A Hybrid Metaheuristic Algorithm for Classification using Micro array Data

Mrs.Aruchamy Rajini, Dr. (Mrs.) Vasantha kalyani David

Abstract—A metaheuristic algorithms provide effective methods to solve complex problems using finite sequence of instructions. It can be defined as an iterative search process that efficiently performs the exploration and exploitation in the solution space aiming to efficiently find near optimal solutions. This iterative process has adopted various natural intelligences and aspirations. In this work, to find optimal solutions for microarray data, nature-inspired metaheuristic algorithms were adapted. A Flexible Neural Tree (FNT) model for microarray data is created using nature-inspired algorithms. The structure of FNT is created using the Ant Colony Optimization (ACO) and the parameters encoded in the neural tree are optimized by Firefly Algorithm (FA). FA is used to produce near optimal solutions and hence it is superior to the existing metaheuristic algorithm. Experimental results were analyzed in terms of accuracy and error rate to converge to the optimum. The proposed model is compared with other models for evaluating its performance to find the appropriate model.

Keywords --- ACO, FNT, FA, Metaheuristic, Nature-inspired

1.0 INTRODUCTION

In the recent decades, researchers have developed many optimization computation approaches to solve complicated problems by learning from life system. Optimizing real-life problems is challenging because of huge, complex and noisy solution space. Researchers have proposed approximate evolutionary-based or metaheuristic algorithms as means to search for near-optimal solutions [1].Optimization problems are solved using approximate mathematical search techniques.

These computational systems that mimic the efficient behavior of species such as ants, bees, birds and frogs, as a means to seek faster and more robust solutions to complex and noisy optimization problems. The evolutionary based techniques introduced in the literature were Genetic Algorithm or GA [2], Memetics Algorithm or MAs [2], Shuffled Frog Leaping Algorithm or SFLA [2], Firefly Algorithm or FA [3, 4, 5], Bees Algorithm or BEES [6, 7],Harmony Search Algorithm or HSA [8], Neural Network or NN [9], Ant Colony Optimization or ACO [10], Evolutionary Programming or EP [11], Differential Evolution or DE [12] and Particle Swarm Optimization or PSO [13]. Moreover, there are some with the socially-based inspiration, e.g.Tabu Search or TS [14] and the physically-based inspiration such as Simulated Annealing or SA [15]. These algorithms have been widely used in many social and industrial areas. These kinds of algorithms for scientific computation are called as ''Artificial-Life Computation".

- Aruchamy Rajini, Assistant Professor, Department of Computer Applications,Hindusthan College of Arts Science,Coimbatore, TamilNadu, India. E-mail aruchamy_rajini@yahoo.co.in
- Dr. (Mrs.) Vasantha kalyani David Associate Professor, Department of Computer Science Avinashilingam Deemed University, Coimbatore TamilNadu, India

A relatively new family of nature inspired metaheuristic algorithms known as Swarm Intelligence (SI) has emerged recently, which is known for its ability to produce accurate solutions to complex search problems. This is focused on insect behavior in order to mimic insect's problem solution abilities. The social insect colonies are focused on the interaction between insects contributing to the collective intelligence.

Generally, meta-heuristics work as follows: a population of individuals is randomly initialized where each individual represents a potential solution to the problem. The quality of each solution is then evaluated via a fitness function. A selection process is applied during the iteration of metaheuristic in order to form a new population. The searching process is biased toward the better individuals to increase their chances of being included in the new population. This procedure is repeated until convergence rules are reached [16].

A paradigm for designing metaheuristic algorithms is Ant Colony Optimization (ACO), which is a technique for solving combinatorial optimization problems .The inspiring source of ant colony optimization is the foraging behavior of real ant colonies. Though they live in colonies, they follow their own routine of tasks independent of each other. They perform many complex tasks necessary for their survival. They perform parallel search over several constructive threads based on local problem data and also have a dynamic memory structure which contains information on the quality of previously obtained results.

The meta-heuristic algorithm, which idealizes some of the flashing characteristics of fireflies, has been recently developed and named as Firefly algorithm (FA). Nature inspired methodologies are currently among the most powerful algorithms for optimization problems. Firefly algorithm is a novel nature-inspired algorithm inspired by social behavior of fireflies. Fireflies are one of the most special, captivating and fascinating creature in the nature. Most fireflies produce short and rhythmic flashes though there are about two thousand firefly species. The rate and the rhythmic flash, and the amount of time form part of the signal system which brings both sexes together. Therefore, the main part of a firefly's flash is to act as a signal system to attract other fireflies. By idealizing some of the flashing characteristics of fireflies, the firefly-inspired algorithm was presented by Xin-She Yang [3]. In this paper the author's previous work [17] is extended to experiment the model with different cancer sets. The objective of this paper is to investigate the performance of Firefly and EPSO algorithm to find optimal solutions with different microarray data.

In modern numerical optimization, biologically inspired algorithms are becoming powerful. From the existing metaheuristic algorithms, Firefly Algorithm (FA) is the superior and solves multimodal optimization problems. A Firefly Algorithm deals with multimodal functions more naturally and efficiently.

The Flexible neural tree (FNT) [18] has achieved much success in areas of classification, recognition, approximation and control. In this paper, a tree-structured neural network is created. A FNT model can be created and evolved, based on the pre-defined instruction/operator sets. FNT allows input variables selection, over-layer connections and different activation functions for different nodes. The tree structure is created using Ant Colony Optimization (ACO) and the parameters encoded in the structure are tuned using Firefly Algorithm (FA).

The remainder of this paper is organized as follows: Section II gives the basic model of FNT. Section III describes the tree structure and the parameter learning. Section IV presents experiment results of various cancer data sets to show the effectiveness and robustness of the proposed metaheuristic method. The conclusion is summarized in Section V and it is followed by references.

2.0 THE BASIC FLEXIBLE NEURAL TREE.

A special kind of Artificial Neural Network (ANN) is FNT with flexible neural tree structures. The flexible tree structure is the most distinctive feature, which make it possible for FNT to obtain near-optimal network structures using tree structure optimization algorithms

The FNT model has two stages in the optimization. Firstly the tree structure is optimized using tree structure evolutionary algorithms like genetic programming (GP) [20], probabilistic incremental program evolution algorithm (PIPE) [19], extended compact genetic programming (ECGP), and immune programming (IP) [21], with specific instructions. A tree structure population is generated at the beginning of the optimization process .Each structure in the population is then evaluated and the best one is selected. After selecting the best structure, the next stage is to optimize the parameters of this structure .The fine tuning of the parameters in the structure could be accomplished using simulated annealing (SA) [19], particle swarm optimization (PSO) [22], memetic algorithm (MA) [20].

A new two-stage optimization is executed and this iterative process is called the evolving process of FNT model. Each time a new structure has been found, the parameters of this new structure has to be optimized. It means both the tree structures and parameters are to be optimized [23].

By using tree structure optimization algorithm, it is relatively easy for a FNT model to obtain near-optimal structure. The leaf nodes of FNT are input nodes and the non-leaf nodes are neurons. The root node output is also the output of the whole system. FNT model use two types of instruction sets, the function set F and terminal instruction set T for constructing nodes in tree structures. F is used to construct non-leaf nodes and T is used to construct leaf nodes. They are described as

$$S = F \cup T = \{+_2, +_3, \ldots, +_N\} \cup \{x_1, \ldots, x_n\},$$

where $+_i (i = 2, 3, \ldots, N)$ denote non-leaf nodes' instructions and taking i arguments. $x_1, x_2, \ldots, x_n$ are leaf nodes instructions and taking no other arguments. The output of a non-leaf node is calculated as a flexible neuron model (see Fig.1). From this point of view, the instruction $+_i$ is also called a flexible neuron operator with i inputs.

In the creation process of neural tree, if a nonterminal instruction, i.e., $+_i (i = 2, 3, 4, \ldots, N)$ is selected, i real values are randomly generated and used for representing the connection strength between the node $+i$ and its children. In addition, two adjustable parameters a_i and b_i are randomly created as flexible activation function parameters and their value range are [0, 1]. For developing the forecasting model, the flexible activation function $f(ai, bi, x) = e^{-((x-ai)/bi)2}$ is used.

The total excitation of $+_n$ is

$$net_n = \sum_{j=1}^{n} w_j * x_j,$$

where x_j $(j = 1, 2, \ldots, n)$ are the inputs to node $+_n$ and w_j are generated randomly with their value range are[0,1].The output of the node $+_n$ is then calculated by

$$out_n = f(a_n, b_n, net_n) = e^{-((net_n - a_n)/b_n)2}.$$

The overall output of flexible neural tree can be computed from left to right by depth-first method, recursively [24].

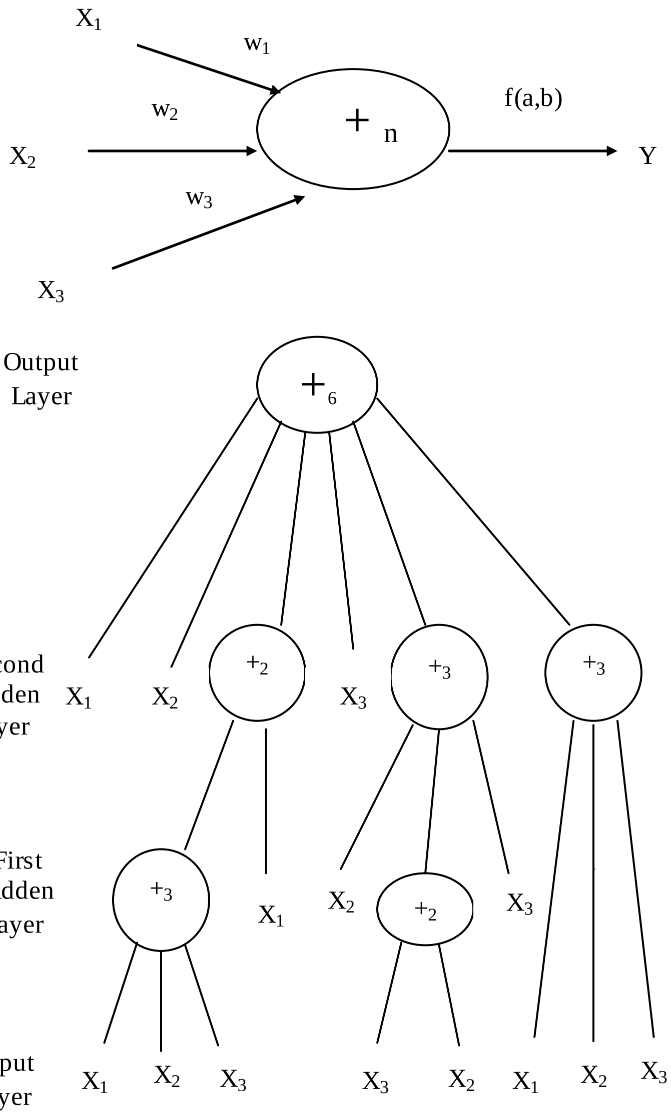


Fig 1. A flexible neuron model and its typical representation of the FNT with function instruction set $F = \{+_2,+_3,+_4,+_5,+_6\}$, and terminal instruction set $T = \{x_1, x_2, x_3\}$ (right)

3.0 TREE STRUCTURE AND PARAMETER LEARNING.

3.1 Tree Structure Optimization

The structure of the FNT model is optimized using Ant Colony Optimization (ACO). ACO is a probabilistic technique developed by Dorigo et al.[10]. This technique was inspired by the foraging behavior of real ants. The ants are able to find the shortest route between their nest and source of food due to the deposition of pheromone in the path. This pheromone trails form has an indirect communication and each ant probabilistically prefers to follow the direction rich in this chemical.

As shown in Fig 2, when ants leave their nest to search for a food source, they randomly rotate around an obstacle, and initially the pheromone deposits will be the same for the right and left directions. When the ants in the shorter direction find a food source, they carry the food and start returning back, following their pheromone trails, and still depositing more pheromone. As indicated in Fig. 2, an ant will most likely choose the shortest path when returning back to the nest with food as this path will have the most deposited pheromone. For the same reason, new ants that later starts out from the nest to find food will also choose the shortest path. Over time, this positive feedback (autocatalytic) process prompts all ants to choose the shorter path [2].

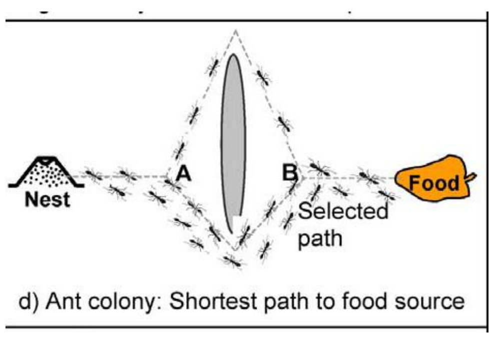


Fig 2: Schematic Diagram of ACO

In the ACO, the process starts by generating m random ants, where each ant will build and modify the trees based on the quantity of pheromone at each node. Each node is memorized with a rate of pheromone and is initialized at 0.5, which means that the probability of choosing each terminal and function is equal initially. At the start, populations of programs are randomly generated. If the rate of pheromone is higher, then the probability to be chosen is also higher. Each ant is then evaluated using a predefined objective function which is given by Mean Square Error (MSE) [24].

$$\text{Fit (i)} = 1/p \sum_{j=1}^{P} (At - Ex)^2 \qquad (1)$$

Where p is the total number of samples, At and Ex are actual and expected outputs of the jth sample. Fit(i) denotes the fitness value of the ith ant.

There are two mechanisms to update the pheromone:

– 1. Trail Evaporation: - Evaporation decreases the rate of pheromone for every instruction on every node, in order to avoid unlimited accumulation of trails, according to following formula:

$$Pg = (1 - \alpha)\, Pg{-}1 \qquad (2)$$

where Pg denotes the pheromone value at the generation g, α is a constant ($\alpha = 0.15$).
– 2.Daemon actions: - The components of the tree will be reinforced according to the Fitness of the tree, for each tree. The formula is

$$P_{i,si} = P_{i,si} + \frac{\alpha}{F_{it(s)}} \qquad (3)$$

where *s* is a solution (tree), *Fit*(*s*) its Fitness, *si* the function or the terminal set at node *i* in this individual, *á* is a constant (*á* = 0.1), *Pi,si* is the value of the pheromone for the instruction *si* in the node *i*[24].

The algorithm is briefly described as follows:(1) every component of the pheromone tree is set to an average value; (2) random generation of tree based on the pheromone; (3) evaluation of ants (4) update of the pheromone; (5) go to step (1) unless some criteria is satisfied[24].

3.2 Parameter learning Optimization

There are a number of learning algorithms such as GA, EP, and PSO that can be used for tuning of parameters (weights and activation parameters) of a neural tree model.

The flashing behavior of fireflies has inspired the development of a metaheuristic algorithm known as firefly algorithm (FA). The fireflies are creatures that generate light inside of it, which is due to the chemical reaction. The bioluminescence is a chemical process generated by the flashing light. The primary purpose of the flashing of fireflies is to act as a signal system to attract other fireflies.

This can idealize the flashing characteristics of fireflies as to consequently develop firefly algorithm. The three idealized rules based on the flashing characteristics of fireflies, are as follows:

Rule 1: All fireflies are unisex, so that one firefly will be attracted to all other fireflies. Each firefly will attract all the other fireflies with weaker flashes [25];

Rule 2: Attractiveness is proportional to their brightness, and for any two fireflies, the less bright one will be attracted (and thus move) to the brightest one; however, the brightness can decrease as their distance increases;

Rule 3: If there are no fireflies brighter than a given firefly, it will move randomly.

The brightness of a firefly is determined by the landscape of the objective function. The brightness can simply be proportional to the value of the objective function, for a maximization problem. The firefly algorithm has two important issues to be considered. They are the variation of light intensity and formulation of the attractiveness .The attractiveness of a firefly determined by its brightness or light intensity which in turn is associated with the encoded objective function.

In the simple case for maximum optimization problems, the brightness I of a firefly at a particular location x can be chosen as $I(x)\,\alpha\, f(x)$.

The attractiveness is relative and so it varies with the distance r_{ij} between firefly i and firefly j. The attractiveness varies with the degree of absorption, as the intensity of light decreases with the distance from its source and the media absorbs the light.
In the simplest form, the light intensity I(r) varies with the distance r monotonically and exponentially. That is

$$I = I_0\, e^{-\gamma r} \qquad (4)$$

where Io is the original light intensity and γ is the light absorption coefficient. As a firefly's attractiveness is proportional to the light intensity seen by adjacent fireflies, the attractiveness β of a firefly can be defined by

$$\beta = \beta_0 \exp(-\gamma r^2) \qquad (5)$$

where β_0 is the attractiveness at r=0. It is worth pointing out that the exponent γr can be replaced by other functions such as γr^m when m>0[4].

The distance between any two fireflies i and j at X_i and X_j can be the Cartesian distance $r_{ij} = ||x_i - x_j||_2$. For other applications such as scheduling, the distance can be time delay or any suitable forms, not necessarily the Cartesian distance [4].

The movement of a firefly i is attracted to another more attractive (brighter) firefly j is determined by

$$X_i = x_i + \beta_0 \exp(-\gamma r^2_{ij})(x_j - x_i) + \alpha\,(\text{rand}-0.5) \qquad (6)$$

where the second term is due to the attraction, while the third term is randomization with α being the randomization parameter. rand is a random number generated uniformly distributed in [0,1] [26].

Firefly Algorithm works as follows:
1. Create an initial population randomly.
2. Light intensity of a firefly is determined by its objective function.
3. Define Light absorption coefficient γ and randomness α in advance.
4. Move firefly towards better brighter ones.
5. Attractiveness varies with distance r through $\exp[-\gamma r]$
6. Evaluate new solutions and update light intensity.
7. If maximum iterations reached, then stop; otherwise go to step (4).
8. Rank the fireflies and find the current best [27].

It can be shown that the limiting case $\gamma \to 0$ corresponds to the standard Particle Swarm Optimization (PSO). If the inner loop (for j) is removed and the brightness I_j is replaced by the current global best g^*, then FA essentially becomes the standard PSO.

3.3 Procedure of the General Learning Hybrid Algorithm .

The proposed procedure interleaves both optimizations. It starts with random structures and corresponding parameters. It first tries to improve the structure and then when the improved structure is found, the parameters are then tuned. It then goes back to improve the structure again and, then tunes the structure and parameters. This loop continues until a satisfactory solution is found or a time limit is reached.

The general learning procedure to find or construct an optimal or near-optimal FNT model structure and parameters optimization are used alternately, combining the ACO and FA algorithm, a hybrid algorithm for evolving FNT model is described as follows:
Step 1: Parameters Definition: Parameters are initialized first i.e, size of population, size of agent and so on.
Step 2: Initialization. An initial population is created randomly (set FNT trees and its corresponding parameters). Create a FNT model M (t) randomly limited by the given parameters .Set t=0.
Step 3: Weights Optimization by firefly algorithm. For each FNT model ($M_0(t), M_1(t), \dots, M_n(t)$) the weights are optimized by firefly algorithm.
Step 4: Structure Optimization by ACO. Create a new population M (t+1) by ACO. Set t=t+1.
Step 5: Iteration: the new solution is submitted to step 2, the process continues iteratively until a stopping criteria is met.

4.0 EXPERIMENTAL STUDIES

The main purpose is to compare the quality of EPSO and FA, where the quality of these algorithms is measured in terms of error rate and accuracy.

4.1 Classification problems

To compare the performance of EPSO and Firefly algorithms, three classification problems are used. They are:

-Breast cancer: The Wisconsin Prognostic breast cancer (WPBC) [28] data set has 34 attributes (32 real-valued) and 2 classes with 198 instances. The methodology adopted for breast cancer data set was applied. Half of the observation was selected for training and the remaining samples for testing the performance.

-Liver cancer: The liver cancer data set (http://genome-www.stanford.edu/hcc/) has two classes, i.e., the nontumor liver and HCC (Hepatocellular carcinoma). The data set contains 156 samples and the expression data of 1,648 important genes. Among them, 82 are HCCs and the other 74 are nontumor livers. The data set is randomly divided into 78 training samples and 78 testing samples. In this data set, there are some missing values and so the k-nearest neighbor method is used to fill those missing values.

-Lymphoma cancer: In the lymphoma data set [5] (http://llmpp.nih.gov/ lymphoma), there are 42 samples derived from diffuse large B-cell lymphoma (DLBCL), nine samples from follicular lymphoma (FL), and 11 samples from chronic lymphocytic leukemia (CLL). The entire data set includes the expression data of 4,026 genes, randomly divided the 62 samples into two parts: 31 samples for training, 31 samples for testing.

4.2 Experimental settings

The algorithms EPSO and Firefly algorithm are applied to above three datasets. For easy comparisons the algorithms was made to run for the same number of iterations .For EPSO, the learning parameters with c1=c2=1.49 are used with the varying inertia. For FA, the parameters $\alpha = 0.2$, $\beta_0 = 1.0$ and $\gamma = 1.0$ are used. Various populations sizes are used from n=15 to 100 and found that it is sufficient to use n= 15 to 50.Therefore a fixed number of population size is used for both the models for comparison.

4.3 Results and Discussions

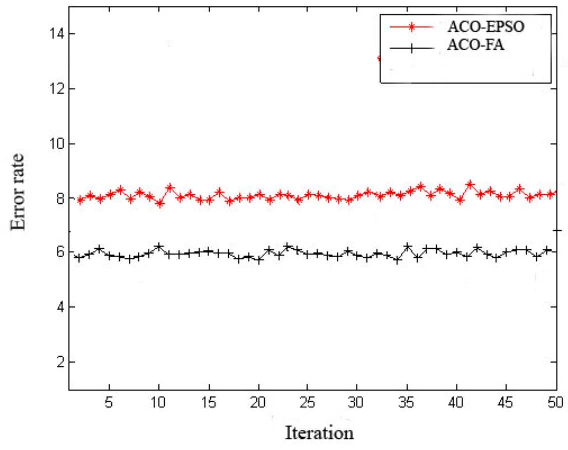
Table 1 summarizes the results obtained from the two algorithms which is applied to medical domain. Three different medical data sets are used to evaluate the performance of the proposed ACO-FA algorithm. This model is compared with ACO-EPSO to show its performance. In this work,the algorithms are implemented in MATLAB. Both the models were trained and tested with same set of data. By applying ACO algorithm an optimal tree structure can be found. In this experiment the input is the number of ant and the number of iterations. Each ant is made to run for a specified number of iterations. Each ant constructs a neural tree with its objective function which is calculated as MSE. The ant which gives the low MSE is taken to be the best tree for which the parameters are optimized with EPSO and FA. The tree which produces the low error is the optimized neural tree and this extracts the informative genes.

Table 1
PERFORMANCE COMPARISION OF EPSO, FIREFLY ALGORITHMS

Problem	Algorithm	Accuracy	Error rate
Breast Cancer	EPSO	86.49	8.00
	Firefly	90.41	6.00
Liver data set	EPSO	86.47	8.02
	Firefly	90.41	6.50
Lymphoma data set	EPSO	86.49	8.03
	Firefly	90.49	6.52

Various iterations imply that FA is more potentially powerful than EPSO for solving many optimization problems. This is partially due to the fact that the broadcasting ability of the current best estimates gives better and quicker convergence towards the optimality. As with different cancer data set, it was well proven that the tree structure with ACO and parameter optimization done with FA can achieve better accuracy and low error rate compared with the other models. The main purpose is to compare the models quality with different medical data set, where the quality is measured according to the error rate and accuracy. The ACO-FA model has the smallest error rate when compared with the other models. The two models are made to run for the same number of iterations and the results shows that ACO-FA success to reach optimal minimum in all runs. This method gives the best minimum points better than the other model. This is depicted in the following figures.

In Figures 3 to 5, it shows that the error rate of the model ACO-FA is low when compared with ACO–EPSO for three medical data sets. From the figures shown below it depicts that the accuracy of the model with ACO-FA is high, which shows that the proposed model is highly efficient that it could be used for faster convergence and low error rate.



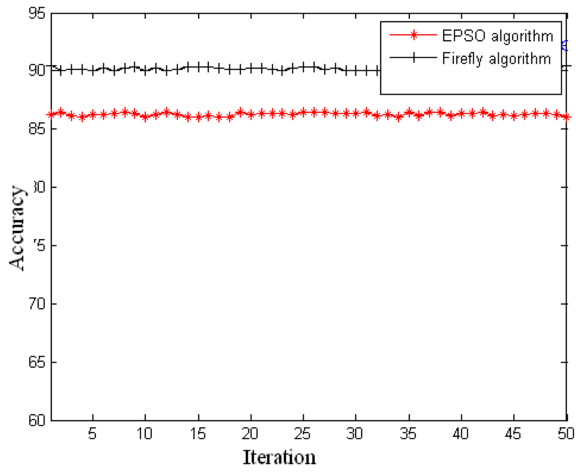
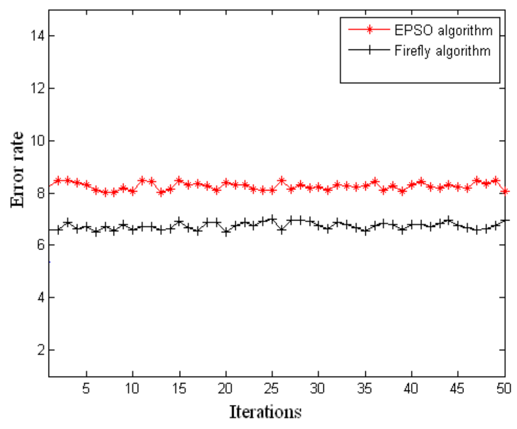


Fig 3: Comparison of models in terms of error rate and accuracy for breast cancer.



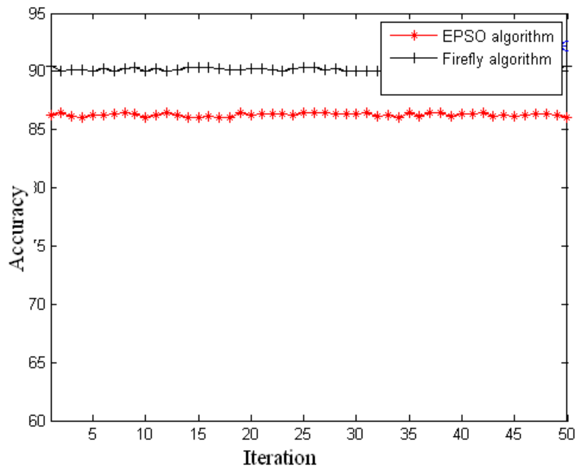


Fig 5: Comparison of models in terms of error rate and accuracy for liver cancer.

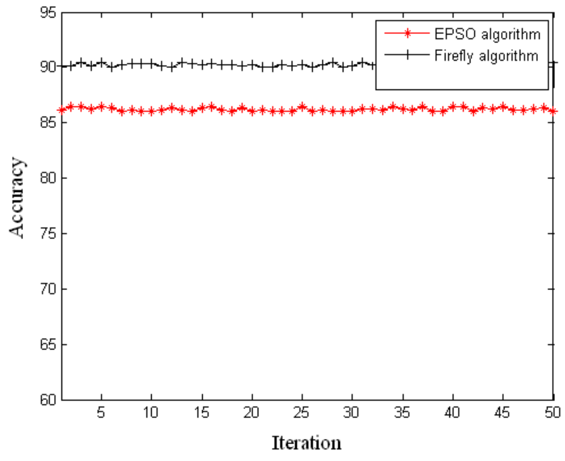
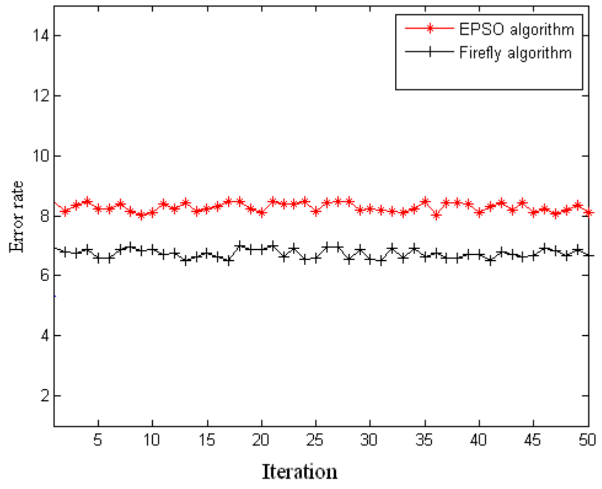


Fig 4: Comparison of models in terms of error rate and accuracy for lymphoma cancer.



5.0 CONCLUSION

A new forecasting model based on neural tree representation by ACO and its parameters optimization by FA was proposed in this paper. A combined approach of ACO and FA encoded in the neural tree was developed. It should be noted that there are other tree-structure based evolutionary algorithms and parameter optimization algorithms that could be employed to accomplish same task but this proposed model yields feasibility and effectiveness .This proposed new model helps to find optimal solutions at a faster convergence. EPSO convergence is slower to low error, while FA convergence is faster to low error. Firefly Algorithm (FA) and Exponential Particle Swarm Optimization (EPSO) were analyzed, implemented and compared. Results have shown that the Firefly Algorithm (FA) is superior to EPSO in terms of both efficiency and success. This implies that the combined approach of ACO-FA is potentially more powerful than the other algorithms, which leads to the proper level of convergence to the optimum. The Proposed method increases the possibility to find the optimal solutions as it decreases with the error rate.

References

[1] E. Elbeltag, T. Hegazy and D. Griersona, “Modified Shuffled Frog-Leaping Optimisation Algorithm: Applications to Project Management”, *Structure and Infrastructure Engineering*, vol. 3, no. 1, 2007, pp. 53 – 60.

[2] E. Emad, H. Tarek, and G. Donald, “Comparison among Five Evolutionary-based Optimisation Algorithms”, *Advanced Engineering Informatics*, vol.19, 2005, pp. 43-53.

[3] X.S. Yang, “A Discrete Firefly Meta-heuristic with Local Search for Make span Minimisation in Permutation

Flow Shop Scheduling Problems", *International Journal of Industrial Engineering Computations*, vol. 1, 2010, pp. 1–10.

[4] X.S. Yang, "Firefly Algorithms for Multimodal Optimisation",Stochastic Algorithms: Foundations and Applications, SAGA 2009,Lecture Notes in Computer Sciences, vol. 5792, 2009, pp. 169-178.

[5] S. Lukasik and S. Zak, "Firefly Algorithm for Continuous Constrained Optimisation Tasks", Systems Research Institute, Polish Academy of Sciences, 2010, pp. 1–10.

[6] D.T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim and M.Zaidi, "The Bees Algorithm. Technical Note". *Manufacturing*Engineering Centre, Cardiff University, UK, 2005.

[7] D.T. Pham, Ghanbarzadeh A., Koc E., Otri S., Rahim S., and M.Zaidi, "The Bees Algorithm - A Novel Tool for Complex Optimisation Problems", Proceedings of IPROMS 2006 Conference,

2006, pp. 454-461.

[8] K.S. Lee and Z.W. Geem, "A New Meta-heuristic Algorithm for

Continues Engineering Optimisation: Harmony Search Theory and

Practice", Comput: Meth. Appl. Mech. Eng., vol. 194, 2004, pp.3902–3933.

[9] P. Muller and D.R. Insua, "Issues in Bayesian Analysis of Neural

Network Models", *Neural Computation*, vol. 10, 1995, pp. 571–592.

[10] M. Dorigo, V. Maniezzo and A. Colorni, "Ant System: Optimisation

by a Colony of Cooperating Agents", *IEEE Transactions on Systems,*

Man, and Cybernetics Part B, vol. 26, numéro 1, 1996, pp. 29-41.

[11] J.Y. Jeon, J.H. Kim, and K. Koh "Experimental Evolutionary

Programming-based High-precision Control," *IEEE Control Sys.*

Tech., vol. 17, 1997, pp. 66-74.

[12] R. Storn, "System Design by Constraint Adaptation and Differential

Evolution", IEEE Trans. on Evolutionary Computation, vol. 3, no. 1,

1999, pp. 22-34.

[13] M. Clerc, and J. Kennedy, "The Particle Swarm-Explosion, Stability,

and Convergence in a Multidimensional Complex Space", *IEEE*

Transactions on Evolutionary Computation, vol. 6, 2002, pp.58-73.

[14] Lokketangen, A. K. Jornsten and S. Storoy "Tabu Search within a

Pivot and Complement Framework", *International Transactions in*

Operations Research, vol. 1, no. 3, 1994, pp. 305-316.

[15] V. Granville, M. Krivanek and J.P. Rasson, "Simulated Annealing: a

Proof of Convergence", Pattern Analysis and Machine Intelligence,

IEEE Transactions, vol. 16, Issue 6, 1994, pp. 652 – 656.

[16] N.Chai-ead,P.Aungkulanon,and P.Luangpaiboon,"*Bees and Firefly Algorithms For Noisy Non-Linear Optimization Problems*"IMECS 2011,Vol II,March16-18,2011,HongKong.

[17] Aruchamy Rajini, Dr. (Mrs)Vasantha Kalayani David,"*AComparative Performance Study on Hybrid Swarm Model for Micro array Data*", International Journal of Computer Applications(0975-8887) , Vol 30 No.6, Sept 2011.

[18] Y. Chen, B. Yang, J. Dong, Nonlinear systems modelling via optimal design of neural trees, International Journal of Neural System 14 (2004) 125–138.

[19] Y. Chen, B. Yang, J. Dong, A. Abraham, Time series forecasting using flexible neural tree model, Information Sciences 174 (2005) 219–235.

[20] Y. Chen, B. Yang, A. Abraham, Flexible neural trees ensemble for stock index modeling, Neurocomputing 70 (2007) 697–703.

[21] Y. Chen, F. Chen, J.Y. Yang, Evolving MIMO flexible neural trees for nonlinear system identification, in: IC-AI, 2007, pp. 373–377.

[22] Y. Chen, A. Abraham, B. Yang, "*Hybrid flexible neural tree based intrusion detection systems*", International Journal of Intelligent Systems 22 (4) (2007) 337–352.

[23] Lizhi Peng,Bo Yang,Lei Zhang,Yuehui Chen,"A Parallel evolving algorithm for flexible neural tree",Parallel Computing 37 (2011),653-666.

[24] Yuehui Chen, Bo Yang and Jiwen Dong, "*Evolving Flexible Neural Networks using Ant Programming and PSO Algorithm*", Springer – Verlag Berlin Heidelberg 2004.

[25] H. Zang, S. Zhang and K. Hapeshi, "A Review of Nature-Inspired Algorithms", *Journal of Bionic Engineering,* vol. 7, 2010, pp. 232–237

[26] Xin-She Yang, X.S., (2010) 'Firefly Algorithm, Stochastic Test Functions and Design Optimisation', Int. J. Bio-Inspired Computation, Vol. 2, No. 2, pp.78-84.

[27] X.-s. Yang,"Firefly Algorithm, Levy flights and global optimization",in: Research and development in Intelligent Systems XXVI (Eds M. Bramer, R. Ellis, M. Petridis), Springer London, pp.209-218(2010).

[28] Available on the UW CS ftp server, ftp ftp.cs.wics.edu, cd math-prog/cpo-dataset/machine-learn/WPBC/