International Journal of Computer Applications (0975 – 8887)
Volume 128 – No.5, October 2015

A Hybrid SFL-Bees Algorithm

Duc Hoang Nguyen
Faculty of Electrical and Electronics Engineering
HCMC University of Technology
Ho Chi Minh City, Vietnam

ABSTRACT

This paper proposes Hybrid SFL-Bees Algorithm that combines strengths of Shuffled Frog Leaping Algorithm (SFLA) and Bees Algorithms (BA). While SFLA can find optimal solutions quickly because of directive searching and exchange of information, BA has higher random that make it easily escape local optima to find global solutions. Thus Hybrid SFL-Bees Algorithm is able to find optimal solutions quickly like SFLA and escape local optima like BA. Numerical simulations are carried out on two well-known continuous benchmark functions: Griewangk's function (F8) and Schwefel's function (F7), and the comparative results have shown the effectiveness of the proposed algorithm, its ability to achieve good quality solutions and processing time, which outperforms the SFLA and BA.

General Terms

Intelligent Swam, Algorithms.

Keywords

Optimization, Hybrid, SFLA, Bees Algorithm.

1. INTRODUCTION

Bio-inspired optimization methods have impressively developed in theory as well as application over the past few years. These methods have been successfully applied in many problems whose complexity could not be solved by traditional methods such as gradient or linear programming method. They can be divided into two groups: one group is evolutionary algorithms which are famous as Genetic Algorithm (GA) and the other is swam intelligence inspired algorithms consisting of Ant Colony Optimization (ACO), Particle Swam Optimization (PSO), Shuffled Frog Leaping Algorithm (SFLA) and Bees Algorithms (BA) [1].

Although at present there are a large number of bio-inspired optimization algorithms with many versions, they have some similar features. The first, these algorithms are based on population. The second, initial agents are generated randomly. The third, there must be a cost function to evaluate new agents. The fourth, there must be a convergent criteria. And finally, all of algorithms have the following properties:

- Selection: to choose the best solution in a collection of solutions under examination, or choose solutions to merge together in order to create a new one.
- Local search: to create new solutions have tendency better than available ones.
- Global search: to create random solutions which aim at restraining ones trapped in local optimum.

In an attempt to reduce searching time and improve quality of optimal solution, researchers combined the strengths among these optimization algorithms in order to produce hybrid algorithms which are capable of searching for better solutions. For instance, in [2] authors suggested a hybrid algorithm SFLA-GA that combines the advantages of the SFLA, namely an exchange of information among individual members of the group (frogs), and implements a local search using a GA to evaluate the process results. In [3] authors proposed two hybrid Particle Swarm Optimizers combining the idea of the particle swarm with concepts from Evolutionary Algorithms. The hybrid PSOs combine the traditional velocity and position update rules with the ideas of breeding and subpopulations. Simulation results showed that hybrid algorithms have the potential to achieve faster convergence and the potential to find a better solution. In [4] authors proposed a hybrid algorithm HGAPSO combines the new individual generation function of both GA and PSO, which together mimics social behaviors of animals, breeding, and survival of the fittest. The results in temporal sequence production by FCRNN and dynamic plant control problems by TRFN demonstrated the superiority of HGAPSO over GA and PSO.

In this paper, a hybrid algorithm between SFLA and BA has suggested which combines strength of SFLA, namely the ability to find optimal solution rapidly and strength of BA, namely the ability to escape locally optimal solutions to find global solutions. The remaining paper consists of following sections: sections II, III introduce shortly shuffled frog-leaping algorithm and bees algorithm respectively, a hybrid SFL-Bees algorithm which combines the strengths of SFLA and BA is presented in section IV, simulation results to illustrate strengths of the suggested algorithm is presented in section V and the final section is conclusions.

2. SHUFFLED FROG-LEAPING ALGORITHM

The SFLA is a meta-heuristic optimization method that mimics the memetic evolution of a group of frogs when seeking for the location that has the maximum amount of available food as illustrated in Fig 1.

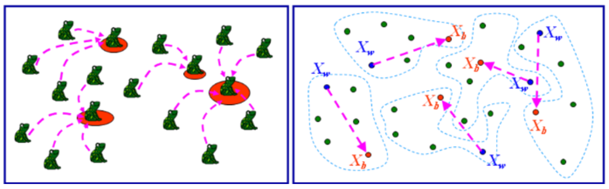

Fig 1: Evolutionary process of frogs

The algorithm contains elements of local search and global information exchange. The SFLA involves a population of possible solutions defined by a set of virtual frogs that is partitioned into subsets referred to as memeplexes. Within each memeplex, the individual frog holds ideas that can be influenced by the ideas of other frogs, and the ideas can evolve through a process of memetic evolution. The SFLA performs simultaneously an independent local search in each memeplex using a particle swarm optimization-like method. To ensure global exploration, after a defined number of memeplex evolution steps (i.e. local search iterations), the virtual frogs are shuffled and reorganized into new memeplexes in a technique similar to that used in the shuffled complex evolution algorithm.

In addition, to provide the opportunity for random generation of improved information, random virtual frogs are generated and substituted in the population if the local search cannot find better solutions. The local searches and the shuffling processes continue until defined convergence criteria are satisfied. The flowchart of the SFLA is illustrated in Fig 2.

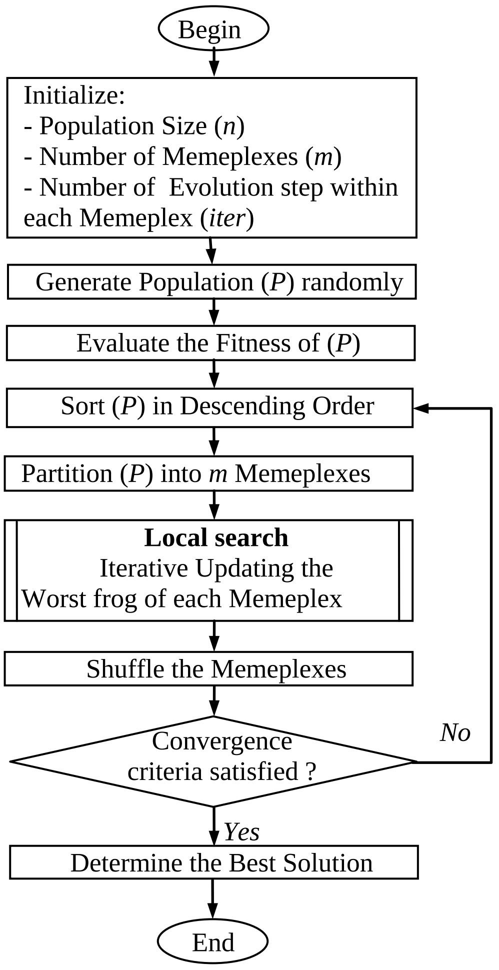
Begin
Initialize:
- Population Size (n)
- Number of Memeplexes (m)
- Number of Evolution step within each Memeplex (iter)
Generate Population (P) randomly
Evaluate the Fitness of (P)
Sort (P) in Descending Order
Partition (P) into m Memeplexes
Local search
Iterative Updating the Worst frog of each Memeplex
Shuffle the Memeplexes
Convergence criteria satisfied ?
No
Yes
Determine the Best Solution
End

Fig 2: Flowchart of the SFLA

The idea updating frog leaping rule which is expressed as :

$$D = r.c(X_b - X_w) \quad (1)$$

$$X_w(new) = X_w + D \quad (2)$$

where Xb and Xw are identified as the frogs with the best and the worst fitness respectively; r is a random number between 0 and 1; c is a constant chosen in the range between 1 and 2. [5]

3. BEES ALGORITHM

The Bees Algorithm is an optimization algorithm inspired by the natural foraging behavior of honey bees to find the optimal solution as illustrated in Fig. 3.

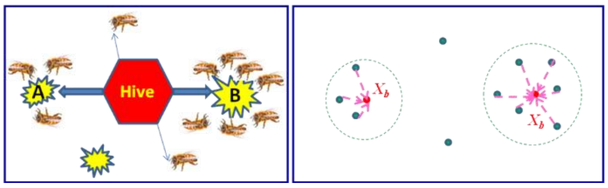

Fig 3: Evolutionary process of bees

The algorithm requires a number of parameters to be set, namely: number of scout bees (n), number of sites selected out of n visited sites (m), number of best sites out of m selected sites (e), number of bees recruited for best e sites (n2), number of bees recruited for the other (m-e) selected sites (n1), initial size of patches (ngh) which includes site and its neighborhood and stopping criterion. The algorithm starts with the n scout bees being placed randomly in the search space. The fitness of the sites visited by the scout bees are evaluated. Bees that have the highest fitness are chosen as "selected bees" and sites visited by them are chosen for neighborhood search. Then, the algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best e sites. The bees can be chosen directly according to the fitness associated with the sites they are visiting. The flowchart of the BA is illustrated in Fig. 4. [6]

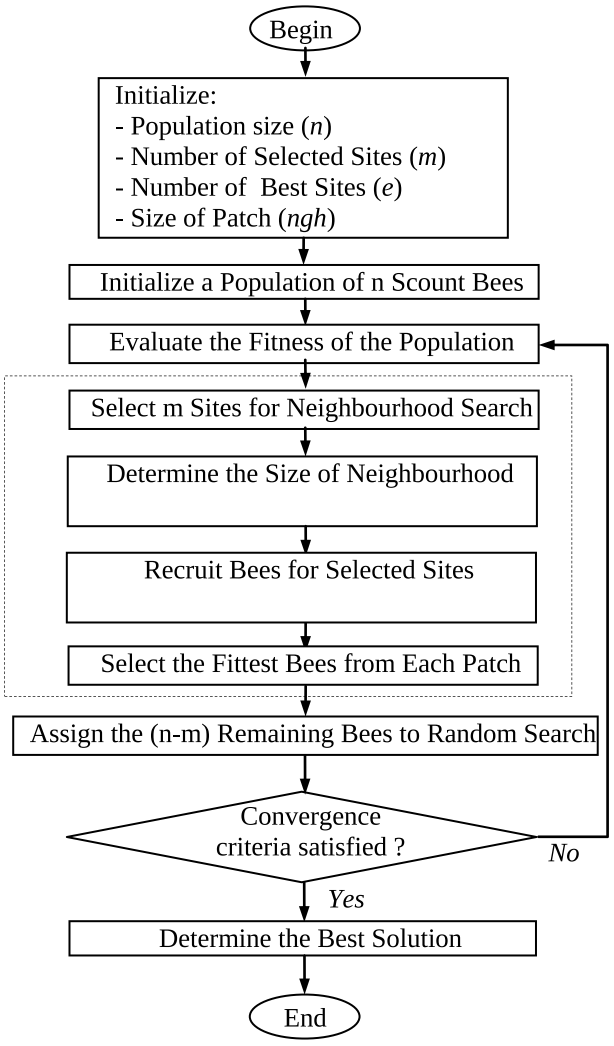
Begin
Initialize:
- Population size (n)
- Number of Selected Sites (m)
- Number of Best Sites (e)
- Size of Patch (ngh)
Initialize a Population of n Scount Bees
Evaluate the Fitness of the Population
Select m Sites for Neighbourhood Search
Determine the Size of Neighbourhood
Recruit Bees for Selected Sites
Select the Fittest Bees from Each Patch
Assign the (n-m) Remaining Bees to Random Search
Convergence criteria satisfied ?
No
Yes
Determine the Best Solution
End

Fig 4: Flowchart of the BA

4. HYBRID SFL-BEES ALGORITHM

In the SFL algorithm, each memeplex evolves independently to locally search at different regions of the solution space. Then, the memeplexes are shuffled and re-divided into new memeplexes in order to globally search through exchanging the information with each other.

From equation (1) $\Rightarrow$ when Xw $\rightarrow$ Xb (Xw $\rightarrow$ Xb) $\Rightarrow$ D $\rightarrow$ 0 $\Rightarrow$ Xw(new) = Xw $\rightarrow$ Xb, i.e, when difference in position between Xw and Xb (Xg) become small, the change in position of frog Xw(new) is small that makes algorithm trapped in local optimum and leads to premature convergence.

Furthermore, information from the best frog is used only once in each update. For Bees Algorithm, information from good bees (Best Sites) are used many times, remaining bees in population will be replaced by random bees. Thus BA tries to "exploit" around position of good bees many times comparing with SFLA. That makes BA be able to find solution with better quality but searching time is longer. While SFLA is capable of finding solution quickly thanks to combining local ang global searching.

Based on analyses of the strengths of SFLA and BA, a Hybrid SFL-Bees Algorithm can be produced as follows:

After completing a generation of SFLA, BA will be used with some minor changes. BA updates their new agents around m selected best bees and m worst bees will be replaced by random bees. (m: number of memeplexes in SFLA). The flowchart of the Hybrid SFL-Bees Algorithm is illustrated in Fig. 5.

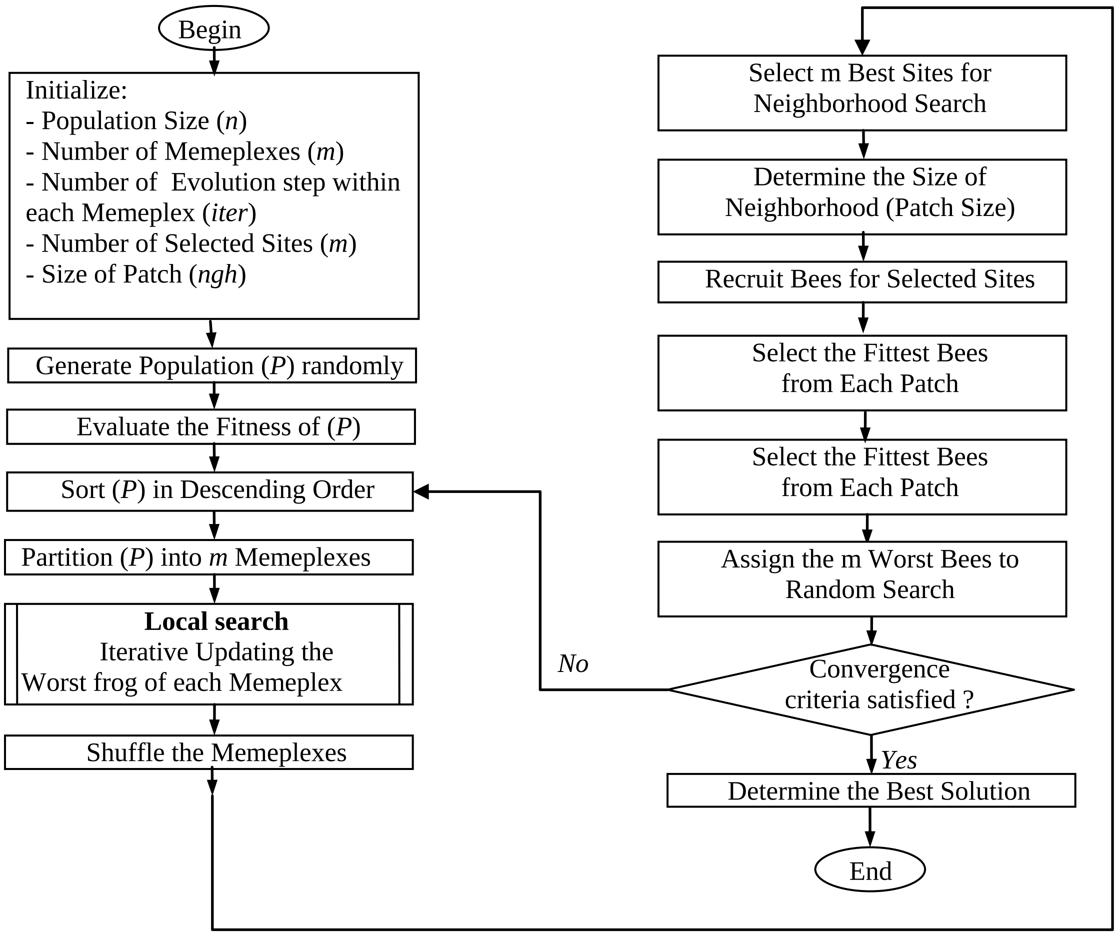
Begin
Initialize:
- Population Size (n)
- Number of Memeplexes (m)
- Number of Evolution step within each Memeplex (iter)
- Number of Selected Sites (m)
- Size of Patch (ngh)
Generate Population (P) randomly
Evaluate the Fitness of (P)
Sort (P) in Descending Order
Partition (P) into m Memeplexes
Local search
Iterative Updating the Worst frog of each Memeplex
Shuffle the Memeplexes
Select m Best Sites for Neighborhood Search
Determine the Size of Neighborhood (Patch Size)
Recruit Bees for Selected Sites
Select the Fittest Bees from Each Patch
Select the Fittest Bees from Each Patch
Assign the m Worst Bees to Random Search
Convergence criteria satisfied ?
No
Yes
Determine the Best Solution
End

Fig 5: Flowchart of the Hybrid SFL-Bees Algorithm

5. SIMULATION RESULTS

All the algorithms are programmed using Microsoft Visual C++ 6.0 programming language and run on Pentium Dual-Core 1.6GHz, RAM 1G Laptop. The performance of three algorithms are compared using two well-known continuous benchmark functions: Griewangk's function (F8) and Schwefel's function (F7). Details of these function are as follows.

5.1 Griewangk's function (F8)

This function is continuous, convex and unimodal. It has many widespread local minima. However, the location of the minima are regularly distributed. It's defined as follows:

$$F8 = 1 + \sum_{i=1}^{N} \frac{x_i^2}{4000} - \prod_{i=1}^{N} cos\left(\frac{x_i}{\sqrt{i}}\right) \quad (3)$$

The F8 function can be scaled to any number of variables N. The values of each variable are constrained to a range (-512 to 511). The global optimum (minimum) solution for this function is known to be zero when all N variables equal zero.

The graphic in Fig. 6 shows the full definition range of the function. When approaching the inner area, the function looks different. Many small peaks and valleys are visible in Fig. 7 . When zooming in on the area of the optimum, Fig. 8 , the peaks and valleys look smooth. This makes the search algorithms be easily trapped in local optima.

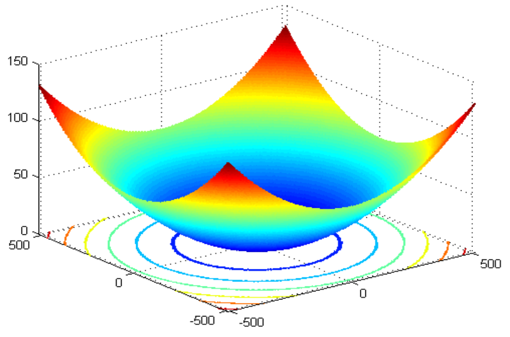

Fig 6: F8 (-512 ÷ 511)

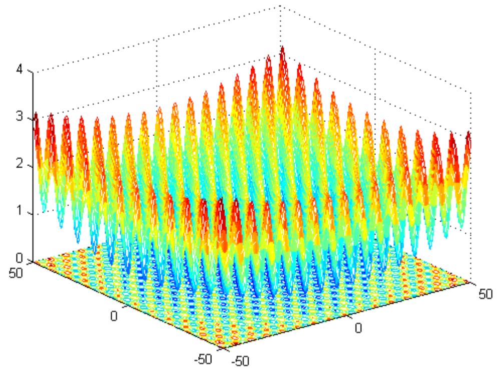

Fig 7: F8 (-50 ÷ 50)

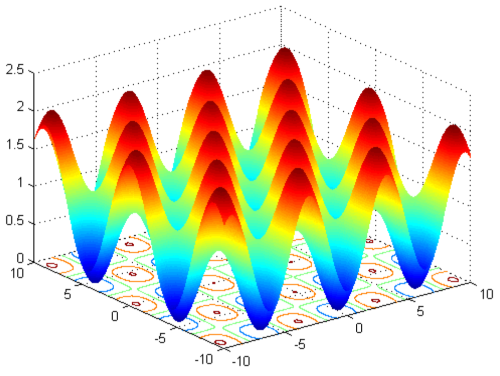

Fig 8: F8 (-10 ÷ 10)

5.2 Schwefel's function (F7)

This function is deceptive in that the global minimum is geometrically distant, over the parameter space, from the next best local minima. Therefore, the search algorithms are potentially prone to convergence in the wrong direction. It's defined as follows:

$$F7 = \sum_{i=1}^{N} -x_i sin\left(\sqrt{|x_i|}\right) \qquad (4)$$

Similar to the F8 function, the F7 function can be scaled to any number of variables N. The values of each variable are constrained to a range (-500 to 500). The global optimum (minimum) solution for this function is known to be -N*418.9829 when all N variables equal 420.9687. The graphic in Fig. 9 shows the full definition range of the function.

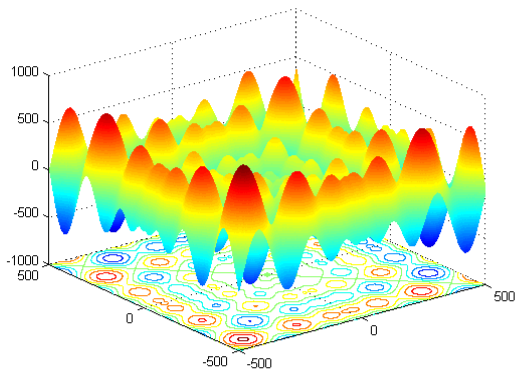

Fig 9: F7 (-500 ÷ 500)

5.3 Parameters setting for algorithms

Each algorithm has its own parameters that effect its performance in terms of solution quality and processing time. The parameters of each algorithm are chosen as in Table 1, 2 and 3. These parameters are chosen through many experiments conducted in order to compromise between solution quality and processing time.

Table 1. Bees Algorithm parameters

n	m	e	n1	n2	ngh
200	20	5	10	20	0.1

Table 2. SFL Algorithm parameters

n	m	p	iter	c
200	20	10	10	2

Table 3. Hybrid SFL-Bees Algorithm parameters

n	m	p	iter	c	e	n1	nrand	ngh
200	20	10	10	2	20	10	20	0.1

5.4 Results and remarks

In all experiments, the solution process stops when one of the two following termination criteria is satisfied:

(i) the objective function reaches a target value of 0.05 (considered as the optimum) for F8 and 99% of optimal value for F7.

(ii) the objective function does not improve in 500 successive generations.

Algorithms are compared based on their successfully search rate, mean objective function value and mean processing time of 50 runs.

Results are summarized in Tables 4-9 and Fig. 10 – 11.

Table 4. Results of Bees Algorithm (F8)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
5	0	1.0068	2.5
10	0	14.3341	5.4
15	24	0.8124	9.8
25	100	0.0449	20.5
50	100	0.0476	60.0
75	100	0.0484	106.8
100	100	0.0491	173.1

Table 5. Results of Shuffled Frog Leaping Algorithm (F8)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
5	50	0.0656	0.3
10	4	0.1815	1.0
15	14	0.1881	1.6
25	96	0.0493	1.8
50	100	0.0492	5.9
75	100	0.0494	11.9
100	100	0.0497	20.9

Table 6. Results of Hybrid SFL-Bees Algorithm (F8)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
5	54	0.0642	0.4
10	6	0.1713	1.2
15	18	0.1269	1.4
25	90	0.0459	0.9
50	100	0.0469	2.1
75	100	0.0482	3.9
100	100	0.0488	6.6

Table 7. Results of Bees Algorithm (F7)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
2	100	-831.7632	0.02
3	100	-1245.3183	0.13
4	98	-1657.9351	0.44
5	66	-2038.9970	1.26
6	18	-2370.9512	2.55
7	2	-2725.4421	3.27
8	2	-2995.4026	3.70

Table 8. Results of Shuffled Frog Leaping Algorithm (F7)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
2	100	-834.1339	0.01
3	80	-1226.2710	0.12
4	44	-1590.3657	0.33
5	16	-1891.9523	0.49
6	4	-2150.3856	0.67
7	0	-2400.2890	0.80
8	0	-2671.5500	1.13

Table 9. Results of Hybrid SFL-Bees Algorithm (F7)

No Variables	% Success rate	Mean obj. func. value	Processing time (s)
2	100	-833.6504	0.01
3	100	-1249.7718	0.08
4	56	-1613.7689	0.49
5	20	-1914.7299	0.80
6	8	-2211.0605	1.10
7	2	-2479.9503	1.21
8	2	-2685.9199	1.42

Remarks

F8

- In case the number of variables that need to be optimized are small (<=10), in spite of not finding optimal solution, the ability to find global optimum of BA is occurred in Hybrid SFL-Bees Algorithm (success rate of Hybrid SFL-Bees Algorithm is 6 comparing to 4 of SFLA, mean objective function value of Hybrid SFL-Bees Algorithm is 0.1713 comparing to 0.1815 of SFLA in case variable is 10, while if the number of variables are 5, respective rates are 54 comparing to 50, 0.0642 comparing to 0.0656).
- In case the number of variables that need to be optimized are 15, although success rate of BA is highest (24%), smallest mean objective function value is of Hybrid SFL-Bees Algorithm (0.1269).
- In case the number of variables that need to be optimized are from 25 – 100, BA searches for the optimum solution better than SFLA does (average greater than 4%). However, processing time of BA is very much greater comparing to SFLA (average greater than 780%). Whereas Hybrid SFL-Bees Algorithm has strength of SFLA, i.e. speed for quickly finding optimum solution and ability to find global optimum solution of BA.

F7

- Results show that mean objective function value of Hybrid SFL-Bees Algorithm is greater than value of BA but smaller than value of SFLA (Hybrid SFL-Bees Algorithm has strength of BA, i.e. be able to globally search for solution) and average processing time of Hybrid SFL-Bees Algorithm is smaller than time of BA but greater than time of SFLA (has strength of SFLA, i.e. can find optimal solution quickly) as demonstrated in Fig. 10 and Fig.11 (plots of Hybrid SFL-Bees Algorithm lie between ones of SFLA and BA).

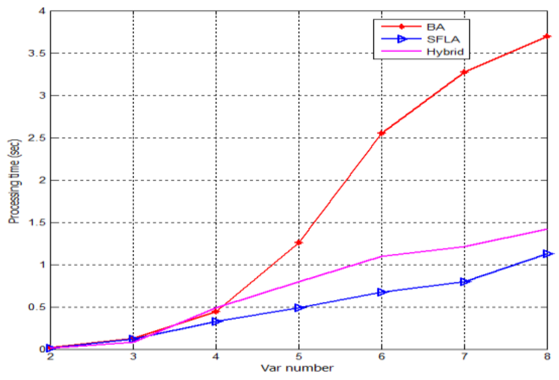

Fig 10: Plot of processing time

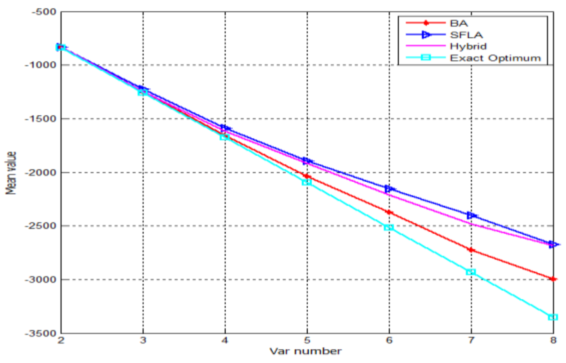

Fig 11: Plot of mean value

6. CONCLUSION

In this paper, a novel algorithm called Hybrid SFL-Bees Algorithm is proposed that combine strengths of SFLA and BA, namely ability to find global optimal solution quickly. Simulation results show that hybrid algorithm outperform each individual algorithm, especially processing time for F8. The future work is to apply the Hybrid SFL-Bees Algorithm for solving other kinds of optimization problems, for instance, tuning parameters of fuzzy controller as in [8].

7. REFERENCES

[1] E. Elbeltagi, T. Hezagy & D. Grierson, “Comparison among five evolutionary-based optimization algorithms”, Advanced Engineering Informatics, vol.19, pp. 43-53, 2005.

[2] Cheng-San Yang, Li-Yeh Chuang , Chao-Hsuan Ke , and Cheng-Hong Yang, “A Combination of Shuffled Frog-Leaping Algorithm and Genetic Algorithm for Gene Selection”, Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.12 No.3, 2008.

[3] Morten Løvbjerg, Thomas Kiel Rasmussen and Thiemo Krink, “Hybrid Particle Swarm Optimiser with Breeding and Subpopulations”, GECCO-2001.

[4] Chia-Feng Juang, “A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design”, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 34, no. 2, April 2004

[5] T.-H. Huynh, “A Modified Shuflled Frog Leaping Algorithm for Optimal Tuning of Multivariable PID Controllers”, IEEE International Conference on Industrial Technology, 2008.

[6] D.T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri , S. Rahim and M. Zaidi, “The Bees Algorithm – A Novel Tool for Complex Optimization Problems”, Manufacturing Engineering Centre, Cardiff University, Cardiff CF24 3AA, UK.

[7] D.T. Pham, A.Haj Dqrwish, E.E. Eldukhri, and S. Otri, “Using the Bees Algorithm to tune a fuzzy logic controller for a robot gymnast”, Proceedings of the 3rd Virtual International Conference on Intelligent Production Machines and Systems, 2007.

[8] Duc-Hoang Nguyen and Thai-Hoang Huynh, “A SFLA-Based Fuzzy Controller for Balancing a Ball and Beam System”, Tenth IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV 2008), Hanoi, Vietnam, 17-20, 2008.