A Comparative Performance Study on Hybrid Swarm Model for Micro array Data

Aruchamy Rajini
Assistant Professor,
Department of Computer Applications
Hindus than College of Arts & Science, Coimbatore
TamilNadu, India

Dr. Vasantha kalyani David
Associate Professor,
Department of Computer Science
Avinashilingam University for Women, Coimbatore
TamilNadu, India

ABSTRACT

Cancer classification based on microarray data is an important problem. Prediction models are used for classification which helps the diagnosis procedures to improve and aid in the physician's effort. A hybrid swarm model for microarray data is proposed for performance evaluation based on Nature-inspired metaheuristic algorithms. Firefly Algorithm (FA) is the most powerful algorithms for optimization used for multimodal applications. In this paper a Flexible Neural Tree (FNT) model for microarray data is constructed using Nature-inspired algorithms. The FNT structure is developed using the Ant Colony Optimization (ACO) and the parameters embedded in the neural tree are optimized by Firefly Algorithm (FA). FA is superior to the existing metaheuristic algorithm and solves multimodal optimization problems. In this research, comparisons are done with the proposed model for evaluating its performance to find the appropriate model in terms of accuracy and error rate.

General Terms

Microarray data, Nature-inspired algorithm

Keywords

ACO, Accuracy, Classification, FNT, FA,

1. INTRODUCTION

Microarray technology has a great potential to discover gene expression levels and hence it has become a powerful tool for clinical diagnosis. Variable selection is used as a preprocess step so that some selected variables can give better solutions. A combinational feature selection method combined with classifier results in the improvement of accuracy and robustness of sample classification.

In recent years ,Artificial Neural Networks (ANNs) was applied successfully to a number of scientific and engineering fields , i.e., image processing, function approximation, system identification and control, time series prediction and so on [1–5]. The performance of a neural network is highly dependent on its structure. The structure specifies the interaction between the various nodes of the network. The Artificial Neural Network structure is not unique for a given problem, and corresponding to the problem there may exist different ways to define a structure. It may be appropriate to have more than one hidden layer, feed forward or feedback connections, or in some cases, direct connections between input and output layer, depending on the problem. A number of attempts have been there to design neural network architectures automatically. The early methods include constructive and pruning algorithms [6–8] but these methods have a main disadvantage is that the topological subsets are often searched using structural hill climbing methods instead of the complete class of ANNs architecture available in the search space [9]. Recent tendencies to optimize ANN architecture and weights include EPNet [10–12] and the NeuroEvolution of Augmenting Topologies (NEAT) [13]. The work of Byoung-Tak Zhang motivated the use of a tree to represent a NN-like model, where a method of evolutionary induction of the sparse neural trees was proposed [14].The architecture and weights of higher order sigma–pi neural networks were evolved by using genetic programming and breeder genetic algorithm, respectively to represent neural trees. An alternative method for the representation of the neural trees was proposed in [15], in which Flexible Neural Tree (FNT) was constructed.

In the past few years, Flexible Neural Tree (FNT) has achieved much success in areas of classification, recognition, approximation and control. The Flexible Neural Tree is a hierarchical structure that could be evolved by using tree structure based algorithms, i.e., genetic programming (GP), gene expression programming (GEP), probabilistic incremental program evolution (PIPE), multi expression programming (MEP), estimation of distribution programming (EDP) and Ant Programming (AP) with specific instructions. The fine tuning of the parameters encoded in the structure could be accomplished by using parameter optimization algorithms, i.e., genetic algorithms (GA), evolution strategy (ES), evolutionary programming (EP), Particle Swarm Optimization (PSO), Exponential Particle Swarm Optimization (EPSO), estimation of distribution algorithm (EDA), and so on.

In this work, a tree-structured neural network is created. A Flexible Neural Tree (FNT) model can be created and evolved, based on the pre-defined instruction/operator sets. This approach allows over-layer connections, and different activation functions for different nodes and input variables selection [16-17]. The tree structure is created using Ant Colony Optimization (ACO) and the parameters encoded in the structure are tuned using Firefly Algorithm (FA).

The most powerful algorithm in modern numerical optimization is the Nature-inspired algorithm. 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.

This paper is an extension of the author's previous work [24], where a Flexible Neural Tree structure is developed using Ant Colony Optimization (ACO) and the parameters are optimized using Exponential Particle Swarm Optimization (EPSO). This model had a slower convergence with low error rate. In this paper, the work is extended to enhance the convergence faster. The rest of the paper is organized as follows: Section II gives the representation and calculation of FNT model. A hybrid learning algorithm is given in Section III. Section IV presents experiment results of breast cancer data set to show the effectiveness and feasibility of the proposed method.

2. REPRESENTATION AND CALCULATION OF NEURAL TREE.

For an Artificial Neural Network (ANN) the commonly used representation are encoding scheme and indirect encoding scheme. There is an alternative approach which is proposed in [15], in which a Flexible Neural Network is constructed by using neural tree representation. This representation reduces computational expenses when calculating the object function.

According to a given problem the user selects instructions for root layer, hidden layer and input layer from the instruction set. For creation of FNT tree a pre-defined instruction set is employed.

The function set *F* and terminal instruction set *T* used for generating a FNT model 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 (instructor) with *i* inputs.

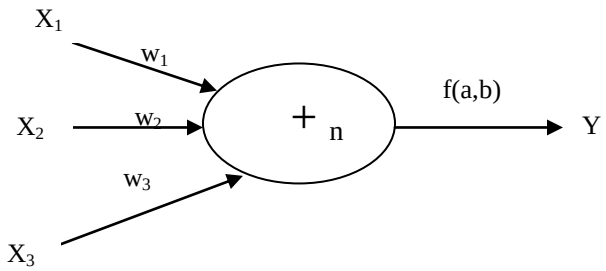

Figure 1. A flexible neuron model

In the creation process of neural tree, if a non-terminal 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]. According to a given task an activation function can vary. For developing the forecasting model, the flexible activation function $f(ai, bi, x) = e^{-((x-ai)/bi)2}$ is used.

The output of a flexible neuron $+_n$ can be calculated as follows: 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 [16]. A typical representation flexible neural tree model is shown in Figure 2.

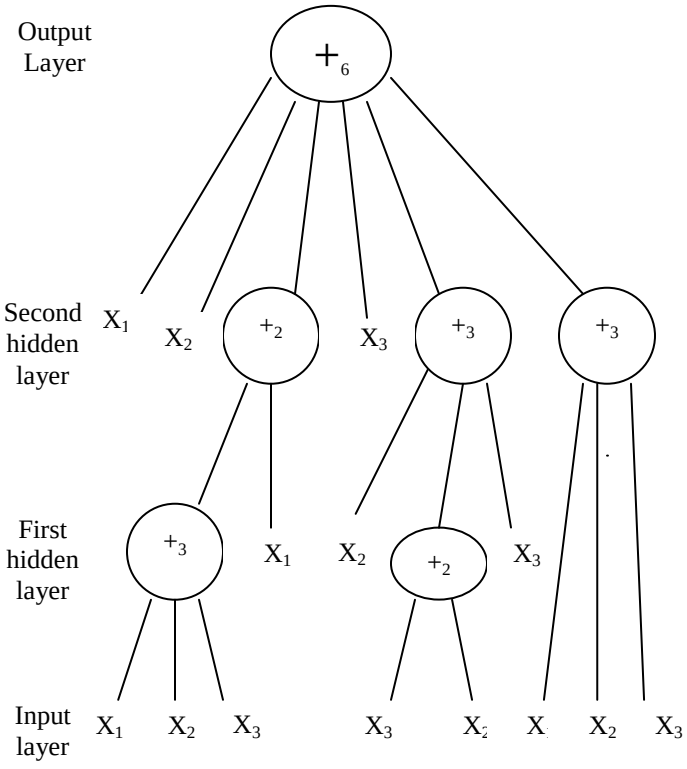

Figure 2.The typical representation of the FNT with function instruction set $F=\{+_2,+_3,+_4,+_5,+_6\}$, and terminal instruction set $T = \{x_1, x_2, x_3\}$.

3.PROPOSED HYBRID ALGORITHM

3.1 Structure Optimization

A new developed form of Artificial Intelligence called Swarm Intelligence, which is a field that studies "the emergent collective intelligence of groups of simple agents". Ant Colony Optimization is a branch of swarm intelligence, in which the ants live in colonies. To optimize the structure of the FNT model, the Ant Colony Optimization (ACO) is used. The algorithm of ACO is based on the concept of each ant will build and modify the trees according to the

quantity of pheromone at each node. The pheromone rate is memorized at each node. At the start, populations of programs are randomly generated. . Each node is initialized at 0.5, which means that the probability of choosing each terminal and function is equal initially. If the pheromone rate is high, the probability to be chosen is also high. Each ant is then evaluated using a predefined objective function which is given by Mean Square Error (MSE) [18].

$$\text{Fit (i)} = 1/p \sum^{p}_{j=1} (At - Ex)^2 \quad (1)$$

Where p is the total number of samples, At and Ex are actual and expected outputs of the j^{th} sample. Fit(i) denotes the fitness value of the i^{th} ant.

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[18].

3.2 Parameter Optimization

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

The Firefly Algorithm (FA) is a nature-inspired, optimization and metaheuristic algorithm which is inspired by the flashing behavior of fireflies. The primary purpose for a firefly's flash is to act as a signal system to attract other fireflies.

The Firefly Algorithm has three idealized rules which are based on the behavior of the flashing characteristics of fireflies [19 - 22]. The three rules are as follows:

1. All fireflies are unisex, so that one firefly will be attracted to the firefly which is brighter regardless of their sex;
2. The degree of attractiveness is proportional to their brightness, which can decrease as fireflies distance increases because of the fact the air absorbs light;
3. If there are no fireflies brighter than a given firefly, it will then move randomly. The brightness of a firefly is determined by the value of the objective function of a given problem. For a maximization problem, the brightness is proportional to the value of the objective function.

This algorithm has two important issues to be considered. They are the variation of light intensity and formulation of the attractiveness. For maximum optimization problems, the simplest case is the brightness I of a firefly at a particular location x can be chosen as $I(x) \alpha f(x)$ [19-22].

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 [19-22]. That is

$$I = I_0 e^{-\gamma r} \quad (2)$$

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

$$\beta = \beta_0 * \exp(-\gamma r^2) \quad (3)$$

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[19-22].

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 [19-22].

The movement of a firefly i which is attracted to by more attractive or brighter firefly j is determined by the equation as follows [19-22]:

$$x_i = x_i + \beta_0 \exp(-\gamma r^2_{ij}) (x_j - x_i) + \alpha (\text{rand} - 0.5) \quad (4)$$

where the first term is the current position of the firefly, 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] [19-22].

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

By limiting the case $\gamma \rightarrow 0$, it shows that it corresponds to the standard Particle Swarm Optimization (PSO). By removing the inner loop (for j) and by replacing the brightness I_j with the current global best g^*, then FA essentially becomes the standard PSO.

3.3 Hybrid Algorithm for FNT Model.

To find an optimal or near-optimal FNT model structure and parameters optimization are used by combining the ACO and FA algorithm. A hybrid algorithm for evolving FNT model is described as follows:
1) Initialization: Create an initial population randomly (Set FNT trees and its corresponding parameters), Parameters are initialized first i.e, size of population, size of agent and so on;
2) Structure optimization is achieved by the Ant Colony Optimization Algorithm in which the fitness function is calculated by mean square error (MSE);
3) If a better structure is found, then go to step 4), otherwise go to step 2);
4) Parameter optimization by the FA algorithm. In this stage, the tree structure or the architecture of FNT model is fixed, and it is the best tree developed during the end of run of the structure search. The parameters (weights and flexible activation function parameters) encoded in the best tree formulate a parameter vector to be optimized by local search;

5) If the maximum number of local search is reached, or no better parameter vector is found for a significantly long time then go to step 6); otherwise go to step 4);
6) If satisfactory solution is found, then its corresponding informative variables are extracted, and the algorithm is then stopped; otherwise go to step (2);

The proposed method interleaves both optimizations. The procedure starts with randomly generated structures and corresponding parameters and weights. It first tries to improve the structure and then as soon as an improved structure is found, it then tunes its parameters and weights. It then goes back to improving the structure again and, fine tunes the structure and rules parameters. This process continues until a satisfactory solution is found or a time limit is reached

4. EXPERIMENTAL RESULTS.

In this section, the results of the ACO-FA model are presented. This model is compared with ACO-EPSO to show its performance. The algorithms are implemented in MATLAB. The breast cancer data set is used to validate the algorithm. The Wisconsin Prognostic breast cancer (WPBC) [23] data set is taken for a preliminary study which has 34 attributes (32 real-valued) and 198 instances. The methodology adopted for breast cancer data set was applied. Half of the observation was selected for training and the rest of the samples were used for testing the performance of the models. Both the models were trained and tested with same set of data. The assessments of the prediction performance of different models were done by quantifying the prediction obtained on an independent data set. The performance of the trained forecasting model for the test data was studied using the Mean Absolute Percentage error (MAPE), error rate and accuracy. MAPE can be defined as:

$$\text{MAPE} = \sum_{i=1}^{N} ((At_i - Pt_i) / At_i) * 100$$

Where N is the total number of samples, At and Pt are actual and predicted outputs of the i^{th} sample.

The instruction set used to create an optimal FNT classifier S = FUT = $\{+_2,\ldots\ldots,+_N\}$ U $\{x_0.x_1,\ldots..,x_{31}\}$Where x_i (i=0,1,….31) denotes the 32 input features. An ACO algorithm is applied to get an optimal tree structure. The number of ant and the number of iterations are given as input, in this experiment. Each ant is made to run for a specified number of iterations. A neural tree is constructed for each ant with its objective function which is calculated as MSE. The best tree is the one in which the ant gives the low MSE and for this best tree 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 variables.

Various iterations show that FA can outperform 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 the iterations of the algorithm continue, the convergence of the algorithms into the global and local optima is achieved, by comparing the best solutions of each iteration with these optima. For EPSO, the learning parameters with c1=c2=1.49 are used with the varying inertia. For FA, the values for control 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 is the above range for both the models for comparison.

As with breast 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, where the quality is measured according to the error rate, mean absolute percentage error 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 Figure 3 and 4 the error rate and mean absolute percentage error of the model ACO-FA is low when compared with ACO–EPSO.

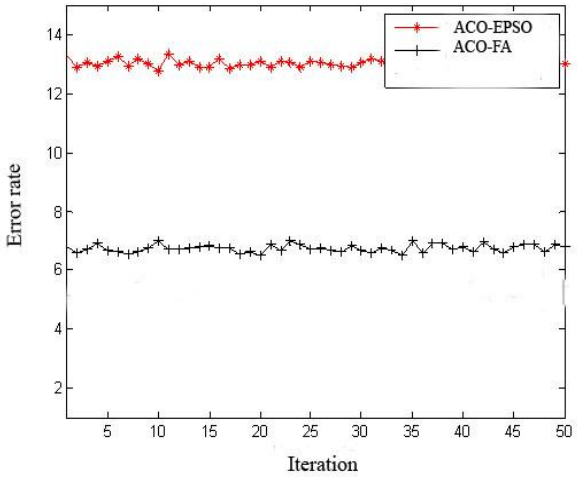

Figure 3: Comparison of models in terms of error rate

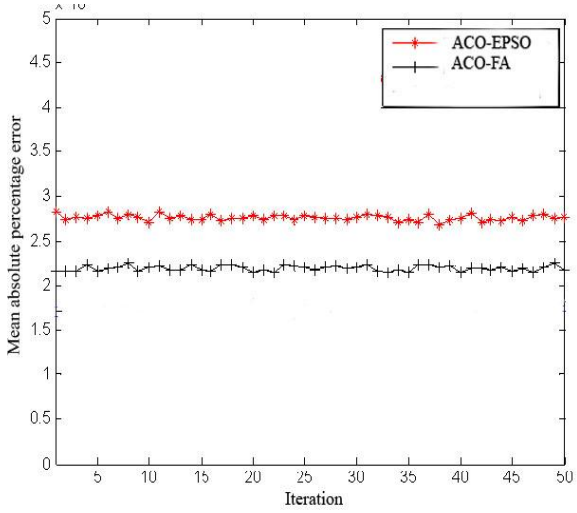

Figure 4: Comparison of models in terms of mean absolute percentage error

In Figure 5 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.

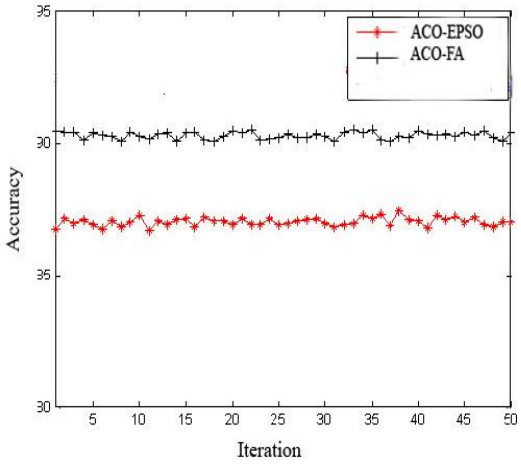

Figure 5: Comparison of models in terms of accuracy

5. 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. The Proposed method increases the possibility to find the optimal solutions as it decreases with the error rate.

6. REFERENCES

[1] S. Omatu, Marzuki Khalid, Rubiyah Yusof, "Neuro-Control and its Applications", SpringerPublisher, 1996.

[2] X. Li, W. Yu, "Dynamic system identification via recurrent multilayer perceptions", Information Science 147 (2002) 45–63.

[3] J.-H. Horng, "Neural adaptive tracking control of a DC motor", Information Sciences 118 (1999) 1–13.

[4] H. Kirschner, R. Hillebr," Neural networks for HREM image analysis", Information Sciences 129 (2000) 31–44.

[5] A.F. Sheta, K.D. Jong, "Time-series forecasting using GA-tuned radial basis functions ", Information Science 133 (2001) 221–228.

[6] S.E. Fahlman, Christian Lebiere, "The cascade-correlation learning architecture", Advances in Neural Information Processing Systems 2 (1990) 524–532.

[7] J.-P. Nadal, "Study of a growth algorithm for a feedforward network", International Journal of Neural Systems 1 (1989) 55–59.

[8] R. Setiono, L.C.K. Hui," Use of a quasi-Newton method in a feedforward neural network construction algorithm", IEEE Transactions on Neural Networks 6 (1995) 273–277.

[9] P.J. Angeline, Gregory M. Saunders, Jordan B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks", IEEE Transactions on Neural Networks 5 (1994) 54–65.

[10] X. Yao, Y. Liu, "A new evolutionary system for evolving artificial neural networks", IEEE Transactions on Neural Networks 8 (1997) 694–713.

[11] X. Yao," Evolving artificial neural networks", Proceedings of the IEEE 87 (1999) 1423–1447.

[12] X. Yao, Y. Liu, G. Lin, "Evolutionary programming made faster", IEEE Transactions on Evolutionary Computation 3 (1999) 82–102.

[13] Kenneth O.Stanley, Risto Miikkulainen, "Evolving neural networks through augmenting topologies", Evolutionary Computation 10 (2002) 99–127.

[14] B.T. Zhang, P. Ohm, H. Muhlenbein," Evolutionary induction of sparse neural trees", Evolutionary Computation 5 (1997) 213–236.

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

[16] Y. Chen, B. Yang, J. Dong, and A. Abraham: "Time-seriesForcasting using Flexible Neural Tree Model", Information Science, In press.

[17] Yuehui Chen, Ajith Abraham and Bo Yang, "Feature Selection & Classification using Flexible Neural Tree", Elsevier Science, 15 January 2006.

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

[19] Xin-She Yang,"Firefly Algorithms for Multimodal Optimization",in:Stochastic Algorithms: Foundations and Applications, SAGA 2009,Vol. 5792, pp. 169-178 (2009).

[20] 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.

[21] 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).

[22] S. Lukasik and S. Zak, "Firefly algorithm for continuous constrained optimization tasks," in Proceedings of the International Conference on Computer and Computational Intelligence (ICCCI '09), N.T. Nguyen, R. Kowalczyk, and S.-M. Chen, Eds., vol. 5796 of LNAI, pp. 97–106, Springer, Wroclaw,Poland, October 2009.

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

[24] Aruchamy Rajini, Dr. (Mrs)Vasantha Kalayani David,"Constructing Models for MicroArray Data with Swarm Algorithms", International Journal of Computer Science and Information Security, Vol 8 No.9, Dec 2010,(pp.237-242).