Differential Shuffled Frog-leaping Algorithm

Bhagyashri Naruka, Tarun Kumar Sharma, Millie Pant, Shweta Sharma and Jitendra Rajpurohit

Abstract Shuffled frog-leaping algorithm (SFLA) is a recent addition to the family of nature-inspired metaheuristic algorithms (NIMA). SFLA has proved its efficacy in solving intricate and real-world optimization problems. In the present study, we have hybridized SFLA into other well-known metaheuristic algorithm called differential evolution (DE) algorithm to enhance the searching capability as well as to maintain the diversity of population. Hybridization is a growing area of interest in research. The process of hybridization results into a new variant that combines the advantages of two or more metaheuristic algorithms in a judicious manner. In this paper, the new variant is named as differential SFLA (DSFLA). The proposal is implemented and shown its efficacy on the problems of optimization of chemical engineering.

Keywords Shuffled frog-leaping algorithm · Differential evolution · Hybridization · SFLA · DE

B. Naruka (✉) · T.K. Sharma · S. Sharma · J. Rajpurohit
Amity University, Jaipur, Rajasthan, India
e-mail: bhagyashree.naruka@gmail.com

T.K. Sharma
e-mail: taruniitr1@gmail.com

S. Sharma
e-mail: shweta_sharma0287@yahoo.com

J. Rajpurohit
e-mail: jiten_rajpurohit@yahoo.com

M. Pant
IIT Roorkee, Roorkee, India
e-mail: millidma@gmail.com

K.N. Das et al. (eds.), *Proceedings of Fourth International Conference on Soft Computing for Problem Solving*, Advances in Intelligent Systems and Computing 336,
DOI 10.1007/978-81-322-2220-0_20

1 Introduction

Optimization in simple terms defined as choosing the best alternative from the given set of solutions. Optimization problems exist in almost every sphere of human activities. Optimization techniques are widely used where decisions have to be taken in some or more complex conditions that can be formulated mathematically. To solve such complex optimization problems, stochastic search techniques gather the attention of many researchers, scientists, and academicians. Stochastic search techniques or nature-inspired metaheuristic algorithms (NIMA) mimic their inspiration from nature or some biological phenomena. Some of the popular stochastic search techniques are GA [1], DE [2], PSO [3], ABC [4], ACO [5], SFLA [6], etc. These algorithms have proved their efficacy in solving intricate and complex optimization problems emerging in various domains.

Shuffled frog-leaping algorithm (SFLA) is a recent addition to the family of NIMA proposed by Eusuff and Lansey [6]. SFLA is formulated on the concept of evolution of memeplexes in frogs. Having the advantage of both, PSO and mixing of the information (taken from GA), SFLA has also proved its efficacy and ability in discovering global optimal solutions to several combinatorial optimization problems. In this study, we have proposed a hybrid differential evolution (DE)-based shuffled leap frog algorithm. This hybridization is done to enhance the searching capability SFLA as well as to maintain the diversity of population.
This paper is structured in five sections including introduction. Section 2 presents working process of SFLA followed by Sect. 3 that describes the proposal. Problem definition is explained in Sect. 4. Simulation settings and results are given in Sect. 5. Finally, the conclusions drawn from the study are mentioned in Sect. 6.

2 Working of SFLA

SFLA, stochastic search algorithm based on evolution of memeplexes. In essence, SFLA contains the element of both the local search method of PSO and the concept of mixing information of the shuffled complex evolution. SFLA has also proved its efficacy in finding global solutions to several combinatorial optimization problems [6]. In SFLA, a set of frogs represents the population of possible solutions, which is partitioned into subsets called memeplexes. Different subsets are having frogs from different cultures and each frog carry out a local search and the position of worst's frog is modified or updated so that the frogs can move toward optimization. When each subset evolves through fixed number of generations or memetic evolution steps, the ideas hold by the frogs within the subset are passed among subsets through shuffling process. This process of local search and shuffling of information continues until the termination criterion is satisfied.

There are four steps in SFLA:

A. *Initialization Process*
The initialization of a set of frogs (solutions) is similar to initialization process of other stochastic techniques, i.e., using Eq. (1). The population of frogs (P) be represented by $X_{Fi} = (x_{i1}, x_{i2}, \ldots, x_{iS})$ and then position of each frog is generated by:

$$x_{ij} = \mathrm{lb}_j + \mathrm{rand}(0,1) \times \left(\mathrm{ub}_j - \mathrm{lb}_j\right) \tag{1}$$

for $i = 1, 2, \ldots, P$ (*set of frogs*); $j = 1, 2, \ldots, S$ (*S-dimensional vector*) and, lb_j and ub_j are the lower and upper bounds, respectively, for the dimension j.

B. *Sorting and Division Process*
The frogs, based on their fitness evaluations, are sorted in descending order. Then, the sorted population of P frogs is distributed into m subsets (memeplexes) each subset holds n frogs such that $P = m \times n$. The distribution is done such that the frog with maximum fitness value will go into subset first, accordingly the next frog into second subset and so on. Then X_b (best) and X_w (worst) individuals in each subset are determined.

C. *Local Search Process*
Worst individual position is improved using Eqs. (2) and (3):

$$D_i = \mathrm{rand}(0,1) \times (X_\mathrm{b} - X_\mathrm{w}) \tag{2}$$

$$X_\mathrm{w} = X_\mathrm{w} + D_i; \quad -D_{\max} \leq D_i \leq D_{\max} \tag{3}$$

where $i = 1, 2, \ldots, N_\mathrm{gen}$; D is the movement of a frog, whereas $D_{\max}$ represents the maximum permissible movement of a frog in feasible domain; N_gen is maximum generation of evolution in each subset. The old frog is replaced if the evolution produces the better solution else X_b is replaced by X_g (optimal solution). If no improvement is observed then a random frog is generated and replaces the old frog. This process of evolution continues till the termination criterion met.

D. *Shuffling Process*
The frogs are again shuffled and sorted to complete the round of evolution. Again follow the same four steps until the termination condition met.

3 Differential SFLA (DSFLA)

Since the inception of SFLA, several attempts have been made and analyzed to improve its performance in terms of accelerating convergence and to balance exploration and exploitation capabilities [7–10]. In this study, we have presented one more variant of SFLA that combines the searching efficiency of DE and SFLA by maintaining the population diversity of frogs. The diversity of populations measures the efficiency of any population-based algorithm.

DE algorithm is proposed by Price and Storn [2] in 2005 to solve global optimization problem. DE uses mutation, crossover, and selection evolutionary operators as that of GA [1] but difference lies in its working. Like SFLA, DE has also been successfully applied and shown its efficacy on benchmark functions and many real-life problems [11–14].

Differential-SFLA (DSFLA) algorithm starts like the usual DE algorithm up to the point of trial vector generation. If the target vector fitness value is better than the fitness value of generated trail vector, then it is included into the population otherwise the SFLA gets activated and generates the new trail vector by following the process of SFLA with expectation of getting better solution. The SFLA phase produces perturbation in the population of solutions (frogs) by dividing it into fixed number of memeplexes. This helps in maintaining the diversity of the population and produces optimal solution. The pseudo code of DSFLA is explained below:

```
Initialize population of frogs
Do
For $i = 1$ to $N$ (population size)
          Select randomly $r_1 \neq r_2 \neq r_3 \in N$ and different from target vector
For $j = 1$ to $D$
          Select $j_{rand} \in D$
          If ($rand$ () < $CR$ or j = $j_{rand}$)
          $U_{ij,g+1} = x_{r_1,g} + F \times (x_{r_2,g} - x_{r_3,g})$

          End If
End for
          If $f(U_{i,g+1} < f(X_{i,g}))$ Then
          $X_{i,g+1} = U_{i,g+1}$
          Else
          SFLA Algorithm Activates
          •    Sorting and Division Process
          •    Local Search
          •    Shuffling
          If $f(X_{Fi}) < f(X_{i,g})$ then
          $X_{i,g+1} = X_{Fi}$
          Else
          $X_{i,g+1} = X_{i,g}$
          End If
          End If
End For
Until termination criteria satisfied
```

If the resulting value falls outside the acceptable range for parameter j, it is set to the corresponding extreme value in that range.

4 Problem Definition

4.1 Optimal Distribution Policy to Minimize the Total Cost of a Company [15]

A company has two alkalyte units (say AL_1 and AL_2) at different locations. From these units, the goods or products are transported to the customers (say CU_1, CU_2, and CU_3). The details about maximum and fixed rate of production from each plant, minimum and fixed customer requirements and the transportation cost are given in Tables 1, 2 and 3, respectively. If the production (ton/day) rate goes down below 0.5 ton/day, than the production cost for AL_1 is 30 \$/ton, while it is 40 \$/ton for production rate above 0.5 ton/day, whereas production cost of AL_2 is steady at 35 \$/ton. The key objective is to find the optimal distribution policy such that the company cost be minimize.

4.2 Circulation Dryer Problem

This problem is given by Luus and Jaakola [16].

Maximize: $P = 0.0064z_1\left[1 - \exp\left(-0.184z_1^{0.3}z_2\right)\right]$

w.r.t.

- power constraint: $(3,000 + z_1)z_1^2 z_2 = 1.2 \times 10^{13}$
- distribution of the moisture content: $\exp\left(0.184z_1^{0.3}z_2\right) = 4.1$.

Table 1 Maximum rates of production from units ($AL_{i=1,2}$)

Units	AL_1	AL_2
Production rate (ton/day)	1.6	0.8

Table 2 Customers ($CU_{i=1,2,3}$) requirements

Customer	CU_1	CU_2	CU_3
Requirements (ton/day)	0.9	0.7	0.3

Table 3 Transfer costs between units ($AL_{i=1,2}$) and customers ($CU_{i=1,2,3}$)

Units	AL_1			AL_2		
Customers	CU_1	CU_2	CU_3	CU_1	CU_2	CU_3
Cost (\$/ton)	25	60	75	20	50	85

4.3 Minimization of Capital Investment for Batch Processes [15]

$$P = 592V^{0.65} + 582V^{0.39} + 1,200V^{0.52} + 370\left(\frac{V}{z_1}\right)^{0.22} + 250\left(\frac{V}{z_2}\right)^{0.40} + 210\left(\frac{V}{z_2}\right)^{0.62} + 250\left(\frac{V}{z_3}\right)^{0.40} + 200\left(\frac{V}{z_3}\right)^{0.85}$$

w.r.t. constraint:

- $V = 50(10 + z_1 + z_2 + z_3)$.

5 Simulation Settings and Results

5.1 Parameter Settings

The above-stated problems are simulated on deb c++ with the following parameters

- The same seed for random number generation is used for all the algorithms (DE, SFLA, and the proposal) so that the initial population is same for all the algorithms.
- All experiments were repeated 25 times independently with 24,000 objective function evaluations for each problem.
- Population size of frogs is fixed to 100.
- Scaling factor (F) = 0.5.
- Crossover rate (Cr) = 0.2.
- m (no. of memeplexes) = 10.
- n (no. of iterations evolves in each memeplexes) = 10.
- N_{gen} = 10.
- $D_{\max}$ = 100 % of variable range.
- Constrained handling is done using parameter-free penalty method [17].
- Integers are handled by rounding of the decision variables to nearest integer [18].

5.2 Results Discussion

The performance of the proposed algorithm D-SFLA is analyzed on a set of three chemical engineering problems taken from the literature. The results for all the three problems are presented in Table 4. All the three algorithms DE, SFLA, and

Table 4 Solution for the problems

Solution for the problem 4.1

Algorithm	Optimal solution	Decision variables					
DE	151.2789	0.782	0.000	0.301	0.100	0.692	0.000
SFLA	151.3173	0.764	0.000	0.320	0.100	0.680	0.000
DSFLA	151.499	0.799	0.000	0.299	0.100	0.700	0.000

Solution for the problem 4.2

Algorithm	Optimal solution	Decision variables	
		z_1	z_2
DE	152.9710	31,765	0.351
SFLA	153.6971	31,764	0.371
DSFLA	153.7101	31,766	0.342

Solution for the problem 4.3

Algorithm	Optimal solution	Decision variables		
		z_1	z_2	z_3
DE	126,302	0.11114	1.46175	3.42476
SFLA	126,301.6	0.11113	1.46136	3.42467
DSFLA	126,302	0.11114	1.46175	

D-SFLA are capable of solving all problems with 100 % success rate. The difference lies in the number of function evaluations taken which are presented in Fig. 1.

It can be analyzed from Fig. 1 that the proposal attained the optimal value 29 and 30 % faster than DE and SFLA, respectively, for the problem 4.1. For the problem 4.2, DSFLA converges 13 and 22 % faster than DE and SFLA. Similarly for the problem 4.3, DSFLA performed 18 and 13 % faster than DE and SFLA. If we further analyze, DFSLA performed 23 and 25 % faster for all the three problems in comparison with DE and SFLA, respectively.

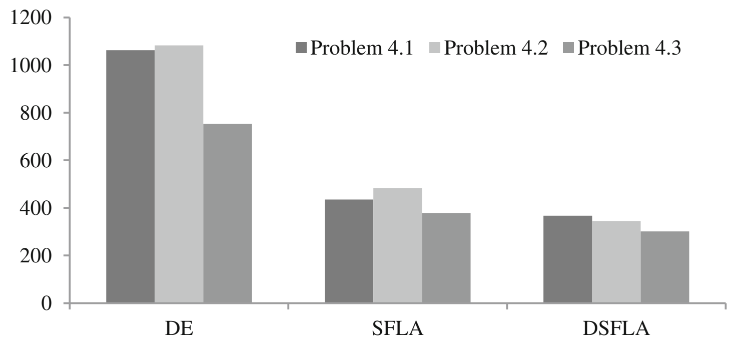

Fig. 1 NFE taken by the problems (4.1, 4.2, and 4.3)

6 Conclusions

In this study, we have proposed the hybridization of DE and SFLA to improve the searching capability of SFLA while maintaining the diversity of the population of the frogs. This kind of hybridization seems to be very efficient in solving computational optimization problems. We have tested the efficacy of the proposal on three chemical engineering problems of different types such linear and nonlinear.

In future, we will try to further investigate the proposal and modify the employment of the proposal on large scale problems.

References

1. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley, Reading (1989)
2. Price, K., Storn, R.: Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report, International Computer Science Institute, Berkley (1995)
3. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceeding of IEEE International Conference on Neural Networks, pp. 1942–1948, Perth, Australia. IEEE Service Center, Piscataway, NJ (1995)
4. Karaboga, D.: An idea based on bee swarm for numerical optimization. Technical Report, TR-06, Erciyes University Engineering Faculty, Computer Engineering Department (2005)
5. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. B **26**(1), 29–41 (1996)
6. Eusuff, M., Lansey, K.E.: Optimization of water distribution network design using the shuffled frog leaping algorithm. Water Resour. Plan. Manage. **129**(3), 210–225 (2003)
7. Wang, L., Gong, Y.: Diversity analysis of population in shuffled frog leaping algorithm. In: Proceedings of ICSI 2013, Part I, LNCS 7928, pp. 24–31 (2013)
8. Elbeltagi, E., Hegazy, T., Grierson, D.: A modified shuffled frog-leaping optimization algorithm: applications to project management. Struct. Infrastruct. Eng. **3**(1), 53–60 (2007)
9. Li, X., Luo, J., Chen, M.-R., Wang, N.: An improved shuffled frog-leaping algorithm with extremal optimisation for continuous optimization. Inf. Sci. **192**, 143–151 (2012)
10. Qiusheng, W., Hao, Y., Xiaoyao, S.: A modified shuffled frog leaping algorithm with convergence of update process in local search. In: Proceedings of 2011 International Conference on Instrumentation, Measurement, Computer, Communication and Control, pp. 1016–1019 (2011)
11. Ali, M., Pant, M.: Improving the performance of differential evolution algorithm using Cauchy mutation. Soft. Comput. **15**(5), 991–1007 (2011)
12. Ali, M., Ahn, C.W., Pant, M.: Multi-level image thresholding by synergetic differential evolution. Appl. Soft Comput. **17**, 1–11 (2014)
13. Ali, M., Ahn, C.W., Pant, M.: A robust image watermarking technique using SVD and differential evolution in DCT domain. Optik Int. J. Light Electron Opt. **125**(1), 428–434 (2014)
14. Jauhar, S.K., Pant, M., Abraham, A.: A novel approach for sustainable supplier selection using differential evolution: a case on pulp and paper industry. Intell. Data Anal. Appl. **II**, 105–117 (2014)
15. Thomas, F.E., Himmelblau, D.M., Leon, L.: Optimization of Chemical Processes, 2nd edn. Mcgraw-Hill, New York (1988)

16. Luus, R., Jaakola, T.: Optimization of nonlinear function subject to equality constraints. Chem. Process Des. Dev. **12**, 38G383 (1973)
17. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. **18**(2–4), 311–338 (2000)
18. Srinivas, M., Rangaiah, G.P.: Differential evolution with tabu list for solving nonlinear and mixed-integer nonlinear programming problems. Ind. Eng. Chem. Res. **46**(22), 7126–7135 (2007)