International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-11, September 2019

Modified Shuffled Frog Leaping Algorithm by adaptive Step Size: Applications to Constraint Engineering Design Problems

Bhagyashri Naruka, Ashwani Kumar Yadav, Vaishali, Shweta Sharma, Janesh Singh Rathore

Abstract*: Shuffled frog leaping algorithm (SFLA) is an ongoing expansion to the group of evolutionary algorithm that imitates the societal and natural conduct of species. Upsides of particle swarm optimization (PSO) and shuffled complex evolution (SCE) is consolidates in SFLA i.e. local searching and information shuffling respectively. In this paper SFLA is improved to solve equality and inequality based constraint engineering design problems using penalty function. In proposed approach linear decreasing function that is adaptive in nature will be utilized to improve worst frog position for better exploration and convergence speed. The simulation results designate the superiority of present study over SFLA in term of global optimum solution and fast convergence rate

Index Terms – Shuffled Frog leaping Algorithm (SFLA), Linear Decreasing function, Constraint Engineering Design Problems, Constraint Handling, Penalty Function.

I. INTRODUCTION

Optimization is the procedure of selecting the best or alternative solution of the objective functions depending on certain constraints. For several problems, proficient methodically (traditional) based algorithms exist, for example linear programming, Divide and conquer, for acquiring global optimum solutions. Traditional algorithm is good for small scale problems but with discrete, large scale and combinatorial optimization problems objective function value and complexity of algorithm increases therefore algorithm takes much time to give optimum solution [1]. To overcome the limitations of the traditional method heuristic algorithms are come in existence. Advantages of heuristic algorithms over traditional ones are as follows:

1) Implementation of heuristic algorithms is usually easy.
2) They do not require any auxiliary information regarding problem province.
3) Computational time and cost is generally less as compared to traditional algorithms.
4) Generally they offer a list of good solution instead of a single solution.

Manuscript published on 30 September 2019.
*****Correspondence Author(s)
Bhagyashri Naruka, Amity School of Engineering and Technology, Amity University Rajasthan, Jaipur, India,bhagyashree.naruka@gmail.com.
Ashwani Kumar Yadav, Amity School of Engineering and Technology, Amity University Rajasthan, Jaipur, India, akyadav@jpr.amity.edu.
Vaishali, School of Computing and Information Technology**,** Manipal University Jaipur, India,vaishaliyadav26@gmail.com.
Shweta Sharma, Amity School of Engineering and Technology, Amity University Rajasthan, Jaipur, India, shweta_sharma0287@yahoo.com.
Janesh Singh Rathore, Body Engineering, Hero Motocorp, Jaipur, India, janesh1989@gmail.com

Heuristic techniques are nature inspired algorithms. Some of the well known algorithms are Genetic Algorithms (GA); multidimensional optimization global search algorithm that is inspired by Darwin's natural selection theory. [2], Differential Evolution (DE) based on the crossover and selection operators similar to GA [3], Ant colony optimization (ACO) work on the concept of food hunting behavior of ant colonies [4], particle swarm optimization (PSO) based on the perception of social and natural behavior of the species [5], Shuffled Frog Leaping Algorithm (SFLA) [6] and a lot more in the rundown. These algorithms proved their efficiency in all most every problem domain [7-11].

In present study, Shuffled Frog leaping Algorithm, introduced by Eusuff and Lansey is used to solve constraint engineering design problem [6]. SFLA based on the conception of foraging behavior of the group of frogs. . Upsides of particle swarm optimization (PSO) and shuffled complex evolution (SCE) is consolidates in SFLA i.e. local searching and information shuffling respectively. Like other heuristic algorithms, conventional SFLA having restrictions with respect to convergence speed. In proposed approach, linear decreasing function is introduced for frog leaping for fast convergence rate.

The remainder of the paper is structured in seven sections including introduction followed by an overview of SFLA algorithm in section 2. Section 3 depicts the constraint handling using penalty function. Section 4 describes the proposal. Section 5 explains EDP's and section 6 gives experimental setup details and analyze the simulated results followed by section 7, which concludes the paper.

II. SFLA OVERVIEW

SFLA is a population based memetic meta-heuristic algorithm. It is based on the notion of food hunting process of frog groups (memeplex). The SFLA is an amalgamation of indiscriminate and deterministic approaches. Deterministic approach enables the SFLA to investigate the information of response surface efficiently to conduct the search as in PSO. Flexibility of the search process is ensured by indiscriminate approach. At the outset, population of frog is initialized randomly that covers the entire swamp. Frog population is sort according to the decreasing fitness value. Now, divide the frogs into subsets entitled as memeplexes. Various subsets owning frogs from diverse ethnicity and every frog complete a neighborhood search to improve the position of worst frog.

At the point when every subset evolves with preset number of memeplex iteration, the thoughts own by the frogs inside the subset are gone among subsets using shuffling procedure. This procedure of neighborhood search and information shuffling carry on until the stopping criteria reach.

SFLA working steps are as follows:

A) Initialization

Choose n and m, where n stands for memeplex size and m represents number of memeplexes. So, the total population size P_f is given by $P_f = mn$.

B) Generate Population

Generate population of frogs $X(1), X(2), \ldots . X(P_f)$in the doable space $\omega \subset \Re^d$,where d stands for decision variables. In population, $X(i) = (X_i^1, X_i^2, \ldots \ldots, X_i^d$ represents the i^{th} frog. The fitness value $f(i)$ is calculated for every frog $X(i)$.

C) Frog ranking

Arrange the P_f frogs so as to diminishing fitness value. Store the sorted value of frogs in an array $Y = \{X(i), f(i), i = 1 \ldots P_f\}$, thus $i = 1$ speaks to the frog owning best value of fitness. Record the position of best frog's X_g in the intact population P_f.

D) Divide frogs into memeplexes

Array Y is divided into m memeplexes $U^1, U^2, \ldots . U^m$, and n represents the frog numbers in every memeplex. For $m = 4$, first frog of array Y is send into memeplex 1, next frog is send into memeplex 2, m^{th} frog assign to memeplex m and $m + 1^{th}$ frog send into memeplex 1,process continues until all the frog is divided into memeplexes.

E) Memeplex Evolution (local search)

Worst frog X_w and Best frog X_b position is identified in every memeplex, and worst frog position is updated through equation (1) to find the better food source.

$$D_s = rand(0,1) * (X_b - X_w) \quad (1)$$

$$X_{wnew} = X_w + D_s, \quad -D_{max} \leq D \leq D_{max}\ , \quad (2)$$

Where s denotes the s^{th} memeplex with range $1 <= s < = m$; $rand(0,1)$ is uniformly distributed random number generator between 0 and 1; Ds is leaping value (step size) through which worst frog move toward best solution;

X_{wnew} represents the updated worst frog value; D_{max} shows maximum step size allowed.

If the fitness value of X_{wnew} is better than worst frog fitness value X_w , X_w is replaced by X_{wnew} otherwise equation (3) is used to improve the X_w.

$$D_s = rand\ (0,1) * (X_g - X_w). \quad (3)$$

In equation (3), instead of using local best X_b ; global best fitness value X_g is used. Now again compare the X_w and X_{wnew} and if X_{wnew} shows no improvement in fitness value; X_w is replaced by uniformly distributed random number between 0 and 1

F) Shuffle memeplexes

After certain number of memeplex evolution, frogs of every memeplex are merged together according to the descending fitness value. Steps A to F repeated until the convergence criterion met.

III. CONSTRAINT HANDLING USING PENALTY FUNCTION

To solve constraint engineering design problems, constraint handling is very imperative and key concern. It is very hard to find feasible solution for optimization problems with existence of equality and inequality constraints. Many constraint handling methods have been invented in the most recent decades. Penalty function is the widely used technique to solve constraint problems. Penalty functions, regardless of their prevalence, have some confinements like there are numerous parameters to be balanced and every time it does not give satisfactory results. Identifying parameters value according to the constraint problem and penalty function is too tough task. To overcome all the limitations, Deb [12] improves penalty function by giving the idea of penalty function without parameters, i.e., one need to solve unconstraint problem in search area S using improved objective function given as

$$F(x) = \begin{cases} f(x) & if\ x \in S \\ f_w + \sum_{z=1}^{p+q} g_z(x) & if\ x \notin S, \end{cases} \quad (4)$$

Where x represents the solution acquired by approaches, f_w represents the worst feasible solution in the population, p stands for equality constraints, q stands for inequality constraints, S shows feasible solutions search space, and g_z used to represent set of constraints.

IV. PROPOSED SFLA

SFLA, in spite of having outstanding features, it is at times condemned due to measured convergence rate and getting caught in local minimum for some engineering design problems which are computationally expensive. Each frog perform local searching process in its memeplex and find locally best solution and then shuffle this information with other memeplexs and again partitioned into memeplexes. This information shuffling process leads to optimal solution. From equation (1) it is depicted that as the position difference of best and worst frog decreases, perturbation decreases proportionally at worst frog position. Therefore, search procedure may guide to premature convergence. To keep away from such occurrence, local searching method is modified in present study. In Proposed approach linear decreasing function that is adaptive in nature will be utilized to improve worst frog position for better exploration and convergence speed. Worst frog searching mechanism is speed up using the concept of scaling factor to improve worst frog position. Modified equation is as follows:

$$D_k = rand\ (0,1) * SF(X_b - X_w), \quad (5)$$

SF denotes scaling factor; used to amplify the exploration of $(X_b - X_w)$. SF is computed using linear decreasing function and presented in equation (6)

$$SF = 2 - it \times (1/beta). \quad (6)$$

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Where, $beta$ = Maximum number of memeplex iterations; it= 1 to $beta$.

By this SF value, function is linearly decreases between 1 and 2. According to the literature, step sizes between 1 and 2 for frog leaping gives best results. [43].

V. CONSTRAINT ENGINEERING DESIGN PROBLEMS

A. Pressure Vessel Problem (PVP)

The main intent of PVP is to reduce the overall cost of cylindrical vessel by taking into consideration its four decision variables name as "pressure vessel thickness (T_s)", "head thickness (T_h)", "vessel's inner radius (R)", and "vessel length without heads (L)" as shown in figure 1. It is constraint EDP having three linear and one non-linear inequality constraint. Detailed discussion of the problem can be read from Ref. [13]. The optimized mathematical equations for PVP are specified as follows:

$$Minimize\ f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

$$s.t.\ g_1(x) = -x_1 + 0.0193x_3 \le 0$$

$$g_2(x) = -x_2 + 0.00954x_3 \le 0$$

$$g_3(x) = -\pi x_3^2x_4 - \tfrac{4\pi}{3}x_3^3 + 1296000 \le 0$$

$$g_4(x) = -x_4 - 240 \le 0$$

Two different variants of this problem are defined in literature as follows:

Variant 1: Variables bounds are as given below

$$0.0625 \le x_1\ ;\ x_2 \le 99 \times 0.0625\ ;$$

$$10 \le x_3\ ;\ x_4 \le 200$$

Variant 2: In variant 1, $g_4(x)$ constraint is automatically satisfied if bound $x_4 \le 200$ is used. Therefore, for more convoluted way, upper limit of x_4 is extended by many researchers [14, 15, 16, 17] i.e., $10 \le x_4 \le 200$. Thus, under this variant 2, variables bound are

$$0.0625 \le x_1\ ;\ x_2 \le 99 \times 0.0625\ ;$$

$$10 \le x_3 \le 200;\ 10 \le x_4 \le 240$$

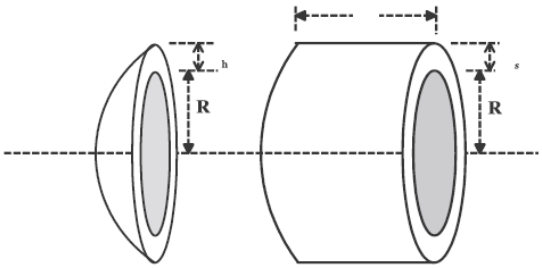


Figure 1. Pressure Vessel

B. Tension String Design Problem (TP)

The objective of this design problem is to minimize weight of the Compression/Tension String by taking into consideration its four decision variables name as wire diameter, coil diameter and active coil numbers as shown in figure 2. It is constraint EDP having one linear and three non-linear inequality constraints. Detailed discussion of the problem can be referred from Ref. [13]. Mathematical model for TP with decision variables $= [d, D, N] = [x_1, x_2, x_3]$, is depicted as follows:

$$Minimize\ f(x) = (x_3 + 2)x_2x_1^2$$

$$s.t.\ g_1(x) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0$$

$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \le 0$$

$$g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0$$

$$g_4(x) = \frac{x_2 + x_1}{1.5} - 1 \le 0$$

Variables Upper and lower bounds are as given below

$$0.05 \le x_1 \le 2; 0.25 \le x_2 \le 1.3; 2 \le x_3 \le 15$$

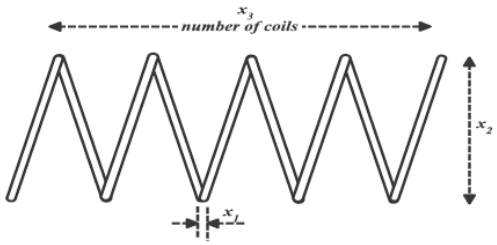


Figure 2. Tension String Design Problem

VI. EXPERIMENTAL SETUP AND EDP'S RESULTS

For the present learning; MATLAB is used for the simulation with following SFLA parameters:

- Frog Population = 50
- Memeplex Size = 10
- Number of Memeplexes = 5
- Beta = Maximum number of memeplex iterations
- D_{max} = 100% variable range

Proposed approach is repeated 50 times to remove stochastic discrepancy and the best value should opt as a response.

PC configuration for the experiment execution is as follows – Windows 7 (2009); processor- Intel(R) Core(TM) i3 CPU M50@2.27GHz; RAM 2.99GB; 32-bit operating system.

Results acquired by present study for two variants of pressure vessel problem and Tension String Problem are compared with the results of literature [13,14,15,17,19,20,22,23,24,25,26,27,29,30,31,32,33,34,36, 38] and shown in table 1, 2, and 3 respectively. Tables give the clear picture that present study performs better than existing ones. Possible improvement (PI) on the existing literature is also revealed in the table.

Graphical comparison of SFLA and proposed SFLA is revealed in figure 3. It is depicted from the figure 3(a-c) that proposed SFLA provides better results than SFLA.

VII. CONCLUSION

The proposed work that is a variant of SFLA, gives a systematic and comprehensive methodology for optimization of Constraint Engineering Design Problems. In the present learning, step size of frog is improved using linear decreasing function. Intend of the proposed approach is to expedite the convergence speed and improve local searching process of basic SFLA. Efficiency of the present study is analyzed by testing it on constraint EDPs. Computational results of EDPs depicts that proposed algorithm provide more reliable results.

Table 1. Comparison of Pressure Vessel variant 1 with other methods of problem solution

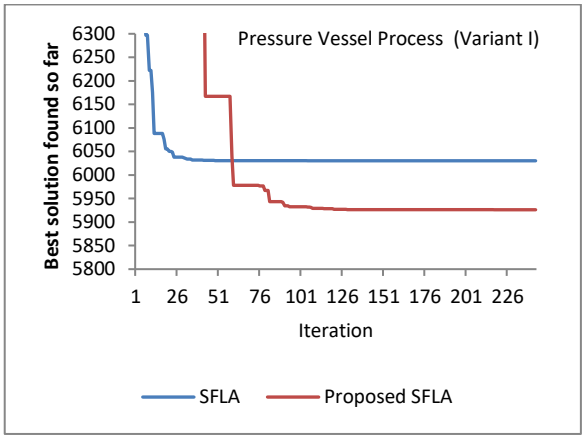
	Kannan and Kramer [28]	Sandgren [24]	Kaveh and Talatahari [15]	He and Wang [29]	Coello [38]	Montes and Coello [30]	Kaveh and Talatahari [32]	Akay and Karaboga [19]	Cagnina et al. [13]	Coelho [20]	He et al. [33]	SFLA(Present study)	Proposed Approach (Present Study)
x_1	1.125000	1.125000	0.812500	0.812500	0.812500	0.812500	0.812500	0.812500	0.812500	0.812500	0.812500	0.8551006	0.8013307
x_2	0.625000	0.625000	0.437500	0.437500	0.437500	0.437500	0.437500	0.437500	0.437500	0.437500	0.437500	0.4226767	0.3960982
x_3	58.291000	47.700000	42.103566	42.091266	40.323900	42.098087	42.098353	42.098446	42.098445	42.098400	42.098445	44.30573	41.51973
x_4	43.690000	117.7010 00	176.573220	176.746500	200.0000	176.640518	176.637751	176.636596	176.636 5950	176.637200	176.6365950	151.0785	183.9423
$f(X)$	7198.0428	8129.1036	6059.0925	6061.0777	6288.7445	6059.7456	6059.7258	6059.714339	6059.714335	6059.7208	6059.7143	6030.264	5926.1145
PI	17.67047%	27.10002%	2.19468%	2.22671%	5.76633%	2.25495%	2.20490%	2.20472%	2.20472%	2.20482%	2.20472%	1.72711%	

Table 2. Comparison of Pressure Vessel variant 2 with other methods of problem solution

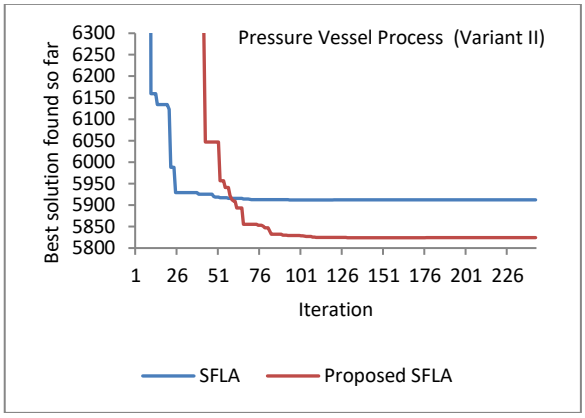
	Dimopoulos [27]	Gandomi et al. [31]	Mahdavi et al. [17]	SFLA(Present study)	Proposed Approach (Present Study)
x_1	0.75	0.75	0.75	0.7936228	0.7407105
x_2	0.375	0.375	0.375	0.3922882	0.3661336
x_3	38.86010	38.86010	38.86010	41.12035	38.37878
x_4	221.36549	221.36547	221.36553	189.1457	228.9023
$f(X)$	5850.38306	5850.38306	5849.76169	5912.2771	5824.337
PI	0.44520%	0.44520%	0.43462%	1.48741%	

Table 3. Comparison of Tension String Design problem with other methods of problem solution

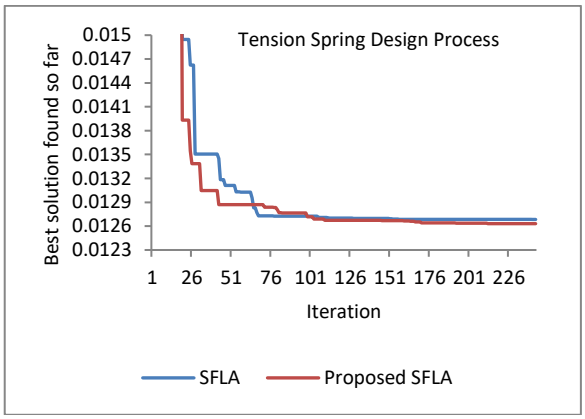
	Belegundu [23]	Coello and Montes [22]	Coello [38]	Ray and Liew [26]	Ray and Saini [25]	Tsai [14]	He and Wang [29]	Mahdavi et al. [17]	Raj et al. [36]	Montes et al. [34]	Cagnina et al. [13]	Montes and Coello [30]	SFLA(Present study)	Proposed Approach (Present Study)
x_1	0.05	0.051989	0.051480	0.0521602170	0.050417	0.05168906	0.051728	0.05115438	0.05386200	0.051688	0.051583	0.051643	0.05	0.0516
x_2	0.315900	0.363965	0.351661	0.368158695	0.321532	0.3567178	0.357644	0.34987116	0.41128365	0.356692	0.354190	0.355360	0.317961	0.355236
x_3	14.25000	10.890522	11.632201	10.6484422590	13.979915	11.28896	11.244543	12.0764321	8.68437980	11.290483	11.438675	11.397926	13.9571	11.3523
$f(X)$	0.0128334	0.0126810	0.01270478	0.01266924934	0.013060	0.01266523	0.0126747	0.0126706	0.0127484840	0.012665	0.012665	0.012698	0.012684	0.012629
PI	1.59271%	0.41006%	0.59646%	0.31769%	3.30015%	0.28605%	0.36056%	0.32831%	0.93658%	0.02842%	0.28424%	0.54339%	0.43617%	



a. Variant 1 Pressure Vessel problem



b. Variant II Pressure Vessel problem



c. Tension Spring Design problem

Figure 3. Comparison graph of SFLA and proposed SFLA

REFERENCES

1. Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2012). Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information sciences*, *183*(1), 1-15.
2. Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, Reading, MA. *SUMMARY THE APPLICATIONS OF GA-GENETIC ALGORITHM FOR DEALING WITH SOME OPTIMAL CALCULATIONS IN ECONOMICS*.
3. Neri, F., & Tirronen, V. (2010). Recent advances in differential evolution: a survey and experimental analysis. *Artificial Intelligence Review*, *33*(1-2), 61-106.
4. He, J., & Hou, Z. (2012). Ant colony algorithm for traffic signal timing optimization. *Advances in Engineering Software*, *43*(1), 14-18.
5. Kennedy, R. (1995, November). J. and Eberhart, Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks IV, pages* (Vol. 1000, p. 33).
6. Eusuff, M. M., & Lansey, K. E. (2003). Optimization of water distribution network design using the shuffled frog leaping algorithm. *Journal of Water Resources planning and management*, *129*(3), 210-225.
7. Naruka, B., Sharma, T. K., Pant, M., Sharma, S., & Rajpurohit, J. (2015). Differential shuffled frog-leaping algorithm. In *Proceedings of Fourth*
8. *International Conference on Soft Computing for Problem Solving* (pp. 249-257). Springer, New Delhi.
9. Naruka, B., Sharma, T. K., Pant, M., Rajpurohit, J., & Sharma, S. (2014, October). Two-phase shuffled frog-leaping algorithm. In *Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization* (pp. 1-5). IEEE.
10. Ali, M., Ahn, C. W., & Siarry, P. (2014). Differential evolution algorithm for the selection of optimal scaling factors in image watermarking. Engineering Applications of Artificial Intelligence, 31, 15-26.
11. Li, P., & Xiao, H. (2014). An improved quantum-behaved particle swarm optimization algorithm. Applied intelligence, 40(3), 479-496.
12. Sharma, T. K., & Pant, M. (2014). Improvised Scout Bee Movements in Artificial Bee Colony. International Journal of Modern Education and Computer Science, 6(1), 1.
13. Deb, K. (2000). An efficient constraint handling method for genetic algorithms. Computer methods in applied mechanics and engineering, 186(2-4), 311-338.
14. Cagnina, L. C., Esquivel, S. C., & Coello, C. A. C. (2008). Solving engineering optimization problems with the simple constrained particle swarm optimizer. Informatica, 32(3).
15. Tsai, J. F. (2005). Global optimization of nonlinear fractional programming problems in engineering design. Engineering Optimization, 37(4), 399-409.
16. Kaveh, A., & Talatahari, S. (2009). Engineering optimization with hybrid particle swarm and ant colony optimization. Asian journal of civil engineering, 10(6), 611-628.
17. Garg, H. (2014). Solving structural engineering design optimization problems using an artificial bee colony algorithm. Journal of Industrial and Management Optimization, 10(3), 777-794.
18. Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. Applied mathematics and computation, 188(2), 1567-1579.
19. Lin, J., & Zhong, Y. (2013). Accelerated shuffled frog-leaping algorithm with Gaussian mutation. *Information Technology Journal*, *12*(23), 7391.
20. Akay, B., & Karaboga, D. (2012). Artificial bee colony algorithm for large-scale problems and engineering design optimization. Journal of intelligent manufacturing, 23(4), 1001-1014.
21. dos Santos Coelho, L. (2010). Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. *Expert Systems with Applications*, *37*(2), 1676-1683.
22. Garg, H. (2019). A hybrid GSA-GA algorithm for constrained optimization problems. Information Sciences, 478, 499-523.
23. Coello, C. A. C., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, *16*(3), 193-203.
24. Belegundu AD. A Study of Mathematical Programming Methods for Structural Optimization, PhD thesis, Department of Civil and Environmental Engineering, University of Iowa, Iowa, USA, 1982.
25. Sandgren, E. (1988). Nonlinear integer and discrete programming in mechanical design. In *Proceeding of the ASME Design Technology Conference* (pp. 95-105).
26. Ray, T., & Saini, P. (2001). Engineering design optimization using a swarm with an intelligent information sharing among individuals. *Engineering Optimization*, *33*(6), 735-748.
27. Ray, T., & Liew, K. M. (2003). Society and civilization: An optimization algorithm based on the simulation of social behavior. IEEE Transactions on Evolutionary Computation, 7(4), 386-396.
28. Dimopoulos, G. G. (2007). Mixed-variable engineering optimization based on evolutionary and social metaphors. Computer methods in applied mechanics and engineering, 196(4-6), 803-817.

29. Kannan, B. K., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. Journal of mechanical design, 116(2), 405-411.
30. He, Q., & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Engineering applications of artificial intelligence, 20(1), 89-99.
31. Mezura-Montes, E., & Coello, C. A. C. (2008). An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. International Journal of General Systems, 37(4), 443-473.
32. Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2011). Mixed variable structural optimization using firefly algorithm. Computers & Structures, 89(23-24), 2325-2336.
33. Kaveh, A., & Talatahari, S. (2010). An improved ant colony optimization for constrained engineering design problems. Engineering Computations, 27(1), 155-182.
34. He, S., Prempain, E., & Wu, Q. H. (2004). An improved particle swarm optimizer for mechanical design optimization problems. Engineering optimization, 36(5), 585-605.
35. Mezura-Montes, E., Coello Coello, C. A., Velázquez-Reyes, J., & Muñoz-Dávila, L. (2007). Multiple trial vectors in differential evolution for engineering design. Engineering Optimization, 39(5), 567-589.
36. Liangwei, C., & Haoping, Y. (2016, October). Regular expression grouping optimization based on shuffled frog leaping algorithm. In *2016 2nd IEEE International Conference on Computer and communications (ICCC)* (pp. 1111-1115). IEEE.
37. Raj, K. H., & Sharma, R. S. (2005). An evolutionary computational technique for constrained optimisation in engineering design.
38. Xue, L., Yao, Y., Zhou, H., & Wang, Z. (2013, March). An improved shuffled frog leaping algorithm with comprehensive learning for continuous optimization. In Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering. Atlantis Press.
39. Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, *41*(2), 113-127.

AUTHORS PROFILE

Bhagyashri Naruka received her B. E. and M. Tech. degree from the University of Rajasthan and Banasthali University, in 2008 and 2010, respectively. She has more than 5 years of rich experience in teaching and joined Amity University Rajasthan in August 2010. Also, she is pursuing her Ph. D. in the field of Computational intelligence. Her areas of interest are in the field of Differential algorithm, Genetic Algorithm, and Statistical Analysis. She has published a plethora of research papers in national and international Journals & conference proceedings and also attended the number of workshops and conferences.

Ashwani Kumar Yadav has done his Ph.D. from Amity University Rajasthan. He did his M. Tech. in VLSI and Embedded System Design and B. Tech. in Electronics & Communication Engineering from reputed technical institutes of Haryana in India. His research interestsinclude image processing, VLSI design and optimization techniques. He has published various research papers in reputed Journals and refereed internationalconferences. He is also serving as editorial board member and reviewer of reputed journals and conferences.

Vaishali did her Ph.D. from Amity University Rajasthan. She has completed her initial education from various reputed educational institutes of Haryana in India. She completed her B. Tech (Information Technology) and M. Tech. (Computer Science & Engineering) in 2007 and 2010 respectively. Her areas of interest are Evolutionary algorithm, and their real world applications. She has published a number of research papers in Journals of repute and in refereed international conferences. She is also contributing as reviewer and editorial boards of a few reputed journals.

Shweta Sharma, B. Tech in Information Technology, M.Tech in Computer Science Engineering and presently pursuing her Ph.D. in Hybridization of Differential Evolution and Shuffled frog leaping Algorithm and their application from Amity University Rajasthan, India.Her research areas are Evolutionary & nature inspired algorithm and their applications for optimization.

Janesh Singh Rathore completed B. Tech. in Mechanical engineering and made all-terrain vehicles and unmanned aero controlled dynamic system during his academics. He has 8 years of rich experience with world-class R&D of Jaguar Land Rover and Mercedes Benz research & development India. He is an expert designer of the fuel system, urea system, and evaporative emission control system.