http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (23): 7391-7395, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.7391.7395

Accelerated Shuffled Frog-leaping Algorithm with Gaussian Mutation

Juan Lin and Yiwen Zhong
College of Computer and Information Science, Fujian Agriculture and Forestry University,
Fujian, Fuzhou, 350002, China

Abstract: This study presents a modified Shuffled Frog Leaping Algorithm (SFLA) which uses a new accelerated method and Gaussian mutation to control its search behavior. Aims to search the solution space more flexible, random disturbance accelerated strategy provides a dynamic expanded neighbor structure. The policy of probabilistic selection is introduced to save the function evaluation times. A Gaussian mutation is also evolved to generate new frog randomly without sacrificing the diversity of the algorithm. Experiments with a wide range of benchmark functions demonstrate good performance of the proposed algorithm when compared with the classic SFLA and other recent variants of SFLA in terms of global optimality, solution accuracy, algorithm reliability and convergence speed.

Key words: Shuffled frog leaping algorithm, acceleration coefficient, gaussian mutation, function optimization

INTRODUCTION

Optimization has been an active area of research for several decades. The Shuffled Frog-Leaping Algorithm (SFLA) is one of the most recent Evolutionary Algorithms (EAs) for solving optimization problems. In SFLA, a frog represents a host for meme. The global optimum is regarded as the stone that has the maximum amount of available food. Through altering the leaping steps, the frogs go forward to the optimum position. The individuals are allowed to communicate and is be infected with each other, so that they can improve their memes using other's information.

SFLA is easy to implement and has been empirically shown to perform well on many optimization problems. However, it may easily get trapped in a local optimum when solving complex multimodel problems. In order to improve SFLA's performance on complex multimodel problems, we present a new improved SFLA utilizing the accelerated strategy and Gaussian mutation (SFLAAG). The random disturbance accelerated factor provides a dynamic and expanded search neighborhood; a probabilistic selection is used to avoid the waste of function evaluation times (FEs). A Gaussian mutation is evolved to generate new frog randomly without sacrifice the diversity of the algorithm. The experimental results show that the proposed SFLAAG balances the explorability and exploitability more effectively, has a more rapid convergence speed and a higher accuracy for global optimization.

This study is organized as follows: In Section 2, the SFLA is briefly reviewed. Section 3 describes the improved SFLA with the accelerated and Gaussian mutation optimizer in detail. Section 4 gives the test functions, the results and discussions. Finally, section 5 summaries the study.

SHUFFLED FLOG LEAPING ALGORITHM

Suppose the search space is D-dimensional, then the I-th frog of the swarm can be represented by a D-dimensional vector, $X_I = (x_{i1},x_{i2},\ldots x_{iD})$. The frogs are sorted in a descending order according to their fitness. The whole population is divided into m memeplexes, each comprising n frogs. (i.e. P = m*n, P is the size of the population). Within each memeplex, the frog with the worst fitness in the submemeplex is identified as X_w, the best one as X_b. Then the step size S and new position of X_w are manipulated according to the following two Eq.:

$$S = \text{rand}()^* (x_b - x_w) \quad (1)$$

$$X^1_w = X_w + S - S_{max} \le S \le S_{max} \quad (2)$$

where, rand () is a uniform random number between 0 and 1; X_w' is the new position. S_{max} is the maximum allowed change in a frogs' position. If this process produces a better solution, X_w is replaced by X_w'. Otherwise the calculation is repeated with respect to the global best frog X_g. If there is still no improvement, a feasible solution to replace X_w is randomly generated. After a specific number of memetic evolution time loops, the memeplexes are shuffled to enhance the exchange of global information. The main parameters of the SFLA are: Number of frogs P,

Corresponding Author: Juan Lin, College of Computerand In formation Science, Fujian Agriculture and Forestry University, Fujian, Fuzhou, 350002, China

number of memeplexes m, number of iterations within eachmemeplex N and the maximum leaping size S_{max}.

Since, Eusuff and Lansey (2003) using the SFLA on the optimization of water distribution network design, a lot of research follow in many areas. On the other side, a large body of research focuses on improving its performance for function optimization problems. Eusuff and Lansey (2003) suggested a set of parameter values for SFLA were used widely. Elbeltagi *et al.* (2005) compared the formulation and results of five algorithms and showed that SFLA is better than Genetic Algorithm (GA) and closer to PSO (Particle Swarm Optimization) with Griewank's function. Elbeltagi *et al.* (2007) introduced a search-acceleration factor to balance the global and local search. Zhen *et al.* (2009) presented a new group policy, in which all frogs participate in the evolvement by keeping the inertia learning behaviors and learning from better ones selected randomly. A Markov chain model for SFLA was established by Lou and Chen (2010). The transition process of the frog memeplex state sequence was analyzed to draw a conclusion that SFLA will eventually converge to the optimal state. Li *et al*., (2012) extended the leaping size and added a leaping inertia component to the update formula. Hybrid Cauchy mutation and Gaussian mutation oddly added to further improve the local search ability. Jiang and Ma (2013) used the principle of orthogonal design to construct the initial population and a new sub-division method to narrow the individual difference. An adaptive factor was also designed to adjust the moving step.

ACCELERATED SFLA WITH GAUSSIAN MUTATION

Acceleration coefficient: In the original SFLA, the size of leaping step is an essential part of individual update, affects the convergence speed and accuracy of algorithm. In the original way, the leaping step deals with component randomly in each dimension, conditioned by a specific maximum size which leads to a restricted scope of the search directly. To expand the possible search range, Elbeltagi *et al.* (2007) add an acceleration factor C into the formula (1) which is named MSFLA in this study. C cannot be too large, or the local search tends to lose in the random search with little improvement. And it cannot be too small to slow down the rate of convergence and lose the diversity of the algorithm. The author suggested *C* is set to 1.3-2.1. In order to make a flexible search neighborhood, we add a random disturbance into the formula to replace *C*, the new update formula is changed as:

$$S = c_1*(X_x-X_w)+c_2*(X_x-X_w) \quad (3)$$

where, c_1, c_2 is random numbers, uniformly distributed in (0, 1). X_x represents X_b or X_g.

In this way, the search coverage expands change dynamically, jumps randomly in the range of (1, 2) which is good for balance the convergence speed and solution accuracy.

Accelerate update with probabilistic selection: In every submemeplex, X_w is first updated with X_b or X_g, only if it cannot be improved by X_b, X_g is used which cause a waste of FEs. In the proposed SFLA, a small probability p_c is designed to choose the X_b or X_g randomly to update the X_w. In this way, X_w is updated with Eq. (2) only once, while keeps the same probability to close the X_b or X_g. P_c must be selected carefully. It cannot be too large, or the slight chance for the X_w to close the X_g. Too small is inappropriate either for all of X_w rush towards the X_g which may cause the convergence. According to the tests not listed here, $p_c = 0.1$ is favourable.

Gaussian mutation: To provide the opportunity for random generation of improved information, random frogs are generated and substituted in the population for the primitive SFLA. The randomly generated strategy is benefit for the diversity of the algorithm, while is not conducive to optimize. Here we evolve a Gaussian mutation to generate new frog randomly. This technique employs the following Eq.:

$$X_w = \text{rand Gaussin } (\mu,\sigma^2) \quad (4)$$

where, μ is the mean of all individuals, σ^2 is the standard deviation of individuals.

Refinement algorithm framework:

- **Step1:** Initialize parameter P, m, n, S_{max},
- **Step2:** While (End condition is not met)

```
Sort P in descending order
Partition P into m memeplexes
For each memeplex T
    Repeat the following operation N times
Update Xw with (3) and (2)
If (Not Improved)
        Update Xw with (4)
    End Repeat
    End For
    Update Xg
End while
```

- **Step 3:** Return X_g

BENCHMARK TESTS AND COMPARISONS

Different experiments to assess the performance of SFLAAG using the test suite described in appendix. The

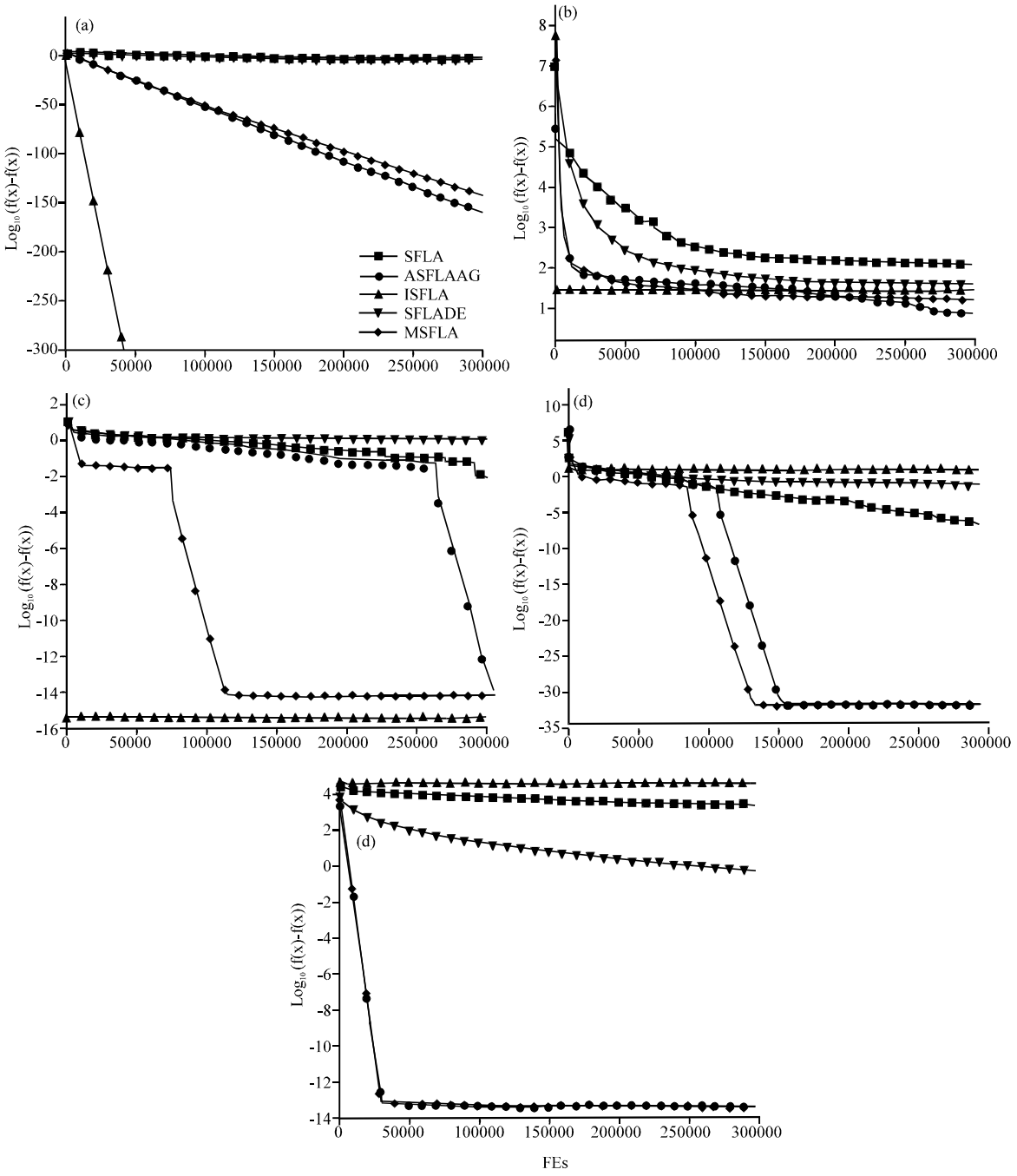

Fig. 1: Convergence performance of the five different SFLAs on 10 test functions. (a) f_1, (b) f_2, (c) f_3, (d) f_4, (e) f_5, (f) f_6, (g) f_7, (h) f_8, (i) f_9, (j) f_{10}

focus of the study was to compare the performance of the proposed SFLAAG with the original SFLA in different experiments. We also studied the performance of SFLAAG comparing with other SFLAs.

Benchmark functions and algorithm configuration: The test suite consists of 10 unconstrained single-objective benchmark functions (Liang *et al*., 2006). The function error value is used to evaluate the performance of the algorithms. With a solution X, the function error value is defined as:

$$\text{Error value} = f(X)-f(X^*) \quad (5)$$

where, X^* is the global optimum of the function.

We follow the parameter settings investigated by Elbeltagi *et al.* (2005). Population size P = 200, the number of memeplexes m = 20, the number of local iterations N = 10. The frogs in each submemeplex are 8. S_{max} is set to 0.4. The maximum number of fitness evaluations that allowed for each algorithm to minimize the error set 10000*D, where D is the dimension of the problem. Each function is processed for 30 times

Comparisons on the solution accuracy: Table 1 were the results of the minimum function error value can be found, recorded in each run and the average and standard deviation (SD) of the error values were calculated and the notation $AVG_{error}\pm SD_{error}$ was used. Table 1 also presents the result on every function of the t-test with a significance level of 0.05 between two algorithms with the best results,+means two algorithm is significantly different. The comparisons in Table 1 show that SFLAAG offers better performance on most test functions than SFLA which are showed in bold.

Table 1: Comparison of SFLAs for mean error values and standard deviation achieved (D = 30)

	SFLA	SFLAAG
f_1	4.61E-06±7.27E-06	2.06E-161±1.08E-160†
f_2	1.27E+02±4.16E+01	7.68E+00±5.45E+00†
f_3	8.37E-03±1.74E-02	8.50E-15±5.19E-15†
f_4	2.84E-08±3.32E-08	1.57E-32±8.35E-48†
f_5	3.80E+03±2.61E+02	3.41E-14±2.83E-14†
f_6	8.73E+03±5.18E+02	4.26E-13±1.26E-13†
f_7	1.02E+04±6.47E+02	6.44E+00±7.75E+00†
f_8	1.37E+04±1.06E+03	4.76E+03±1.39E+03†
f_9	1.52E+08±3.63E+07	6.68E+00±4.92E+00†
f_{10}	1.16E+02±1.06E+01	6.97E+01±1.94E+01†

Table 2: Comparison of SFLA and SFLAAG for *FEs* required achieving goal accuracy values

	SFLA	SFLAAG
f_1	2.95E+05±9.05E+03 (20)	2.50E+04±1.06E+03(100)
f_3	-	3.75E+04±1.34E+03(100)
f_4	2.35E+05±3.12E+04 (49)	2.31E+00±4.26E+03(100)
f_5	-	2.54E+03±9.13E+02(100)
f_6	-	2.97E+05±7.03E+03(100)
f_7	-	2.88E+05±6.21E+00(100)
f_8	-	2.98E+05±5.88E+00(66)
f_{10}	2.89E+05±5.68E+0 (50)	2.68E+05±5.82E+00(100)

Comparisons on the convergence speed: Table 2 were the results of Fes required to reach an error value less than the goal listed in appendix. For this criterion, the notation $AVG_{FEs}\pm SD_{FEs}$ (CNT) was used where CNT is the number of runs in which the algorithm could reach goal using 10000*D Fes at maximum. Table 2 reveals that SFLAAG offers a much higher speed and higher success rate of reaching the goal accuracy values in all functions.

Comparison with Other SFLAs: Table 3 shows the comparison with three other improved SFLA introduced in section II. The first one (Zhen *et al.*, 2009) proposed a new group and update strategy, we denote as ISFLA. The second algorithm (Zhao, 2010) added the mutation idea to disturb updating strategy denoted as SFLADE. The third algorithm (Elbeltagi *et al.*, 2007) introduced a new search-acceleration parameter denoted as MSFLA mentioned in the previous section. The same parameters set to all algorithms expect for the specific parameters mentioned in the studys.

From the results, we observe that ISFLA reaches the global optimum in f1. MSFLA gets the same best results as SFLAAG in f4 and make no significantly different with SFLAAG in f5. For the rest of functions, SFLAAG avoids falling into the deep local optimum successfully and reaches a far minimum value. So we can come to the conclusion that for all unimodal problems, unrotated multimodal problems and rotated multimodal problems, SFLAAG exhibit overall better performance than the other algorithms.

Convergence curves of SFLAs: The speed in obtaining the global optimum is also a yardstick for measuring the algorithms; this section presents the iteration process for five algorithms mentioned above. The Fig. 1 shows the average accuracy values performance of the total runs in respective experiments. From the Fig. 1 we can get the same conclusion as tables above. Ulteriorly, SFLAAG exhibits a higher convergence velocity and higher convergence accuracy. So, in general, it can be

Table 3: Comparison with other SFLAs in terms of error values

	SFLAAG	ISFLA	SFLADE	MSFLA
f_1	2.06E-161±1.08E-160	0.00E+00±0.00E+00†	2.23E-04±1.18E-04	2.01E-143 ±5.22E-143
f_2	7.68E+00±5.45E+00†	2.89E+01±2.01E-02	4.08E+01±3.07E+01	1.60E+01±1.65E+01
f_3	8.50E-15±5.19E-15†	4.44E-16±0.00E+00	1.15E+00±7.79E-01	7.55E-15±0.00E+00
f_4	1.57E-32±8.35E-48	7.53E-01±1.55E-01	4.63E-03±1.92E-02	1.57E-32±7.22E-28
f_5	3.41E-14±2.83E-14	5.78E+04±5.29E+03	7.40E-01±2.88E-01	5.12E-14 ±1.73E-14
f_6	4.26E-13±1.26E-13†	9.14E+04±2.10E+04	5.86E+02±2.19E+02	8.54E-05±6.06E-05
f_7	6.44E+00±7.75E+00†	1.10E+05±2.74E+04	3.92E+03±1.61E+03	4.26E+02 ±1.99E+02
f_8	4.76E+03±1.39E+03†	2.98E+04±1.51E+03	5.51E+03±9.07E+02	5.92E+03 ±9.54E+02
f_9	6.68E+00±4.92E+00†	1.00E+09±0.00E+00	1.09E+04±1.64E+04	1.25E+01±1.21E+01
f_{10}	6.97E+01±1.94E+01†	3.88E+02 ±1.69E+01	7.39E+01±2.79E+01	1.00E+02±1.54E+01

concluded that the improved SFLAAG exhibits overall better performance than the other algorithms shown in the Tables.

CONCLUSION AND FUTURE WORK

To enhance the search ability of SFLA, we present a random disturbance accelerated factor to expand a dynamic search neighbourhood, involve the probabilistic section to reduce the function evaluation times. The Gaussian mutation is used to replace the random generation to keep the diversity of the original algorithm which is also conductive to optimize. The simulation results show that SFLAGA outperforms the classic SFLA both in convergence accuracy and convergence speed in all experimental studies and the overall performance is superior than some other SFLAs selected from literature. In our future study, we will use it to solve some real-word problems.

ACKNOWLEDGMENTS

The research was sponsored by Nature Science Foundation of Fujian Province of P. R. China (Project No. 2013J01216).

REFERENCES

Elbeltagi, E., T. Hegazy and D. Grierson, 2005. Comparison among five evolutionary-based optimization algorithms. Adv. Eng. Inform., 19: 43-53.

Elbeltagi, E., T. Hegazy and D. Grierson, 2007. A modified shuffled frog-leaping optimization algorithm: Applications to project management. Struct. Infrastructure Eng.: Maintenance Manage. Life-Cycle Design Performance, 3: 53-60.

Eusuff, M.M. and K.E. Lansey, 2003. Optimization of water distribution network design using the shuffled frog leaping algorithm. J. Water Resour. Plann. Manage., 129: 210-225.

Jiang, J.G. and P.L. Ma, 2013. An improved shuffled frog leaping algorithm. J. Inf. Comput. Sci., 10: 1665-1673.

Li, X., J. Luo, M.R. Chen and N. Wang, 2012. An improved shuffled frog-leaping algorithm with extremal optimization for continuous optimization. Inform. Sci., 192: 143-151.

Liang, J.J., A.K. Qin, P.N. Suganthan and S. Baskar, 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Tans. Evol. Comput., 10: 281-295.

Lou, J.P. and M.R. Chen, 2010. Study on trajectory and convergence analysis of shuffled frog leaping algorithm and its improved algorithm. Signal Process., 26: 1428-1433.

Zhao, P.J., 2010. Shuffled frog leaping algorithm based on differential disturbance. J. Comput. Appl., 30: 2575-2577.

Zhen, Z., D. Wang and Y. Liu, 2009. Improved shuffled frog leaping algorithm for continuous optimization problem. Proceedings of the IEEE Congress on Evolutionary Computation, May 18-21, 2009, Trondheim, pp: 2992-2995.