Neuro Fuzzy Based Routing Protocol for Mobile Ad-Hoc Networks

First A. Siddesh Gundagatti Karibasappa , Second B. K.N.Muralidhara

Abstract-**Wireless Ad Hoc Networks are capable of communication through wireless medium without the need for a pre-existing infrastructure. Much effort has gone into mobile ad-hoc network (MANET) research over the past decade. Yet, even today, mobile ad-hoc networking is seen as a relatively new area of research. The reason for this can be traced to the fact that the maturity in truly understanding these networks is still alarmingly low and actual deployment of these networks rare. There are plenty of techniques in route finding and link establishment in MANET based on various concepts such as "pro-active", "reactive", "power awareness", "cross-layering" etc. Most of these techniques are rather restrictive, taking into account a few of the several aspects that go into effective route establishment. The several factors that decide and influence the routing have to be considered as a whole in the difficult task of finding the best solution in route finding and optimization. The inputs to the system are manifold and apparently unrelated. Most of the parameters are imprecise or non-crisp in nature. The uncertainty and imprecision lead to think that intelligent routing techniques are essential and important in evolving robust and dependable solutions to route finding. The obvious method by which this can be achieved is the deployment of soft computing techniques such as Neural Nets, Fuzzy Logic and Genetic algorithms. Our paper presented here seeks to explore new horizons in this direction. The results of our experimentation with simulator named hypernet have been very satisfactory and we have achieved the goal of optimal route finding to a large extent.**

Keywords-Fuzzy logic, Genetic algorithm, MANET, Neural nets, Routing protocol

I. INTRODUCTION

Wireless Ad Hoc Networks are capable of communication through wireless medium without the need for a pre- existing infrastructure[1]. Wireless Ad Hoc Networks. (MANETs for short) are characterized by their mobility, ease of deployment, self-configuration without a centralized administration and ability of nodes to communicate with each other even in out-of-range conditions with intermediate nodes performing the routing function[3].The features that delineate them from traditional networks are the mobility of the nodes, the absence of need for an infrastructure/ centralized administration and the ability to configure on the fly as the situation demands and can be considered as self healing wireless networks [7].

These unique features impose additional overheads in protocol implementations[8].Compared to Cellular Networks, MANETs are adaptable to changing traffic demands and other physical conditions. Since the attenuation characteristics of wireless media are non-linear, energy efficiency will be superior and increased spatial reuse will guarantee superior capacity and spectral efficiency. These characteristics make Ad Hoc Networks highly attractive for pervasive communications

Since MANETs are generally deployed in disaster management and critical situations, there is a substantial amount of real-time content in their operation. Time plays a crucial role in the communication activities, be it a protocol transfer session or a plain routing operation. In view of these facts, efficient routing protocol implementation assumes the highest level of importance in practical implementations. The efficiency of a routing protocol is directly related to numerous factors such as node mobility, dynamic topology, the communication capabilities of the nodes, power consumption issues, bandwidth constraints, traffic congestion, security and a host of other related parameters, all of which have to be well orchestrated to achieve an optimal performance that is adequate at the minimum level.

When we take all these factors into consideration, evolution of an optimal routing strategy[5] is an indomitable task. These factors are mutually exclusive and there is no explicit relationship of these factors amongst themselves and more importantly we do not see how these are related to an optimal routing strategy. Artificial Neural Network (ANN for short) steps in at this juncture as our savior. An ANN is akin to a biological network, capable of thinking, reasoning, decision making and a high degree of parallelism. It draws inferences from a vast storehouse of knowledge and experience gained over a period of time in

F.A. Author is a research student and is doing Phd in ad hoc wireless communication at PET Research Center, PES College of Engineering, Mandya ,Karnataka, India under the guidance of Dr.K.N.Muralidhara, PES College of Engineering, Mandya and is working as a Professor , Department of Electronics and Communication, Coorg Institute of Technology, Ponnampet, Coorg District, Karnataka,India. Author is available with phone ; 9180 26715595 and mobile 919740112122 . email: professorsiddeshgk@gmail.com

F.B. Author obtained his Masters and Doctoral degree from IIT , Roorke, India and is working as a Professor and Head of the Department , Department of Electronics and Communication, PES College of Engineering ,Mandya, Karnataka ,India .email: knm08@rediffmail.com

978-1-61284-0035-4/11/$26.00 ©2011 IEEE

solving problems. It can work with imprecise and ill-defined parameters in arriving at solutions. Fuzzy Logic and Genetic Algorithms are additional ingredients that can make an ANN more powerful and aggressive in solving unsolvable problems by analytical methods. Fuzzy Logic helps us to work with ill-defined parameters and Genetic Algorithms represent a powerful paradigm in searching for optimal solutions in a solution space. A judicious combination of ANN with Fuzzy Logic and Genetic Algorithms [14] personifies a powerful mechanism in protocol development and routing strategies in Ad Hoc Networks.

1. *Artificial Neural Networks*

According to a simplified account, the human brain consists of about ten billion neurons and a neuron is, on average, connected to several thousand other neurons. Neurons both send and receive varying quantities of energy. One very important feature of neurons is that they don't react immediately to the reception of energy. Instead, they sum their received energies, and they send their own quantities of energy to other neurons only when this sum has reached a threshold. The brain learns by adjusting the number and strength of these connections. Even though this picture is a simplification of the biological facts, it is sufficiently powerful to serve as a model for the neural net. The first step toward understanding neural nets is to abstract from the biological neuron, and to focus on its character as a threshold logic unit (TLU). A TLU is an object that inputs an array of weighted quantities, sums them, and if this sum meets or surpasses some threshold, outputs a quantity. Let's label these features. First, there are the inputs and their respective weights: $X_1, X_2, \ldots, X_n$ and $W_1, W_2, \ldots, W_n$. Then, there are the $X_i * W_i$ that are summed, which yields the activation level which is given by the mathematical expression as

$$\text{Å } a = \sum_{k=1}^{n} X_k W x_k$$

The threshold is called theta. Lastly, there is the output: When $a >= theta$, $y = 1$, else $y = 0$. Notice that the output doesn't need to be discontinuous, since it could also be determined by a squashing function, s (or sigma), whose argument is a, and whose value is between 0 and 1. Then, $y = s(a)$ is as shown in" Fig. 1,".

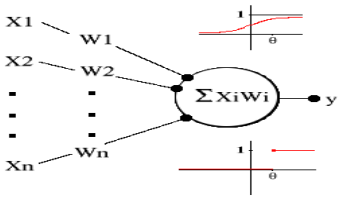


Fig. 1. Threshold logic unit, with sigma function (top) and cutoff function (bottom)

A TLU can classify. Imagine a TLU that has two inputs, whose weights equal 1, and whose theta equals 1.5. When this TLU inputs <0,0>, <0,1>, <1,0>, and <1,1>, it outputs 0, 0, 0, and 1 respectively. This TLU classifies these inputs into two groups: the 1 group and the 0 group. A human brain that knows about logical conjunction (Boolean AND) would similarly classify logically conjoined sentences, this TLU knows something like logical conjunction. This TLU has a geometric interpretation that clarifies what is happening. Its four possible inputs corresponding to four points on a Cartesian graph. From $X_1 * W_1 + X_2 * W_2 = theta$, in other words, the point at which the TLU switches its classificatory behaviour, it follows that $X_2 = -X_1 + 1.5$. The graph of this equation cuts the four possible inputs into two spaces that correspond to the TLU's classifications. This is an instance of a more general principle about TLUs. In the case of a TLU with an arbitrary number of inputs, N, the set of possible inputs corresponds to a set of points in N-dimensional space. If these points can be cut by a hyper plane -- in other words, an N-dimensional geometric figure corresponding to the line in the above example -- then there is a set of weights and a threshold that define a TLU whose classifications match this cut.

2. *Fuzzy Logic (FL)*

Fuzzy Logic [13] was originally conceived in the context of building control systems based on micro-controllers. FL incorporates a simple, rule-based *IF X AND Y THEN Z* approach to a solving control problem rather than attempting to model a system mathematically. The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with temperature control in terms such as "SP =500F", "T <1000F", or "210C <TEMP <220C", terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" or "IF (process is too hot) AND (process is heating rapidly) THEN (cool the process quickly)" are used. These terms are imprecise and yet very descriptive of what must actually happen. Consider what one would do in the shower if the temperature is too cold: one would make the water comfortable very quickly with little trouble. FL is capable of mimicking this type of behaviour but at very high rate.

FL requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them. For example, a simple temperature control system could use a single temperature feedback sensor whose data is subtracted from the command signal to compute "error" and then time-

differentiated to yield the error slope or rate-of-change-of-error, hereafter called "error-dot". Error might have units of degs F and a small error considered to be 2F while a large error is 5F. The "error-dot" might then have units of degs/min with a small error-dot being 5F/min and a large one being 15F/min. These values don't have to be symmetrical and can be "tweaked" once the system is operating in order to optimize performance. Generally, FL is so forgiving that the system will probably work the first time without any tweaking.

In the context of modeling protocol for an Ad Hoc Network, most of the parameters are imprecise or not so-well defined. For example, mobility can be expressed in vague terms by means of a motion vector (precise values will never be known and not essential either). Similarly, distance limitations, power available at the nodes, traffic density etc. are parameters where determination of precise values are not practical and not important either. A fuzzy model helps us to work with imprecise values in a very predictable way.

3. *Genetic Algorithm (GA)*

The basic purpose of genetic algorithms (GAs) is optimization. Since optimization problems arise frequently, this makes GAs quite useful for a great variety of tasks. As in all optimization problems, we are faced with the problem of maximizing/minimizing an objective function over a given space of arbitrary dimension. A brute force which would consist in examining every possible x in X in order to determine the element for which f is optimal is clearly infeasible. GAs give a heuristic way of searching the input space for optimal x that approximates brute force without enumerating all the elements and therefore bypasses performance issues specific to exhaustive search.

We will first select a certain number of inputs, say, $x_1, x_2 \dots x_n$ belonging to the input space X. In the GA terminology,each input is an organism or chromosome. The set of chromosomes is designated as a colony or population. Computation is done over epochs. In each epoch the colony will grow and evolve according to specific rules reminiscent of biological evolution.

To each chromosome x_i, we assign a fitness value which is nothing but $f(x_i)$. Stronger individuals, that is those chromosomes with fitness values closer to the colony optimal will have greater chance to survive across epochs and to reproduce than weaker individuals which will tend to perish. In other words, the algorithm will tend to keep inputs that are close to the optimal in the set of inputs being considered (the colony) and discard those that under-perform the rest.

The crucial step in the algorithm is reproduction or breeding that occurs once per epoch. The content of the two chromosomes participating in reproduction are literally merged together to form a new chromosome that we call a child. This heuristic allows us to possibly combine the best of both individuals to yield a better one (evolution).Moreover during each epoch, a given fraction of the organisms is allowed to mutate. This provides a degree of randomness which allows us to span the whole input space by generating individuals with partly random genes.

Each epoch ends with the deaths of inapt organisms. We eliminate inputs exhibiting bad performance compared to the overall group. This is based on the assumption that they're less inclined to give birth to strong individuals since they have poor quality genes and that therefore we can safely disregard them (selection).

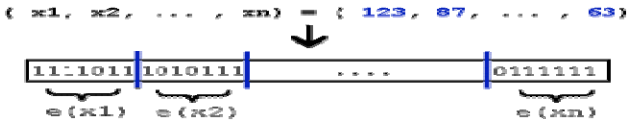
II. ALGORITHM

Based on the conviction regarding the limitless research possibilities, we propose to employ a hybrid form of artificial intelligence that will learn and upgrade itself over a period of time with usage. The proposed components that are used are:

1. Feed-Forward Artificial Neural Networks
2. Fuzzy Logic & Genetic Algorithms

Let's examine in further detail how this whole process is accomplished and how the algorithm works in practice. Let's take the example of optimizing a function f over a space X contained in Nd.

Every input x in X is an integer vector $x = (x_1, x_2, \dots, x_n)$. For the sake of simplicity, assume $0 <= x_i <= k \text{ for } i = 1 \dots n$. In order to implement our genetic algorithm for optimizing f, we first need to encode each input into a chromosome. We can do it by having $log(k)$ bits per component and directly encoding the value x_i. Each bit will be termed *gene*. Of course, we may choose any other encoding based on our requirements and the problem at hand.



At epoch 0, we generate (possibly randomly) an initial set of inputs in X. Then at each epoch i, we perform fitness evaluation, reproduction, mutation and selection. The algorithm stops when a specified criterion providing an estimate of convergence is reached.

1. *Neuro-Fuzzy-Genetic Based Network*

Based on the previous discussion of the three essential ingredients, our ANN acts like a powerful inference engine, drawing all the inference rules from an extensive knowledge base. Our hybrid Neural Network functions with the cooperation of Fuzzy Logic, operating on inputs

(which are fuzzy in nature) and generating a set of solutions in the solution space with minimal searching using Genetic algorithms. A representative schema of the proposed network would look as shown in " Fig. 2,".

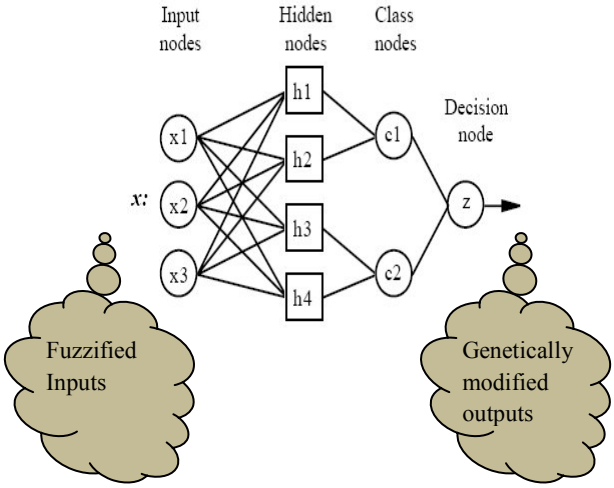


Fig. 2. Neuro – Fuzzy – Genetic Based Network

III. RESULTS

It is important to observe that the objective function of any protocol is to establish a link as quickly as possible, taking into account the various input constraints. Our goal has been to solve the objective function and establish a route within the shortest possible time. Our implementation surpasses traditional routing algorithms by implicitly taking into account all the network input parameters at the same time in reaching an optimal solution. These include: Varying number of nodes in the network, The mobility of the nodes across a geographical region Limitations in the communication capabilities of the nodes, Congested and blocked routes, Nodes that are currently active, Link failure history.

The results generated from the network are independent of the protocol philosophy that one chooses to follow. The chief merit in our implementation is the independency of our simulator to traditional protocol paradigms such 'pro-active', 'reactive', 'power-aware' and similar such principles.

A detailed report of the simulator results for HyperNet has been presented in for various types of standard protocols as well as our own protocol.

HyperNet Simulator has been designed specifically for protocol evaluation of Ad Hoc Networks. Based on the study of various simulators like NS-2 [14] available in the public domain, we designed the features of the simulator that would most aptly reflect the inner workings of an Ad Hoc Network. The following capabilities were incorporated into the simulator: Real Time Node Mobility evaluation , Communication capabilities of the nodes, Monitoring of Inactivity of the nodes and intermittent connections , Power Saving issues during inactive periods, Route optimization based on heuristic and Neuro-Fuzzy Algorithms

For protocol evaluation issues, Back Propagation Artificial Neural Networks with Fuzzy Logic & Genetic Algorithms have been used. These Soft-Computing ingredients make the simulator highly suitable in conducting realistic evaluation of existing and newly-designed protocols.

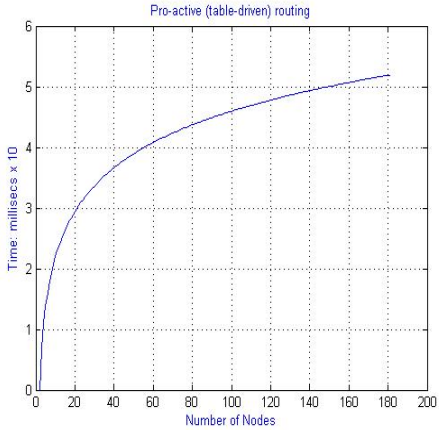
Training sets required to train the ANN were prepared on a rough estimate and these were later refined by generation of intermediate results. These refined training sets were used to train the network towards fine-tuning of the performance.

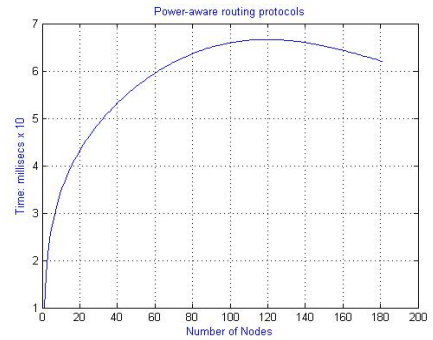
The following parameters were used for evaluation purposes :

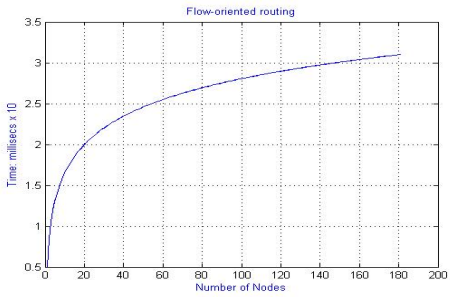
Maximum number of nodes limited to 200,Timeout for on-demand connections programmed for 1200 milliseconds Mobility of the nodes restricted to a radius of 750 meters from initial position, Power status monitoring wherever the nodes are capable of emitting this information, Real Time monitoring of nodes' idling states, Route activity information maintained in a knowledge base, Inference Engine based on Back-Propagation ANN with Fuzzy Logic and Genetic Algorithms, High amount of parallelism within the network activities

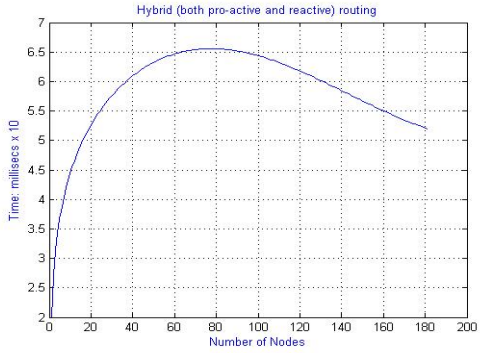
Based on the above parameters, the results of protocol evaluation are being presented here below in the form of graph outputs which are as shown in" Fig. 3,". The graphs are depicted with the number of nodes in the network along the x axis with the routing time between 2 selected nodes represented along the y axis in milliseconds.

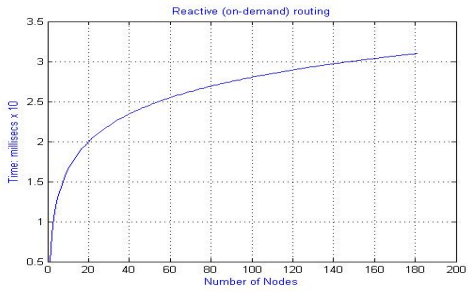
These graph outputs were obtained after 100 iterations through the Neural Networks, reflecting the averaged results in statistical sense. This approach affords maximum reliability in performance.

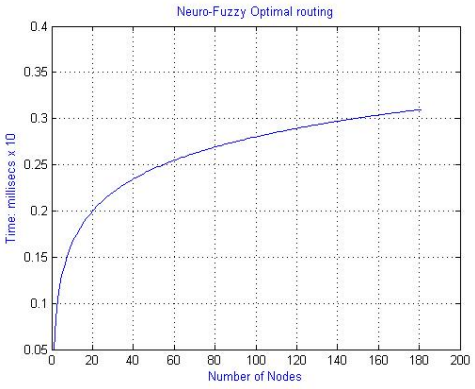












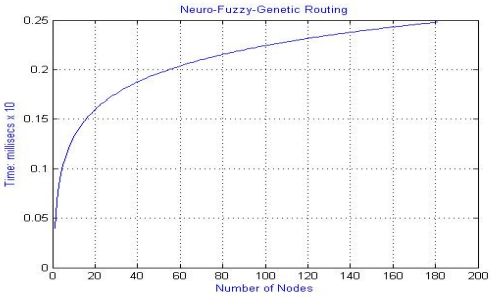


Fig. 3. A plot of Link Establishment time and Number of nodes using various protocols using Hypernet Simulator

TABLE I
LINK ESTABLISHMENT TIME FOR A MAXIMUM OF 200 NODES USING VARIOUS PROTOCOLS USING HYPERNET SIMULATOR

Protocol	Maximum Link Establishment Time in milliseconds
Pro Active (Table Driven) Routing	60
Reactive (On Demand) Routing	35
Flow Oriented Routing	35
Power Aware Routing	70
Hybrid (Pro Active & Reactive) Routing	70
Neuro Fuzzy Optimal Routing	4
Neuro Fuzzy Genetic Routing	2.5

TABLE II
LINK ESTABLISHMENT TIME FOR A MAXIMUM OF 200 NODES USING VARIOUS PROTOCOLS USING NS2 SIMULATOR

Protocol	Maximum Link Establishment Time in milliseconds
Pro Active (Table Driven) Routing	87
Reactive (On Demand) Routing	61
Flow Oriented Routing	62
Power Aware Routing	110
Hybrid (Pro Active & Reactive) Routing	113
Neuro Fuzzy Optimal Routing	24
Neuro Fuzzy Genetic Routing	22

IV. CONCLUSIONS

A few observations on the results can be made:

- The performance of routing is approximately logarithmic in nature and Connection time increases with increase in the number of nodes
- The best performance is to be found in Neuro-Fuzzy and Neuro-Fuzzy-Genetic cases

The superior performance of Neuro/ Fuzzy/ Genetic Routing Algorithms comes with a price – a substantial overhead in preparation of training sets and training the simulator to achieve an equilibrium point. But this is really not an issue as the training phase is a onetime effort.

In conclusion, it appears reasonable to assume that the essential ingredients of ANN with Fuzzy Logic and Genetic Algorithms go a long way in improving the performance of protocol in very dramatic terms.

It can also be seen that by using hypernet simulator and incorporating the neuro- fuzzy- genetic ingredients into it the performance is vastly improved. NS-2 [14] which is widely used simulator has lot of debugs and these are overcome in our simulator. The tables show that there is a remarkable improvement in link establishment time using hypernet. Creating clusters [12] and making cluster head will also minimize the link established time which can also be incorporated into the simulator

REFERENCES

[1] F. Baker, "An outsider's view of MANET," Internet Engineering Task Force document 17 March 2002.

[2] C. Barrett et al., "Characterizing the Interaction Between Routing and MAC Protocols in Ad-hoc Networks," *Proc. MobiHoc 2002* , pp. 92-103

[3] J. Broch et al., "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proc. Mobicom '98.*

[4] D. Cavin et al., "On the accuracy of MANET simulators," *Proc. ACM Workshop on Princ. Mobile Computing (POMC'02),* Oct. 2002, pp. 38-43

[5] K.-W. Chin, et al., "Implementation Experience with MANET Routing Protocols," *ACM SIGCOMM Computer Communications Review,* Nov. 2002, pp. 49- 59..

[6] M. S. Corson et al., "Internet-Based Mobile Ad Hoc Networking," *IEEE Internet Computing,* July-August 1999, pp. 63-70

[7] C. Elliott and B. Heile, "Self-Organizing, Self-Healing Wireless Networks," *Proc.2000 IEEE Int'l Conf. on Personal Wireless Comm*., pp. 355-362.

[8] L. M. Feeney, "A Taxonomy for Routing Protocols in Mobile Ad Hoc Networks," *Swedish Institute of Computer Science Technical Report* T99/07, October 1999

[9] M. Frodigh, et al, "Wireless Ad Hoc Networking: The Art of Networking without a Network," *Ericsson Review,* No. 4, 2000.

[10] David F. Bantz and Fr´ed´eric J. Bauchot. Wireless LAN design alternatives. *IEEE Network,* 8(2):43–53, March/April 1994.

[11] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. MACAW: A media access protocol for wireless LAN's. *In Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications,* pages 212–225, August 1994.

[12] Candy Yiu and Suresh Singh, "A Model for Comparing Rate Adaptation Algorithms", *The Fourth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH 2009) in conjunction with ACM MobiCom 2009*

[13] T.J.Ross " Fuzzy Logic with Engineering applications, McGraw Hill, 1995

[14] An Adaptive Genetic Fuzzy Multi-path Routing Protocol for Wireless Ad-Hoc Networks Hui Liu, Jie Li, Yan-Qing Zhang Yi Pan" *Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05*)