International Journal of Engineering & Technology
CENTRE OF PROFESSIONAL RESEARCH PUBLICATIONS
ISSN 2049-3444

International Journal of Engineering and Technology Volume 4 No. 10, October, 2014

A Comparative Study on Nature Inspired Algorithms with Firefly Algorithm

M. K. A. Ariyaratne, T. G. I. Fernando

Department of Statistics and Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Gangodawila, Nugegoda. Sri Lanka

ABSTRACT

Nature inspired algorithms for their powerfulness, acquire a unique place among the algorithms for optimization. This paper intends to provide a comparison of firefly algorithm (FA) with 3 other nature inspired algorithms; genetic algorithms (GA), particle swarm optimization algorithm (PSO) and ant colony systems (ACS). Traveling salesman problem (TSP) has been used as the problem to be solved and hence, discrete versions of firefly algorithm and particle swarm optimization algorithm were used. Four sets of travelling salesman problems with different number of cities (16, 29, 51, and 100) from a popular TSP library and five sets of randomly generated TSP problems with 29 cities were used to obtain conclusions. Accuracy of the final paths given by each algorithm was compared with the so far best path provided by the TSP library. Times required by each algorithm to solve the problems were also taken in to account when making conclusions. To keep the likeness of all algorithms, individuals in a population, number of iterations in a run and number of runs were kept constant. Simulations and results indicate that the firefly algorithm is superior to the other 3 nature inspired algorithms in solving Traveling Salesman Problems. The significant features of this discrete firefly algorithm are the method of calculating distance between two fireflies and their movement strategy. Finally we will discuss some suggestions for further research.

Keywords: *Firefly Algorithm, Genetic Algorithms, Particle Swarm Optimization Algorithm, Ant Colony Systems, Travelling Salesman Problem, Evolutionary Discrete Firefly Algorithm [EDFA], Nature Inspired Algorithms*

I. INTRODUCTION

Nature inspired optimization techniques have been used to solve many difficult problems of optimization, since their remarkable successfulness in solving optimization problems where other techniques fail. These techniques resemble the optimization processes in natural world which proves their success through the long term existence. Evolutionary algorithms and swarm intelligence are two main areas of nature inspired algorithms. Evolutionary algorithms are meta-heuristic optimization algorithms inspired by biological evolution; reproduction, mutation, recombination, and selection (E.g. genetic algorithms) [1]. Swarm intelligence, by its name, defined as optimizing technique develops due to the grouping behavior of individuals (e.g. bacteria, ants, termites, bees, spiders, fish, and birds) [2]. Firefly algorithm (FA) is a recent nature-inspired meta-heuristic algorithm belong to swarm intelligence, developed by Xing-She Yang [3], originally designed to solve continues optimization problems. However, it can be and it has been discretized effectively to solve permutation problems [4, 5]. The performance of nature inspired algorithms on a problem varies according to the behavior of the algorithm, the operators they use and the tuning of their parameters and etc. The aim of this paper is to compare some nature inspired algorithms with the firefly algorithm on traveling salesman problem (TSP). Three other algorithms were selected; genetic algorithms from evolutionary algorithms; ant colony systems and particle swarm optimization algorithm from swarm intelligence, for the comparison purpose. The study is focus on standard versions of selected nature inspired algorithms and discrete versions of some nature inspired algorithms. However, some ideas to improve the study are slightly discussed at the end of this paper, but no deep analysis.

II. METHODOLOGY

All algorithms were implemented using Matlab. The testing was done using a 32 bit computer with 1GB of RAM, 80GB of hard disk space and 2.39 GHz Intel core 2 CPU under windows XP operating system. The fundamental Genetic Algorithm was implemented. Fitness is taken as (1 / tour length). Roulette wheel selection, partially matched cross over operator with a probability of 95%, reciprocal exchange mutation operator with a probability of 0.1%; were used [6]. Basic Ant colony systems were implemented using same fitness criteria [7]. Appropriate discretization of Particle swam optimization algorithm was used [8]. The only discretization attempt of firefly algorithm to solve traveling salesman problem found at the time of this research, was selected for this comparative study [4]. Symmetric Traveling Salesman Problems with 16, 29, 51 and 100 cities were taken from the standard TSPLIB [9]. Five randomly generated TSP problems with 29 cities were also tested.

III. TRAVELING SALESMAN PROBLEM

To complete the research, four nature inspired algorithms, suitable discrete versions of continuous algorithms and the back ground of travelling salesman problem were studied. The standard travelling salesman problem (TSP) can be stated as; given a list of cities and their pair wise distances, the task is to find the shortest possible route that visits each city exactly once and returns to the origin city [10]. The TSP belongs in the class of combinatorial optimization problems known as NP-complete. Thus, execution time complexity of this problem will be exponential to the size of the input (Number of cities).The TSP is a representative of a larger class of problems

known as combinatorial optimization problems. Specifically, if one can find an efficient algorithm (i.e., an algorithm that guaranteed to find the optimal solution in a polynomial number of steps) for the TSP, then efficient algorithms could be found for all other problems in the NP-complete class. As yet, however, no one has found a polynomial-time algorithm to solve a TSP.

IV. FIREFLY ALGORITHM

Firefly Algorithm, a recent discovery of Xin-She Yang (2008) [3] is one of the best nature inspired meta-heuristic algorithms which stimulates the flashing behavior of natural fireflies. Natural fireflies use their flashing power to attract other fireflies. The Algorithm was implemented assuming following things about their behavior.

1. All fireflies are unisexual, so that one firefly will be attracted to all other fireflies.
2. Attractiveness is proportional to their brightness, for any two fireflies, the less brighter one will be attracted by (and thus move to) the brighter one; however, the brightness can decrease as their distance increases.
3. If there are no fireflies brighter than a given firefly, it will move randomly.

The algorithm was originally proposed for continuous domain problems but its meta-heuristic property allows users to adopt it for discrete problems as well. The pseudo code of the original firefly algorithm is as follows.

Objective function f(x), x= (x_1, …, x_d)

Generate initial population of fireflies x_i (i= 1, 2, ..., n)

Light intensity I_i at x_i is determined by f (x_i)

Define light absorption coefficient γ

while (t < MaxGeneration)

for i= 1 : n all n fireflies

for j= 1 : n all n fireflies

if (I_j > I_i),

move firefly i towards j;

end if

Vary attractiveness with distance r via (exp (-γ r^2))

Evaluate new solutions and update light intensity;

end for j

end for i

Rank fireflies and find the current best;

end while

Post-processing the results and visualization;

The original implementation of firefly algorithm by Xin-She Yang proved that it is superior to both Genetic Algorithms and Particle Swarm Optimization Algorithm. It was very efficient and the success rate was remarkably better than the two algorithms. These conclusions were made only for the continuous domain problems and to check its performance over discrete domains several researches were carried out.

V. EVOLUTIONARY DISCRETE FIREFLY ALGORITHM [EDFA] FOR TRAVELING SALESMAN PROBLEM

One of the discretization attempts of firefly algorithm was made by Gilang Kusuma Jati and Suyanto Suyanto by implementing firefly algorithm to solve TSP [4]. Their paper revealed that it worked fine by protecting the fine properties of the original firefly algorithm. One possible TSP tour was represented by a firefly and the distance ***'r'*** between two fireflies was taken as the arc distance value between the two tours.

Firefly i	1	2	4	6	3	5	7
Firefly j	1	2	4	3	5	7	6

1-2, 2-4, 3-5, 5-7 arcs in firefly i are also presented in firefly j. But 4-6, 6-3 arcs are missing in firefly j. so the arc distance between firefly i and firefly j is 2 here.

In the original algorithm, the attractiveness between two fireflies is calculated using the following formula. Here they have used the same.

$$\beta(r) = \beta_0 * \exp(-\gamma r^2)$$

B_0 is the attractiveness at r=0 and γ is a fixed light absorption coefficient (0<γ<1). According to the equation the attractiveness decreases with the increasing distance.

When moving fireflies towards the brighter ones, each firefly will create '**K**' number of new fireflies using inversion mutation strategy. Length of their mutation cycle is determined by

$$L = \text{random}(2, r_{ij})$$

where "r_{ij}" is the difference between firefly i and firefly j. If we had "**n**" fireflies at the beginning, after an iteration, there will be **nxK** number of fireflies and the best n among **n x K** will be selected for the next iteration.

VI. MOTIVATION

The discrete implementation of firefly algorithm to solve TSP [EDFA] was tested among 7 TSP problems with different number of cities. Results were compared only with the memetic algorithm. Therefore it was felt that a strong

comparison is a need to test the ability of the Firefly algorithm solving TSP with other available nature inspired algorithms. Since the original firefly algorithm was tested with GA and PSO, this comparison study also selected those two and apart from them Ant colony systems was selected. Though a comparative study is easy to carry out, the results are very useful for those who use the applications of TSPs for the real world situations.

VII. RESULTS

Four instances of Travelling Salesman problem (TSP) are applied to the algorithms. They are selected from TSPLIB [9] as well as generated randomly. **Table 1** lists the problem names, numbers of cities and the lengths of the optimal tour, which are taken from TSPLIB [9]. **Table 2** lists the randomly generated Travelling salesman problems. Here, optimal tour length is considered as the best solution given so far from the 4 algorithms.

Table 1 - Problem Names, Numbers of Cities and the Lengths of the Optimal Tour Taken From TSPLIB

Problem names	Number of cities	Length of the optimal tour
ulysses16	16	6859
bays29	29	2020
eil51	51	426
kroA100	100	26282

Table 2 - Randomly Generated TSPs

Problem names	Number of cities	Length of the optimal tour
RandomCitySet1	29	353
RandomCitySet2	29	434
RandomCitySet3	29	396
RandomCitySet4	29	449
RandomCitySet5	29	490

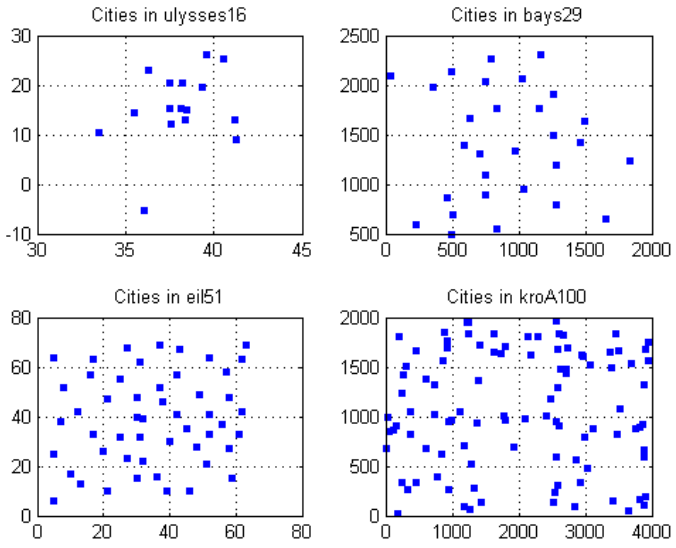

Figure 1 - 4 TSP instances taken from TSPLIB

The 4 algorithms were implemented to solve these 4 TSP instances and the 5 randomly generated TSPs. Each algorithm was run 20 times to take the final solution for each TSP instance. The best solution out of 20 runs, Average best solution out of 20 runs, average time taken to give a solution, and Relative error were collected (A run is consist of predefined number of iterations/Generations). Summary of the obtained results are shown in **Table 3**.

Table 3-The comparison between Firefly Algorithm with other 3 algorithms

Problem Name	**Firefly Algorithm**	**Ant Colony Systems**	**Genetic Algorithms**	**PSO Algorithm**
	Opt/Runs	**Opt/Runs**	**Opt/Runs**	**Opt/Runs**
ulysses16	16/20	1/20	0/20	0/20
bays29	1/20	0/20	0/20	0/20
eil51	0/20	0/20	0/20	0/20
kroA100	0/20	0/20	0/20	0/20
RandomCitySet1	6/20	0/20	0/20	0/20
RandomCitySet2	15/20	0/20	0/20	0/20
RandomCitySet3	12/20	0/20	0/20	0/20
RandomCitySet4	6/20	2/20	0/20	0/20
RandomCitySet5	13/20	2/20	0/20	0/20

Relative error was calculated according to the following formula.

Relative Error = [(Best Tour Length found – Optimum Tour Length) / Optimum Tour Length] X100%

Table 4- Comparison over relative error

Problem Name	**Firefly Algorithm**	**Ant Colony Systems**	**Genetic Algorithms**	**PSO Algorithm**
	Relative Error	**Relative Error**	**Relative Error**	**Relative Error**
ulysses16	0%	0%	6.2254%	0.7290%
bays29	0%	0.9406%	65.6436%	6.0396%
eil51	0.2347%	6.5728%	153.0516%	9.6244%
kroA100	3.7496%	8.7022%	399.3187%	17.0097%
RandomCitySet1	0%	0.57%	110%	2.27%
RandomCitySet2	0%	0.23%	103.22%	4.60%
RandomCitySet3	0%	0.25%	81.06%	1.52%
RandomCitySet4	0%	0%	83%	2.004%
RandomCitySet5	0%	0%	87%	3.06%

The time taken by each algorithm strictly relies on the implementation pattern and the used hardware and software. The recorded time for the 4 algorithms from the implementations using Matlab is shown in **table 5**. But no exact conclusions has made on the performances of the algorithms regarding time.

Table 5- Comparison over time(s)

Problem Name	**Firefly Algorithm**	**Ant Colony Systems**	**Genetic Algorithms**	**PSO Algorithm**
	Avg Time (Sec)	**Avg Time (Sec)**	**Avg Time (Sec)**	**Avg Time (Sec)**
ulysses16	26	124	24	59
bays29	35	206	37	1064
eil51	246	738	126	7373
kroA100	1236	2035	238	15729
RandomCitySet1	35	215	40	1095
RandomCitySet2	35	214	40	1120
RandomCitySet3	36	210	40	1070
RandomCitySet4	35	204	39	1324
RandomCitySet5	35	207	40	1265

CONCLUSIONS AND DISCUSSION

With the results obtained, it can be clearly said that the firefly algorithm is remarkably successful and better than other three algorithms solving different TSPs in its discrete version. In Xin-She Yang's 'Firefly Algorithms for Multimodal Optimization' [3], this is proved for its continuous version by comparing with particle swarm optimization algorithm, which also originally invented for continuous domains.

This discrete version of firefly algorithm is capable of giving better solutions for all 9 TSP instances than other 3 nature inspired algorithms. Ant Colony Systems algorithm also performed well.

The specialty in the firefly algorithm on TSP is that it has three best qualities that a nature inspired algorithm should have.

- It gives solutions which are very closer to the optimal solution.
- It gives the solutions in a very short time, when compared with others.
- It gives good solutions in less number of iterations, when compared with others.

Each algorithm has unique criterion which affect the performance of the algorithm. Number of moves, number of fireflies and their distance measuring criterion are the factors for firefly algorithm. 'Different Arcs' distance measuring concept and the number of moves gives positive effects to the performance. In ant colonies the number of ants is the main factor that affect to the speed and the quality of the algorithm. Increase of the ants to some extent; increase the quality of the solution. Number of chromosomes and the population size directly affect quality of the solution in genetic algorithms. In this research, less number of chromosomes is used to keep the equivalence of the algorithms and hence we can see a great backward of the genetic algorithms in giving better solutions. In particle swarm optimization, number of particles is the stimulating factor for the quality of the solution.

Why this "Discrete Firefly Algorithm solving TSP" outperforms

The significant features of this discrete firefly algorithm are the ***method of calculating distance between two fireflies*** and ***their movement strategy***.
Distance between two fireflies (**'r'**) gives the different arcs between them. It directly measures how unlike the two routes are. This distance **'r'** participates in calculating attractiveness between two fireflies by.

$$\beta(r) = \beta_0 * \exp(-\gamma r^2)$$

β(r) value gets smaller when **'r'** becomes large, meaning low attractiveness when distance increases. This method is much better than hamming distance because sometimes hamming distance gives higher **'r'** values than this method even when the connation between cities are more likely same but when the starting order is different. For example, consider these two fireflies.

1	2	3	5	7	4	6
2	3	5	7	4	6	1

Different arcs between two fireflies: 1

Hamming distance between two fireflies: 7

These fireflies are very similar. But mainly their starting order is different. Here Arc distance value gives more reasonable answer where hamming distance value gives a very contrasting meaning.

The moment strategy used in this approach, directly affect the speediness of the convergence of the problem to a better solution. The moment strategy is consisting of Simple Cycle Inversion Mutation. A random integer is defined within (2, r) [say m] as the cycle length. During a single iteration, a particular firefly will have 'K' number of simple cycle inversion mutations of length 'm'. In another words one firefly will generate 'K' number of new fireflies during a single iteration.

E.g. if we have 10 firefly population with K = 8 number of simple cycle inversion mutations of length 'm', during a single iteration they will produce 80 new fireflies. The best 10 among 81 fireflies (including the best in the previous iteration) are then select to the next round. This feature allows the algorithm to faster convergence and gives better solutions and the algorithm carries on having very small calculations which results great timing. **Table 6** shows the performance of firefly algorithm solving 51 cities TSP with different number of simple cycle inversion mutations

Table 6- firefly algorithm solving 51 cities TSP with different number of simple cycle inversion mutations

Number of moves	Best tour length given	Difference with optimum tour length (426)	Average time (s)	Relative error
1	430	4	160	0.93
2	435	9	170	2.11
4	430	4	195	0.93
6	430	4	225	0.93
8	427	1	246	0.23
10	427	1	306	0.23
15	429	3	440	0.70

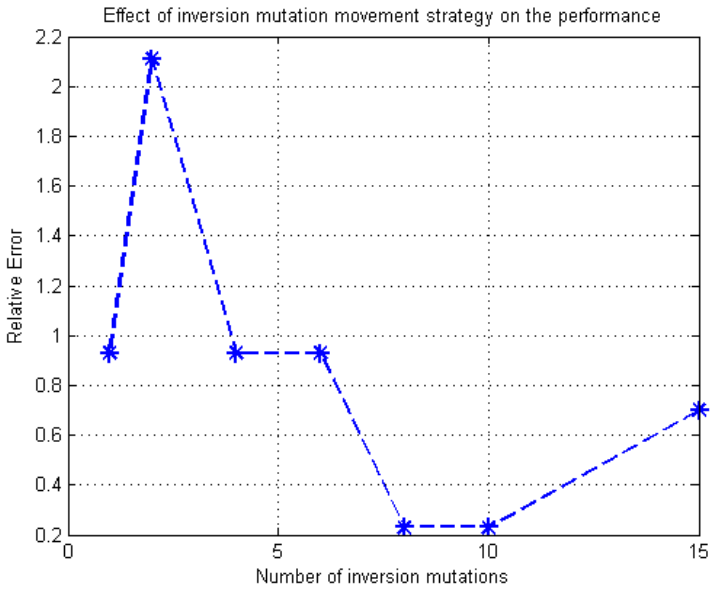

Figure 2 - Effect of inversion mutation movement strategy on the performance of firefly algorithm

SOME IDEAS TO IMPROVE THE RESEARCH

Different discretization versions

Here in this study, one discretization of firefly algorithm [4] and one discretization of particle swarm optimization [5] were used. Anyone can use other discretization of these algorithms or enhanced versions of those algorithms to increase the quality of the results.

Different Nature Inspired Algorithms to compare with the firefly algorithm

In this study, three major nature inspired algorithms were used for the comparison purpose. One can use any other nature inspired algorithms and give the conclusions on the performances of the firefly algorithm.

ACKNOWLEDGEMENTS

The authors wish to thank G. K. Jati and S. Suyanto in Telkom Institute of Technology (IT Telkom), Indonesia for the suggestions and comments on the initial manuscript of EDFA [4], Mr. D. D. A. Gamini, Senior Lecturer, Department of statistics and computer science, Faculty of applied sciences, University of Sri Jayewardenepura, for his valuable suggestion on improving the quality of the research and our colleagues in University of Sri Jayewardenepura for the advices and motivations.

REFERENCES

[1] Wikipedia. “Evolutionary algorithm", Wikipedia.org. [Online]. Available http://en.wikipedia.org/wiki/Evolutionary algorithm [Last Modified: 3 February 2014 at 09:32].

[2] Wikipedia. "Swarm intelligence", Wikipedia.org. [Online].Available http://en.wikipedia.org/wiki/Swarm intelligence [Last Modified: 7 March 2014 at 14:09].

[3] Yang, X.S.: Firefly Algorithms for Multimodal Optimization. In: Watanabe, O., Zeugmann, Teds.) SAGA 2009. LNCS, vol. 5792, pp. 169178. Springer, Heidelberg (2009).

[4] Jati, G.K, and Suyanto. Evolutionary Discrete Firefly Algorithm for Travelling Salesman Problem. In Proceedings of the 2nd International Conference on Adaptive and Intelligence System (ICAIS 2011), Klagenfurt, Austria, September 6-8.

[5] Sayadi, M. K.; Ramezanian, R.; Ghaffari-Nasab, N. (2010). "Discrete fireflies meta-heuristic with local search for make span minimization in permutation flow shop scheduling problems". International Journal of Industrial Engineering Computations 1: 110.

[6] Goldberg, D. (1989), Genetic algorithms in search, optimization, and machine learning. Addison-Wesley: Reading, MA.

[7] Marco Dorigo and Thomas Stutzle, Ant Colony Optimization. The MIT Press, Massachusetts, USA, 2004.

[8] W. L. Zhong, J. Zhang, and W.N. Chen, "A novel discrete particle swarm optimization to solve traveling salesman problem," in Proc. IEEE Int. Conf. Evol. Comput. (CEC), Singapore, Sep. 2007, pp. 32833287.

[9] TSPLIB95: Ruprecht - Karls - UniversitatHeildelberg (2011), https://www.iwr.uniheidelberg.de/groups/comopt/software/TSPLIB95/

[10] Wikipedia. "Travelling salesman problem", Wikipedia.org. [Online].Available http://en.wikipedia.org/wiki/Travelling salesman problem [Last Modified: 14 February 2014 at 09:52].