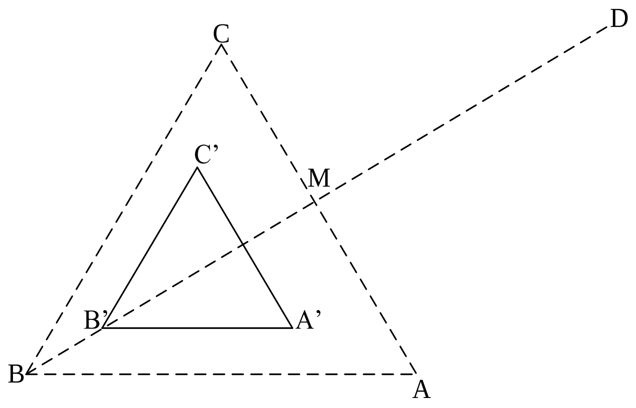
(a) Reflection of D to C



(b) Expansion of D to C



(c) Contraction of D to F



(d) Reduction of ABC to A'B'C'

Figure 7. (a – d) Processes involved in Non-Linear Simplex Method

x_1	x_2	…	X_M
x_{m+1}	x_{m+2}	…	x_{2m}
x_n	x_{2n}	…	x_{mn}
Memeplex (m_1)	Memeplex (m_2)	…	Memeplex (m_n)

Figure 6. Formation of memeplexes in SFLA

This formation of memeplexes leads to imbalance i.e. the performance of 1st memeplex would always be better than the last memeplex. This also leads to poor learning process and enhancing the performance of worst frog. To make the uniform performance of each memeplex, the formation of memeplex is done using the following [28] Eq. (4).

$$M_i^{'} = \left\{ \left(x_j, f_j\right) \middle| \begin{array}{l} x_j = \begin{cases} x_{i+m(j-1),} j = 1,3,...,n \\ x_{mj-1,} j = 2,4,...,n \end{cases} \\ f_j = \begin{cases} x_{i+m(j-1),} j = 1,3,...,n \\ x_{mj-1,} j = 2,4,...,n \end{cases} \end{array} \right\}$$

(4)

For an example if there are 6 frogs and divided into 2 memeplexes than as per the Eq. (4) 1st memeplex will have 1st, 3rd& 5th frog and 2nd, 4th and 6th will go to memeplex 2nd. This modification also supports in solving continuous optimization problems as initially SFL algorithm was proposed to solve discrete optimization problems

V. Test Bed and Experimental Setup

Two test bed are referred in this study. First test bed consists of fifteen benchmark problems [29] taken from literature to validate the proposal. Their characteristics are stated in Table 1. There are four unimodal (UM) and eleven multimodal (MM) problems. Unimodal problems validate the proposal in terms of local searching capability where as the rest of the problems verify the global searching capability of the proposed algorithm. Further to validate the efficacy of proposed algorithm, second test bed of four real life time problems (*RLTP*) [30]-[32] are taken into consideration to validate the efficiency of the proposal.

For unbiased simulated statistical result comparison nine (9) state-of-art algorithms (off course enhanced variants of some algorithms) are considered. The parameter settings of the state-of-art algorithms are presented in Table 2. 30 runs are performed with (different seeds/population each time) with the frog population size (*XF*) of 20; dimension (*D*) 30 with 5 (*m*) memeplexes i.e. in each memeplexes there would be 4 frogs (*n*). C++ is used to program and executed on windows 7 with *i*3-5005U CPU@2.00GHz having 4.00 GB RAM. D x 1E4 is fixed as maximum number of function evaluations as a termination criterion.

Stat.	Dimension	Limit Lower	Upper Limit	*F*(*x*)*	Problem Type
F1: Sphere	30	-100	100	0	UM
F2: Schwefel's (2.22)	30	-10	10	0	UM
F3: Schwefel's (1.2)	30	-100	100	0	UM
F4: Quartic	30	-1.28	1.28	0	UM
F5: Rastrigin	30	-5.12	5.12	0	MM
F6: Ackley	30	-32	32	0	MM
F7: Griekwank	30	-600	600	0	MM
F8: Rosenbrock	30	-10	10	0	MM
F9: Penalized	30	-50	50	0	MM
F10: Weierstrass's	30	-0.5	0.5	0	MM
F11: Zakharov	30	-5	10	0	MM
F12: Alpine	30	-10	10	0	MM
F13: Saloman	30	-100	100	0	MM
F14: Periodic	30	-10	10	0.9	MM
F15: Inversted Cosine	30	-1	1	0	MM

Table 1. Characteristics of benchmark problems

A. RLTP-1: Estimation of Parameters of Frequency Modulated Sound Waves [31][32]

In most of the music systems in modern era, synthesis of frequency modulation plays a significant role. In general there are six sound wave parameters that need to be optimized for an FM Synthesizer namely α1, ω1, α2, ω2, α3, and ω43. Estimated and targeted sound waves are presented below:

$$sw(t) = \alpha_1 \sin(\omega_1 t\theta + \alpha_2 \sin(\omega_2 t\theta + \alpha_3 \sin(\omega_3 t\theta)))$$
$$sw_0(t) = 1.0\sin(5.0t\theta - 1.5\sin(4.8t\theta + 2.0\sin(4.9t\theta)))$$

Range for the parameters is [-6.4, 6.35] and $\theta = \frac{2\pi}{100}$.

The objective (Eq. (2)) is to optimize the sum of squared errors between the estimated and targeted sound waves:

$$f(X) = \sum_{t=0}^{100} (sw(t) - sw_0(t))^2$$

B. RLTP-2: Optimal Thermohydralic Performance of an Artificially Roughened Air Heater [30]

This is a simple maximize optimization problem. The mathematical formulation of this problem is given as below:

$$\textit{Maximize } F = 2.51 \times \ln e^+ + 5.5 - 0.1 M_R - H_G$$

and

$$M_R = 0.95 x_2^{0.53}; \; H_G = 4.5(e^+)^{0.28}(0.7)^{0.57}$$

$$e^+ = x_1 x_3 (\frac{\breve{f}}{2})^{\frac{1}{2}}; \; \breve{f} = (f_s + f_r)/2$$

$$f_s = 0.079 x_3^{-0.25}; \; f_r = 2(0.95 x_2^{0.53} + 2.51 \times \ln(\frac{1}{2x_1})^2 - 3.75)^{-2}$$

The variables are bounded as:

$$0.02 \le x_1 \le 0.8$$
$$10 \le x_2 \le 40$$
$$3000 \le x_3 \le 40000$$

C. RLTP-3: Spread Spectrum Radar Poly phase Code Design [30] [31]

This problem is modeled as minmax non-convex nonlinear optimization problem having numerous local optima's in continuous search space.

$$Minimize\ f(X) = \max imize\{\phi_1(X), \phi_2(X), ..., \phi_{2n}(X)\}$$

The vector

$$X = (x_1, x_2, ..x_D) \in R^D \mid 0 \le x_i \le 2\pi, i = 1, 2, ..., D$$

where

$$n = 2D - 1$$

$$\phi_{2j-1}(X) = \sum_{i=j}^{D} \cos(\sum_{k=|2j-i-1|+1}^{i} x_k), j = 1, 2, ..., D$$

$$\phi_{2j}(X) = 0.5 + \sum_{i=j+1}^{D} \cos\left(\sum_{k=|2j-i|+1}^{i} x_k\right), j = 1, 2, ..., D-1$$

$$\phi_{n+j}(X) = -\phi_j(X), j = 1, 2, ...n$$

Here the objective is to minimize the module of the biggest among the samples of the so-called auto-correlation function which is related to the complex envelope of the compressed radar pulse at the optimal receiver output, while the variables represent symmetrized phase differences. According to [24], the above problem has no polynomial time solution.

D. RLTP-4: Transistor Design Modeling [32]

$$Minimize\ f(X) = \gamma^2 + \sum_{k=1}^{4}(\alpha_k^2 + \beta_k^2)$$

where

$$\alpha_k = (1 - x_1x_2)x_3\{\exp[x_5(g_{1K} - g_{3k}x_7 \times 10^{-3}$$
$$-g_{5k}x_8 \times 10^{-3})] - 1\}g_{5k} + g_{4k}x_2$$

$$\beta_k = (1 - x_1x_2)x_4\{\exp[x_6(g_{1k} - g_{2k} - g_{3k}x_7 \times 10^{-3}$$
$$+g_{4k}x_9 \times 10^{-3})] - 1\}g_{5k}x_1 + g_{4k}$$

$$\gamma = x_1x_3 - x_2x_4$$

subject to: $x_i \ge 0$

The given below matrix defines the numerical constants (g_{ik})

$$\begin{bmatrix} 0.485 & 0.752 & 0.869 & 0.982 \\ 0.369 & 1.254 & 0.703 & 1.455 \\ 5.2095 & 10.0677 & 22.9274 & 20.2153 \\ 23.3037 & 101.779 & 111.461 & 191.267 \\ 28.5132 & 111.8467 & 134.3884 & 211.4823 \end{bmatrix}$$

VI. Statistically Simulated Results

The comparative simulated statistical (Stat.) results for the 15 benchmark functions are presented in Table 3(A & B). The observed mean, median and SD for each function is taken for result comparison. Firstly NL-SFLA is able to solve all the 15 problems. The result from the Table 3(A & B) shows that the proposed variant performed at par with the state-of-art algorithms, especially when the results are compared with basic SFLA, DE, CA, GSA, and SaDE for all most all functions. For SPSO, GbABC, JADE, and jDE the evaluated results are at par. The bold faces show the best results. GbABC has better results for F5, F6, F8 and F12. SPSO presented better results for F11 and 13. JADE performed better for F1, F2, and F3 whereas better standard deviation is observed by NL-SFLA. For the function F4, F7 and F14 again NL-SFLA performed comparatively better than the state-of-art algorithms.

The statistical simulated results for the four real time problems are presented in Tables 4 – 8. For RLTP – 1 statistical results (mean, best, worst and SD) are presented in Table 1. The results are compared with DE, variant of DE i.e. MDE and basic SFLA. NL-SFLA and MDE able to evaluate same best fitness value however the SD for NL-SFLA was better. In other Table 2, parameter values of the problem are presented and compared with DE, basic SFLA and NSDE.

For the RLTP – 2, results are discussed and compared with DE, SFLA and MDE in Table 6. It is clearly observed that NL-SFLA achieved better results with significant SD.

Similarly, the results for RLTP – 3 and 4 are presented in Table 7 and 8. For RLTP – 3 results are compared with DE, MDE and basic SFLA. For the RLTP – 4 results are presented along with parametric values and compared with DE, SFLA and NSDE.

From the results it can be concluded that the proposed variant performed well on all four problems and achieved better or at par best fitness values. Also the observed SD is better for all problems, which reflects accuracy.

State of Art Algorithms	Experimental Settings
SFLA [33]	c = 1; le = 5
DE [28][34]	CR = 0.9; F = 0.5
SPSO [29][35]	w = 1/(2 * log(2)); Cl = 0.5
jDE [30][36]	CR = 0.9; F = 0.5
SaDE [30][36]	Not pre-specified
JADE [30][36]	c = 1/10; p = 0.05; Afactor = 1
GSA [31][37]	Elitist Check = 1; Rpower = 2; Rnorm = 2
CS [32][38]	beta = 1.5; pa = 0.25
GbABC [33][39]	SN = 12; C = 1.507; lf = 1.12

Table 2. State-of-art Algorithms taken for statistical comparisons

Problem(*F_i*)	**Stat.**	**SFLA**	**DE**	**SPSO**	**GbABC**	**GSA**
	Mean	1.09250E-21	2.87010E-74	0.00000E+00	5.06790E-16	1.47630E+04
F1	**Median**	4.44570E-23	1.36650E-87	0.00000E+00	5.55600E-16	1.51050E+04
	SD	2.90420E-21	1.54830E-73	0.00000E+00	1.28720E-16	3.04970E+03
	Mean	2.11750E-23	3.33330E+00	0.00000E+00	5.36270E-16	5.62480E-09
F2	**Median**	5.38800E-25	1.60790E-89	0.00000E+00	4.85900E-16	5.49970E-09
	SD	5.84650E-23	1.82570E+01	0.00000E+00	1.80140E-16	1.58610E-09
	Mean	3.2039E+00	3.59460E-42	1.15110E-50	4.6682E+03	4.5775E+04
F3	**Median**	2.8127E+00	4.23040E-45	1.06280E-51	4.6226E+03	4.2754E+04
	SD	1.4048E+00	1.68300E-41	3.42530E-50	1.8280E+03	1.6455E+04
	Mean	2.07940E-03	1.81270E-02	1.49360E-03	1.81270E-02	4.2480E-02
F4	**Median**	2.08950E-03	1.48510E-02	**1.20800E-03**	1.48510E-02	4.1547E-02
	SD	7.45060E-04	1.25680E-02	8.17800E-04	1.25680E-02	8.2225E-03
	Mean	2.4447E+01	7.0012E+01	4.4914E+01	**1.8957E-11**	3.0147E+01
F5	**Median**	2.3891E+01	6.9647E+01	3.9798E+01	**1.1369E-13**	3.1839E+01
	SD	5.3640E+00	1.9182E+01	2.1653E+01	**4.7447E-11**	6.8443E+00
	Mean	9.7550E-03	1.0721E+01	2.1516E+00	**5.6962E-14**	5.5644E-05
F6	**Median**	2.5160E-07	1.0620E+01	2.3168E+00	**5.6843E-14**	5.5620E-05
	SD	5.2082E-02	2.5977E+00	4.7042E-01	**1.1407E-14**	5.3296E-06
	Mean	3.9635E-02	3.9197E-01	7.7154E-03	1.4200E-02	5.9557E+02
F7	**Median**	2.7037E-02	1.0991E-01	7.3960E-03	7.6361E-03	5.9949E+02
	SD	3.7907E-02	8.0794E-01	7.9842E-03	1.7787E-02	8.2391E+01
	Mean	3.6924E+01	1.6965E+01	7.1336E+00	**6.5351E-01**	7.3253E+00
F8	**Median**	2.7230E+01	6.9737E+00	6.5930E+00	**2.9508E-01**	7.3361E+00
	SD	2.2323E+01	3.1387E+01	1.8580E+00	**9.9064E-01**	6.0670E-01
	Mean	3.4889E+01	4.0227E+00	7.2796E-01	**5.7739E-16**	1.5408E-01
F9	**Median**	6.8430E-17	1.2555E+00	5.1912E-01	5.5804E-16	4.6459E-11
	SD	1.9109E-02	6.0936E+00	8.8114E-01	2.2063E-16	2.8020E-01
	Mean	4.4492E+00	1.2421E+01	9.8602E+00	**2.2945E-16**	9.8496E-02
F10	**Median**	4.5651E+00	1.1552E+01	9.4740E+00	0.0000E+00	9.7545E-02
	SD	2.1069E+00	3.3714E+00	1.9998E+00	**7.8121E-16**	1.0691E-02
	Mean	3.2900E+00	7.3727E+00	**3.2166E-66**	2.8802E+02	5.7899E-08
F11	**Median**	3.0721E+00	5.0946E-62	**8.0527E-68**	2.9517E+02	5.5031E-08
	SD	1.2968E+00	1.8712E+01	**1.6273E-65**	5.8734E+01	1.8167E-08
	Mean	3.6686E-04	2.0081E-10	1.3919E+00	5.6660E-04	2.6247E-05
F12	**Median**	2.4658E-05	1.5465E-12	9.2397E-01	2.4409E-08	2.6567E-05
	SD	9.4918E-04	1.0415E-09	1.4960E+00	1.9155E-03	3.2402E-06
	Mean	4.0654E-01	2.6599E+00	**2.1321E-01**	1.2133E+00	1.2696E+01
F13	**Median**	3.9987E-01	2.5999E+00	**1.9987E-01**	1.1999E+00	1.2600E+01
	SD	7.8492E-02	1.2050E+00	**3.4575E-02**	1.9773E-01	1.3020E+00
	Mean	1.0126E+00	1.7596E+00	2.8661E+00	1.0126E+00	1.0000E+00
F14	**Median**	1.0000E+00	1.5919E+00	2.8628E+00	1.0000E+00	1.0000E+00
	SD	3.7334E-02	6.0045E-01	6.6970E-01	3.7334E-02	1.2410E-09
	Mean	1.4779E-02	1.5164E+00	6.1084E-01	**1.6502E-16**	4.9262E-03
F15	**Median**	**2.8497E-17**	1.4778E+00	5.9114E-01	9.0526E-17	6.9378E-08
	SD	5.9495E-02	4.5606E-01	2.5666E-01	**2.2903E-16**	2.6982E-02

Table 3(A). Statistical Comparison of results with state-of-art algorithms.

Problem(*F_i*)	**Stat.**	**CS**	**SaDE**	**JADE**	**jDE**	**NL-SFLA**
	Mean	2.53880E-38	1.18750E-101	**1.10710E-223**	3.49790E-212	1.5545E-209
F1	**Median**	3.63700E-39	1.34990E-115	**1.79720E-246**	1.87050E-221	3.7646E-211
	SD	6.55640E-38	6.44140E-101	0.00000E+00	0.00000E+00	**5.9742E-112**
	Mean	4.48950E-39	2.07670E-103	**3.30170E-221**	2.01120E-205	4.7861E-172
F2	**Median**	1.18410E-39	7.31530E-118	**1.34490E-234**	1.13300E-222	3.0911E-213
	SD	8.69760E-39	1.11700E-102	0.00000E+00	0.00000E+00	**2.0293E-121**
	Mean	2.92310E-05	8.83320E-02	**2.14460E-55**	1.39870E-11	2.6456E-47
F3	**Median**	1.42620E-05	1.70420E-04	**6.35760E-57**	2.28260E-12	3.7355E-51
	SD	3.62270E-05	4.49610E-01	**1.02700E-54**	3.47770E-11	1.9039E-44
	Mean	9.08690E-03	9.1211E-03	3.24140E-03	6.71590E-03	**1.2643E-03**
F4	**Median**	8.71540E-03	8.0039E-03	3.21450E-03	4.49160E-03	2.8231E-03
	SD	3.68140E-03	4.8538E-03	1.73980E-03	5.64550E-03	**6.0878E-04**

	Mean	1.1528E+01	3.3829E+00	1.3266E-01	1.4593E+00	1.2988E+00
F5	**Median**	1.2322E+01	3.9798E+00	0.0000E+00	9.9496E-01	1.1388E+00
	SD	3.5613E+00	1.4216E+00	3.4400E-01	1.4949E+00	2.3971E+00
	Mean	1.2417E-01	1.9051E+00	7.6297E-01	6.5272E-01	4.5213E-04
F6	**Median**	3.5527E-15	1.8978E+00	9.3130E-01	2.1316E-14	4.5433E-07
	SD	3.2199E-01	6.2596E-01	8.4242E-01	1.4677E+00	1.3841E-02
	Mean	1.0678E-03	7.1937E-02	7.6256E-03	1.2328E-02	**1.1276E-03**
F7	**Median**	0.0000E+00	3.9382E-02	0.0000E+00	1.4433E-15	**1.9336E-03**
	SD	**3.4040E-03**	1.5602E-01	1.2744E-02	1.9931E-02	4.0882E-03
	Mean	3.5345E+00	2.4080E+01	1.7272E+00	3.8558E+00	3.8919E+00
F8	**Median**	3.6292E+00	2.2900E+01	6.6871E-02	3.7318E+00	3.6789E+00
	SD	1.8282E+00	1.8179E+01	4.5235E+00	4.8032E+00	2.1900E+00
	Mean	7.5065E-30	8.2957E-02	8.9972E-02	4.8404E-02	2.1977E-02
F9	**Median**	**1.5705E-32**	2.0869E-32	1.5705E-32	1.6996E-32	1.8663E-02
	SD	**2.2671E-29**	1.6204E-01	2.8058E-01	1.3820E-01	2.6788E-02
	Mean	1.6957E-01	7.3606E-01	5.2637E-01	1.3573E-01	2.6423E-02
F10	**Median**	6.7130E-02	6.5443E-01	2.3098E-01	**7.4871E-03**	2.2655E-02
	SD	2.5386E-01	5.7344E-01	7.2053E-01	2.5827E-01	1.5454E-02
	Mean	6.4823E-07	6.3595E-02	2.8179E-09	2.1659E-19	8.5433E-10
F11	**Median**	4.3043E-07	1.9753E-05	7.0518E-63	1.4699E-21	7.2387E-10
	SD	7.0728E-07	3.3502E-01	1.5435E-08	9.2580E-19	1.0848E-10
	Mean	6.6568E-01	3.4446E-16	4.7184E-16	**5.6321E-17**	1.0933E-10
F12	**Median**	6.9049E-01	2.2204E-16	2.2204E-16	**1.2014E-25**	4.9837E-12
	SD	6.2163E-01	4.1040E-16	7.3741E-16	**1.8760E-16**	1.7837E-14
	Mean	3.2990E-01	5.7654E-01	5.1987E-01	3.9321E-01	3.1623E-01
F13	**Median**	2.9987E-01	5.9987E-01	4.9987E-01	2.9987E-01	2.3636E-01
	SD	5.9585E-02	1.7555E-01	1.9191E-01	2.2733E-01	1.9786E-01
	Mean	1.0480E+00	1.0000E+00	1.0000E+00	1.0000E+00	**1.0000E+00**
F14	**Median**	1.0474E+00	1.0000E+00	1.0000E+00	1.0000E+00	**1.0000E+00**
	SD	1.0279E-02	0.0000E+00	1.0286E-07	1.2370E-16	**5.8374E-17**
	Mean	4.9261E-03	1.0838E-01	2.6109E-01	5.4188E-02	1.2334E-02
F15	**Median**	2.0670E-04	1.4778E-01	1.4778E-01	6.8282E-02	4.0838E-02
	SD	2.6982E-02	1.1600E-01	2.2524E-01	1.1304E-01	2.0018E-01

Table 3(B). Statistical Comparison of results with state-of-art algorithms.

Stat.	**DE**	**SFLA**	**MDE**	**NL-SFLA**
Best	4.21417	4.21412	4.21421	4.21421
Mean	4.20422	4.21421	4.21418	4.21422
Worst	4.21312	4.21514	4.21418	4.21431
SD(σ)	0.01056	0.01105	0.00504	0.00429

Table 4. Statistical Comparison of *RLTP* – 1 results.

Stat.	**DE**	**SFLA**	**NSDE**	**NL-SFLA**
x_1	1.00008	1.00019	1.0015	1.00149
x_2	4.99993	4.999981	4.99998	4.999929
x_3	1.49979	1.50008	1.50006	1.50002
x_4	4.79993	4.79999	4.79999	4.79999
x_5	-2.00031	-2.000045	-2.00004	-2.000037
x_6	-4.90005	-4.900034	-4.90003	-4.900032
$F(x)$	4.700E-06	2.985E-06	2.341E-06	2.340E-06

Table 5. Parametric values of *RLTP* – 1 for DE, SFLA, NSDE and NLSFLA

Stat.	**DE**	**SFLA**	**MDE**	**NL-SFLA**
Best	15.2875	15.2912	14.4301	8.87449
Mean	17.2614	17.2891	15.735	13.714
Worst	19.3476	20.011	16.4621	12.9581
SD(σ)	4.55E-02	4.69E-02	5.53E-03	6.09E-03

Table 6. Statistical Comparison of *RLTP* – 2 results.

Stat.	**DE**	**SFLA**	**MDE**	**NL-SFLA**

Best	0.6745	0.6751	0.6625	0.6625
Mean	0.7952	0.8125	0.7523	0.7496
Worst	0.9009	0.9122	0.8551	0.8966
SD(σ)	8.22E-02	8.12E-02	4.66E-02	2.23E-03

Table 7. Statistical Comparison of *RLTP* – 3 results.

Stat.	**DE**	**SFLA**	**NSDE**	**NL-SFLA**
x_1	0.901341	0.901342	0.901336	0.9013359
x_2	0.891174	0.891176	0.891053	0.8910531
x_3	3.87757	3.87867	3.87933	3.879329
x_4	3.94642	3.94652	3.94662	3.94662
x_5	5.32623	5.32625	5.32509	5.32511
x_6	10.6239	10.6241	10.6162	10.6161
x_7	0.0000	0.0000	0.0000	0.0000
x_8	1.08914	1.089214	1.08881	1.0889
x_9	0.705575	0.706105	0.706727	0.706726
$F(x)$	0.054371	0.054372	0.054366	0.0543679

Table 8. Parametric values of *RLTP* – 4 for DE, SFLA, NSDE and NLSFLA

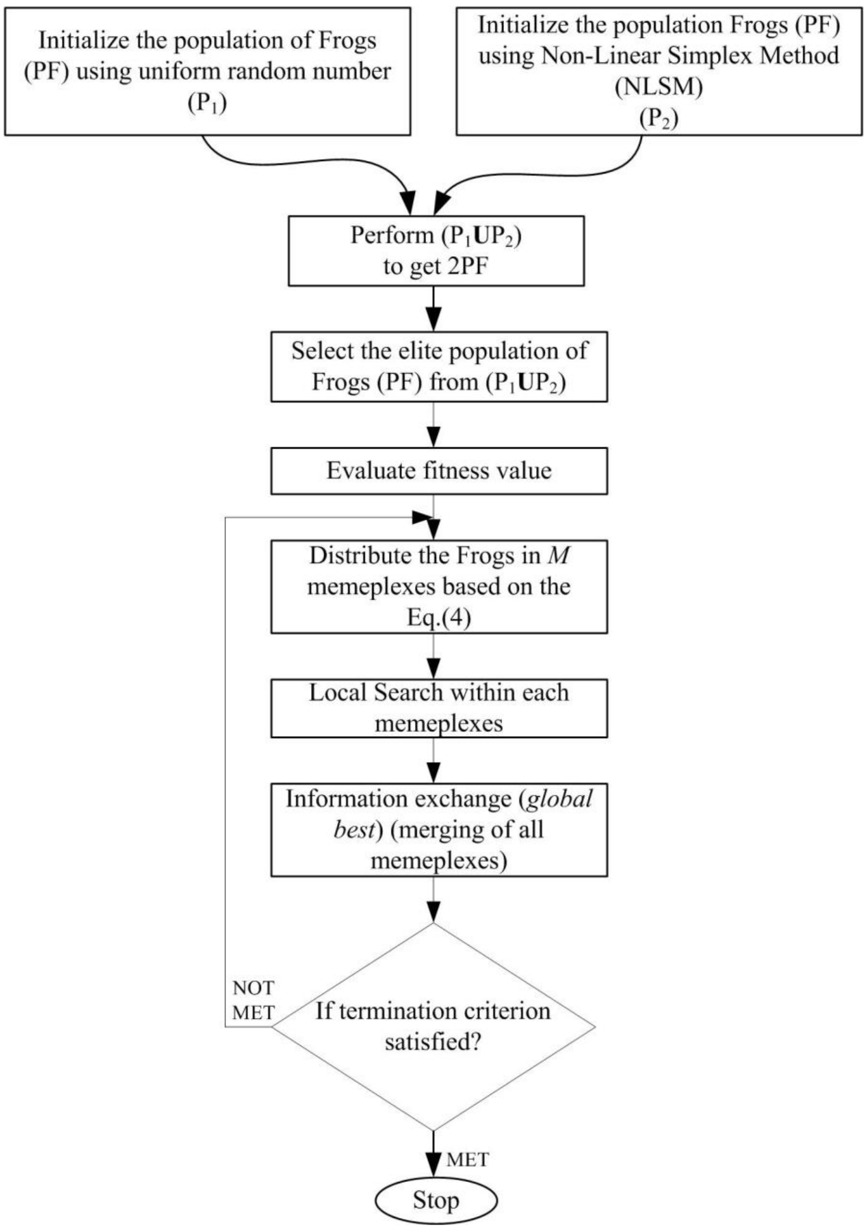

Initialize the population of Frogs (PF) using uniform random number (P$_1$)
Initialize the population Frogs (PF) using Non-Linear Simplex Method (NLSM) (P$_2$)
Perform (P$_1$UP$_2$) to get 2PF
Select the elite population of Frogs (PF) from (P$_1$UP$_2$)
Evaluate fitness value
Distribute the Frogs in M memeplexes based on the Eq.(4)
Local Search within each memeplexes
Information exchange (global best) (merging of all memeplexes)
If termination criterion satisfied?
NOT MET
MET
Stop

Figure 7. A schematic of proposed NL-SFLA

VII. Conclusions and future scope

Present study is focused on the population diversity and handling the continuous optimization problems by Shuffled Frog Leaping Algorithm. Two modifications are done to in the basic structure of SFLA. The population of virtual frogs is generated using nonlinear simplex method of Nelder and Mead along with the traditional method of initializing population i.e. random. Then elite population is taken as initial solution. Secondly, the distribution of frogs in different memeplexes is modified to handle continuous optimization problems. The variant embedded with two modifications is named as NL-SFLA. This variant is applied to solve 15 benchmark problems (unimodal and multimodal) and four real life time optimization problems. The statistically simulated results present the efficiency and accuracy to solve the problems.

In future an attempt would be made to enhance the application area of the proposed variant of SFLA as it performed will on continuous optimization problems as well as on constrained problems

Acknowledgment

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

References

[1] J Rajpurohit, T.K. Sharma, A. Abraham, Vaishali. “Glossary of Metaheuristic Algorithms”, *International Journal of Computer Information Systems and Industrial Management Applications*, 9(2017), pp. 181 – 205, 2017.

[2] E. Muzaffar M., and K. E. Lansey “Optimization of water distribution network design using the shuffled frog leaping algorithm”, *Journal of Water Resources Planning and Management*, 129(3), pp. 210-225, 2003.

[3] J, Cai, R. Zhou, D. Lei. “Dynamic shuffled frog-leaping algorithm for distributed hybrid flow shop scheduling with multiprocessor tasks”, *Engineering Applications of Artificial Intelligence*, 90, pp. 103540, 2020

[4] R. Dash, R. Dash, R. Rautray. “An evolutionary framework based microarray gene selection and classification approach using binary shuffled frog leaping algorithm”, *Journal of King Saud University - Computer and Information Sciences*, 2020 https://doi.org/10.1016/j.jksuci.2019.04.002

[5] Y. Guo, X. Tian, G. Fang, Yue-Ping Xu. “Many-objective optimization with improved shuffled frog leaping algorithm for inter-basin water transfers”, *Advances in Water Resources*, 138 Article 103531, 2020

[6] T.K. Sharma, J. Rajpurohit, D. Prakash. “Enhanced Local Search in Shuffled Frog Leaping Algorithm”, In *Proceedings of Soft Computing: Theories and Applications*, 1053. Springer, Singapore, pp. 1441-1448, 2020.

[7] Q. Huang, W. Song. “A land-use spatial optimum allocation model coupling a multi-agent system with the shuffled frog leaping algorithm”, *Computers, Environment and Urban Systems*, 77, Article 101360, 2019.

[8] H. Moayedi, D. T. Bui, Phuong T. T. N.. “Shuffled Frog Leaping Algorithm and Wind-Driven Optimization Technique Modified with Multilayer Perceptron”, *Appl. Sci.*, 10(2), pp. 689, 2020.

[9] J. Tang, R. Zhang, P. Wang, Z. Zhao, L. Fan, X. Liu. “A discrete shuffled frog-leaping algorithm to identify influential nodes for influence maximization in social networks”, *Knowledge-Based Systems*, 187, 104833, 2020

[10] Y. Li, Z. Yan. “Improved shuffled frog leaping algorithm on system reliability analysis”, *Brain Inform.*, 6(1), pp. 1-7, 2019.

[11] Q. Huang, W. Song. “A land-use spatial optimum allocation model coupling a multi-agent system with the shuffled frog leaping algorithm”, *Computers, Environment and Urban Systems*, 77, Article 101360, 2019.

[12] E. E. Elattar. “Environmental economic dispatch with heat optimization in the presence of renewable energy based on modified shuffle frog leaping algorithm”, *Energy*, 17115, pp. 256-269, 2019.

[13] T.K. Sharma, M. Pant. “Shuffled artificial bee colony algorithm,” *Soft Computing*, 21, 6085–6104, 2017. https://doi.org/10.1007/s00500-016-2166-2

[14] J.A. Nelder, R. Mead. “A simplex method for function minimization”, *Computer Journal*, 7, pp. 308–13, 1965.

[15] S. Janani, D. Ramyachitra, R. Ranjani Rani. “PCD-DPPI: Protein complex detection from dynamic PPI using shuffled frog-leaping algorithm”, *Gene Reports*, 12, pp. 89-98, 2018

[16] S. P. Rajamohana, K. Umamaheswari. “Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection”, *Computers & Electrical Engineering*, 67, pp. 497-508, 2018.

[17] R. Dash. “An improved shuffled frog leaping algorithm based evolutionary framework for currency exchange rate prediction”, *Physica A: Statistical Mechanics and its Applications*, 48615, pp. 782-796, 2017.

[18] P. Kaur, S. Mehta. “Resource provisioning and work flow scheduling in clouds using augmented Shuffled Frog Leaping Algorithm”, *Journal of Parallel and Distributed Computing*, 101, pp. 41-50, 2017.

[19] T. K. Sharma, M. Pant. “Opposition based learning ingrained shuffled frog-leaping algorithm”, *Journal of Computational Science*, 21, pp. 307-315, 2017

[20] D. Tang, J. Yang, S. Dong, Z. Liu. “A lévy flight-based shuffled frog-leaping algorithm and its applications for continuous optimization problems”, *Applied Soft Computing*, 49, pp. 641-662, 2016.

[21] A. M. Dalavi, P. J. Pawar, T. P. Singh. “Tool path planning of hole-making operations in ejector plate of injection mould using modified shuffled frog leaping algorithm”, *Journal of Computational Design and Engineering*, 3(3), pp. 266-273, 2016.

[22] B. Tripathy, S. Dash, S. K. Padhy. “Multiprocessor scheduling and neural network training methods using shuffled frog-leaping algorithm”, *Computers & Industrial Engineering*, 80, pp. 154-158, 2015.

[23] M. Jadidoleslam, A. Ebrahimi. “Reliability constrained generation expansion planning by a modified shuffled frog leaping algorithm”, *International Journal of Electrical Power & Energy Systems*, 64, pp. 743-751, 2015.

[24] S. Sharma, T. K. Sharma, M. Pant, J.Rajpurohit, B. Naruka. "Centroid Mutation Embedded Shuffled Frog-Leaping Algorithm", *Procedia Computer Science*, 46, pp. 127-134, 2015.
[25] K. K. Bhattacharjee, S. P. Sarmah. "Shuffled frog leaping algorithm and its application to 0/1 knapsack problem", *Applied Soft Computing*, 19, pp. 252-263, 2014.
[26] J. Vijaya Kumar, D. M. Vinod Kumar. "Generation bidding strategy in a pool based electricity market using Shuffled Frog Leaping Algorithm", *Applied Soft Computing*, 21, pp. 407-414, 2014.
[27] A. Sarkheyli, A.M. Zain, S. Sharif. "The role of basic, modified and hybrid shuffled frog leaping algorithm on optimization problems: a review", *Soft Comput.*, 19, pp. 2011–2038, 2015.
[28] H. Pu, Z. Zhen, D. Wang. "Modified shuffled frog leaping algorithm for optimization of UAV flight controller", *International Journal of Intelligent Computing and Cybernetics*, 4(1) pp. 25 -39, 2011.
[29] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. -P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore And KanGAL Report Number 2005005 (Kanpur Genetic Algorithms Laboratory, IIT Kanpur) , 2005.
[30] S. Das, A. Abraham, U. Chakraborty and A. Konar. "Differential Evolution Using a Neighborhood Based Mutation Operator," *IEEE Transaction of Evolutionary Computing*, 13(3), pp. 526–553, 2009.
[31] A.E. Dor, M. Clerc and P. Siarry, "—*Hybridization of Differential Evolution and Particle Swarm Optimization in a New Algorithm: DEPSO-2S*", In Proceeding of SIDE 2012 and EC 2012, LNCS 7269, Springer-Verlag Berlin Heidelberg, pp. 57 – 65, 2012.
[32] M. Ali, M. Pant, A. Abraham. "A simplex differential evolution algorithm: development and applications," *Transactions of the Institute of Measurement and Control*, 34(6), pp. 691–704, 2011.
[33] D. Tang, J. Yang, S. Dong, Z. Liu. "A lévy flight-based shuffled frog-leaping algorithm and its applications for continuous optimization problems," *Applied Soft Computing*, 49, pp. 641-662, 2016.
[34] N. Veček , M. Mernik, M. Črepinšek. "A chess rating system for evolutionary algorithms: A new method for the comparison and ranking of evolutionary algorithms," *Information Sciences*, 277, pp. 656-679, 2014.
[35] M.G.H. Omran, M. Clerc, 2011. <http://www.particleswarm.info/> (accessed 15.10.12)
[36] Q. Zhang, 2011. <http://dces.essex.ac.uk/staff/qzhang/> (accessed 6.10.13).
[37] E. Rashedi, H. Nezamabadi-pour, Saeid Saryazdi. "GSA: A Gravitational Search Algorithm," *Information Sciences*, 179, pp. 2232-2248, 2009.
[38] X. Yang, S. Deb. "*Cuckoo search via Levy flights*," In *Proceedings of World Congress on Nature & Biologically Inspired Computing* USA, 210-214, 2009.
[39] T. Liao, D. Aydın, T. Stützle, "Artificial bee colonies for continuous optimization: Experimental analysis and improvements," *Swarm Intelligence*, 7(4), pp. 327-356, 2013

Author Biographies

Tarun K. Sharma presently associated with Department of Computer Science and Engineering, Graphic Era Hill University, Dehradun, Uttarakhand, India. He did his PhD in Computational Intelligence from Indian Institute of Technology (IIT) Roorkee, India. His research interest includes designing hybrid variants of nature inspired algorithms and their engineering applications. He has published several research papers in peer reviewed International Journals and Conferences of repute. He has also edited several books in Springer. He served in Editorial board and reviewers of several International Journals. He is also the founding members of International Conference on Soft Computing: Theories and Applications (SoCTA) Series

Prof. Ajith Abraham received his Ph.D. degree in Computer Science from Monash University, Melbourne, Australia. He is currently coordinating the efforts of Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, USA, which has members from more than 100+ countries. He has a worldwide academic experience with formal appointments in several Universities in Asia, Australia, Europe and the US. He serves/has served the editorial board of over 50 International journals and has also guest edited 50 special issues on various topics related to machine intelligence. He is a co-author of more than 1000+ research publications, and some of the works have also won best paper awards at international conferences. He is also the founder member of several International Conferences of repute.