An application of Genetic Fuzzy System in Active Queue Management for TCP/IP multiple congestion networks

Phuong Huy Nguyen, Thi Mai Thuong Duong, Thu May Duong

Abstract— **In this paper, authors proposed a new algorithm to improve the Random Exponential Marking (REM) on TCP/IP network by combining Fuzzy System and Modified Genetic Algorithm (MGA). This new approach allows automatic adjustment of fuzzy parameters according to the network dynamics. Using NS-2 simulator, the results are verified and compared to some traditional AQM algorithms on a multiple-congested TCP/IP network.**

Index Terms— **Active Queue Management, Random Exponential Marking, Genetic Algorithm, Genetic Fuzzy System.**

I. INTRODUCTION

Up to now, TCP/IP protocol has been introduced and developed for more than 50 years. The role of TCP/IP is becoming more and more important, especially in the trend that networks and communication services are converged on the NGN network. In order to provide the best quality of services for different traffic flows on the converged network platform, in addition to improving the TCP/IP performance, there have been studies proposing solutions to support the operations of this topology [1][7].

In the field of traffic management and congestion control, the operations of primitive algorithms for TCP/IP such as slow start, congestion avoidance, fast recovery, fast retransmit do not satisfy the needs in reality because they are algorithms only for controlling congestion on the source side [6]. Therefore, Active Queue Management (AQM) algorithms (running on routers on the TCP/IP network) were proposed aiming at enhancing the capacity of TCP/IP in bandwidth management and congestion control on the network. The nature of these algorithms are to offer a mechanism to actively drop packets such that the network parameters (packet loss rate, mean delay, delay variation and utilization…) are at the most appropriate values.

Currently, there are more than 70 AQM algorithms published and can be devided into three main categories: Queue management based on the queue length, queue management based on the queue load, and queue management based on both the length and the load of the queue [8]. However, the above mentioned approaches for AQM still have two important drawbacks. First, they do not embrace intelligence in maintaining the mean queue length and the fairness in dropping packets for incoming flows on network with dynamic changes. The second drawback is that the selection of parameters for the algorithms still depends on expert knowledge. Additionally, evaluating the performance of AQM algorithms mainly is done on single-congestion network and thus not suitable for real-world TCP/IP networks.

The above drawbacks can be solved by applying soft computing (SC) tools from the computer science field. These tools are to solve approximate problems. Although they are new and still emerging but are earning remarkable achievements. Basically, the model for soft computing is the reasoning of human being. It employs the special characteristics of human reasoning in dealing with problems with uncertainty and inaccuracy based on traditional computing and logic reasoning methods [5]. Because of those reasons, in this paper, the authors propose a model combining fuzzy logic system and modified genetic algorithms for improving the REM_AQM. The proposed approach allows automatically adjust operation parameters according to the dynamics of the network based on training samples without using expert knowledge. The results are verified by using simulation on NS2 tool with the multiple-congestion TCP/IP topology.

II. REM ALGORITHM

REM is a typical AQM algorithm based on queue length and load using one congestion metric called cost which is computed from operation parameters of the system such as packet loss rate, queue length. REM periodically sample the queue of the router and update congestion metrics to reflect the difference between the speed of incoming and outgoing packets at each connection, the difference between real and target value of the queue length. For kth sample of the router queue, congestion metrics p(kT) at time kT is calculated by [2]:

$$p(kT) = \max(0, p(k-1)T + \gamma(\alpha(q(kT) - q_{ref}) + x(kT) - c)) \quad (1)$$

where c is the link capacity (packet speed over time), q(kT) is the queue length and x(kT) is the speed of the incoming packet. The probability of marking or dropping the packet is calculated as follows:

$$prob(kT) = 1 - \phi^{-p(kT)} \quad (2)$$

The parameters for REM are summarized in Table 1.

TABLE I. REM PARAMETERS

Parameters	Description
q_{ref}	Target queue reference for the instantaneous queue.

Huy Nguyen Phuong, Department of Electronics Engineering, Thai Nguyen University of Technology, Thai Nguyen, Vietnam, +84912488515.

Thi Mai Thuong Duong, Department of Computer Science, University of Information and Communication Technology, Thai nguyen, Vietnam, +84945373858.

Thu May Duong, Department of Computer Network, University of Information and Communication Technology, Thai nguyen, Vietnam, +84947105629.

α and γ	Constants for computing the "congestion price"
ϕ	Constant for computing the mark or drop probability
T	Sampling interval for the instantaneous queue

III. FUZZY LOGIC SYSTEM FOR THE REM_AQM PROBLEM

To build a fuzzy logic system for the REM problem, first we have to build a Sugeno fuzzy system for the REM_AQM problem. This fuzzy system is also based on the cost parameter as the REM algorithm. The fuzzy system uses two inputs, one for the sample at the current time and another for the sample at the time of the previous cycle. Based on these two inputs, the fuzzy system will decide the value for Droping Probability (DVP) representing the output of the network. First, we need to the fuzzy sets, and the membership functions. The fuzzy sets for the input parameter can be defined as follows:

$$p_l(t+1) = \{NLS, NVL, NL, NS, ZO, PS, PL, PVL, PLS\} \quad (3)$$

Similar for $p_l(t)$:

$$p_l(t) = \{NLS, NVL, NL, NS, ZO, PS, PL, PVL, PLS\} \quad (4)$$

Equation (5) is used to compute the values for the samples in cycles at the current time:

$$p_l(t+1) = \left[p_l(t) + \gamma(\alpha_l(b_l(t) - b_l^*) + x_l(t) - c_l(t)\right]^+ \quad (5)$$

Similar for $p_l(t)$:

$$p_l(t) = \left[p_l(t-1) + \gamma(\alpha_l(b_l(t-1) - b_l^*) + x_l(t-1) - c_l(t-1)\right]^+ \quad (6)$$

Where $\gamma > 0$ and $\alpha_l > 0$ are constants with small values, operator $[Z]^+ = \max(Z,0)$. $b_l(t)$ is the sum of usage time for queue l in period t and $b_l^* \geq 0$ is the length of the target queue, $x_l(t)$ is the sum of speed of the input of queue l in period t and $c_l(t)$ is the available bandwidth of queue l in period t. $x_l(t) - c_l(t)$ shows the difference in speeds and $b_l(t) - b_l^*$ shows the difference in lengths of the buffer. The set of output values of the probability DVP is determined by:

$$DVB = \{mf1, mf2, mf3, mf4, mf5, mf6, mf7, mf8, mf9\}$$

Membership functions for inputs of the fuzzy system are given in Figure 1. We use triangular membership functions for the ease of computation and we can change to suit the designer. The structure of the fuzzy rules includes if-then rules which are the foundation for making decisions of the fuzzy reasoning. These fuzzy rules are designed independently and can be changed for better performance. For example:

if pr_{t+1} is NLS and pr_t is PL then DVP is mf5

These fuzzy rules are presented on the reasoning surface and make up a fuzzy reasoning mechanism.

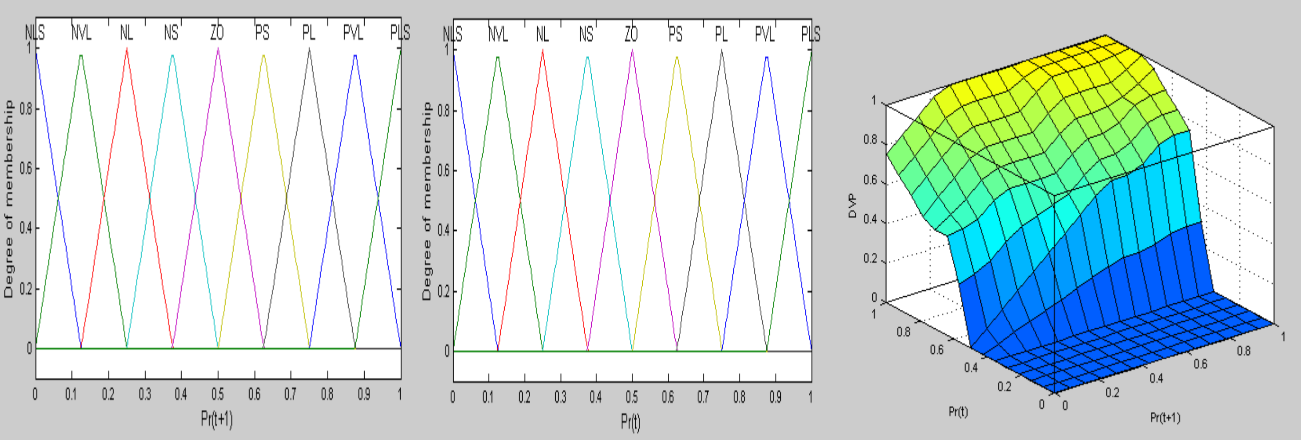

Figure 1. Membership functions for fuzzy inputs $p_l(t+1)$, $p_l(t)$ *and the reasoning curved surface*

IV. MODIFIED GENETIC ALGORITHM MGA FOR ADJUSTING THE FUZZY SYSTEM

The adjustment of the fuzzy system designed in the previous section is represented in Figure 2. The data used for making adjustment are 100 reprehensive samples for the two fuzzy inputs and output taken within 10 seconds. GA for optimization progress is as follows:

Step 1: Encoding and initializing a population of chromosomes.

Step 2: Searching for the fitness function and identifying the fitness values of each chromosome using the fuzzy system.

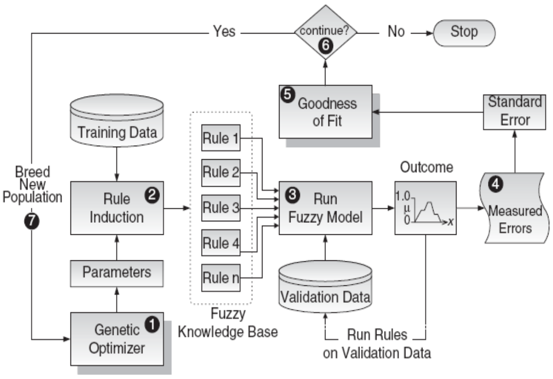

Figure 2. Adjustment model for the fuzzy system by using GA

Step 3: Evaluating the convergence after each generation. If the convergence is not satisfying, the chromosomes will be copied basing on their fitness values (selection) and generate new ones by genetic operations (crossover and mutation), then return to step 2. In contrast, the process should be finalized to come up to a report of the results.

TABLE II. PARAMETERS OF GA

Parameters	Description
Population size	20
Number of generations	30
Encoding	Real
Length of chromosome	63
Crossover probability	0.46
Mutation probability	0.008

Encoding: is to represent unidentified variables by a series of chromosome. Each variable represents a gene. To reduce the training time, MGA algorithm will perform genetic operations using real numbers encoding. Matlab is used to program the training for the fuzzy control without using the integrated toolbox of GA. Each chromosome contains 63 genes of which 54 are input parameters and 9 are predictive output parameters. The range of space of input parameters stays within [0,1].

Fitness function: reflects the natural selection process based on a specified fitness level. Regarding the adjustment of future combination models, the changes in the system are reflected by its fitness level; while the fitness level of each individual is calculated as follows:

$$f_m(k) = \exp(-[(\Delta\varepsilon(k)/e(k)) - 1]^2 \qquad (7)$$

with *e(k)* as the difference between theoretical and actual outputs (the length of the queue);

Δε(k)= e(k) - e(k-1) as the difference (error) between two generations.

The selection of fitness functions according to (7) transforms the relation of the difference-based fitness function to an exponential function; thus, boost the convergence of the system and shorten the convergence time of the algorithm.

Selection: is the combination of Roulette wheel and fixed selection. To a specified generation in this method, there is a particular number of chromosomes with better fitness values among the preceding populations are accepted to the following population. This is to ensure an effective scheme when there is a complicated and non-linear search space.

Crossover: The real-number-based crossover is performed. A pair of mating chromosome exchange a sub-set of its components at an integer kth position selected according to (8). Two new chromosomes are generated due to the exchange of genes at the positions of k+1 and L (with L as the length of the chromosome). Note that this does not affect the value of gene within the chromosome [3].

$$C_r = ROUND[F_{fit}(i,j) \times L] \quad \in [0.....L] \qquad (8)$$

ROUND(.) here is to determine the closest satisfying integer number. As can be seen from (8), the individual of small fitness level is situated at a position of great weight, leading to offsrprings to move to a different region with a big hop. The greater the fitness level of the individual is, the closer to the binary digits in the smallest weight the crossover point moves. This results in a non-remarkable changes in the next generations.

Mutation: happens to individuals by the crossover operation with the probability *Pm*. The mutation operation brings random changes to the chromosome components in the new population. In real-number encoding GA, it is simply adjusted by replacing mutant genes with a random number selected from the same range assigned to each "gene". Such process generates new individuals. To avoid local optimal, each individual is randomly varied with the *Mr* mutation positions as follows:

$$M_r = ROUND[(L - C_r) \times M_b / L] \quad \in \ \{0...M_b\} \qquad (9)$$

With *Mb* as the upper limit of the mutation positions. The mutation probability depends on both the crossover position and the assigned mutation position. When the individual's fitness level remains low, the mutation position is put in the position of most significant weight, and the mutation probability is high. This is to extend the search space. Chronologically, the fitness level increases, followed by a gradually faster move to the position of less significant of mutation crossover to make the algorithm converged. The evolution continues until it reaches the expected level of fitness. After the adjustment of the fuzzy system by the algorithm through the aforesaid steps for training the network with parameters shown in Table 2, a set of optimal parameters lying within the highest fitness chromosome is produced, from which it is possible to identify the accurate structure of the fuzzy system with values of the membership input functions and the curved surface inferred from Figure 3.

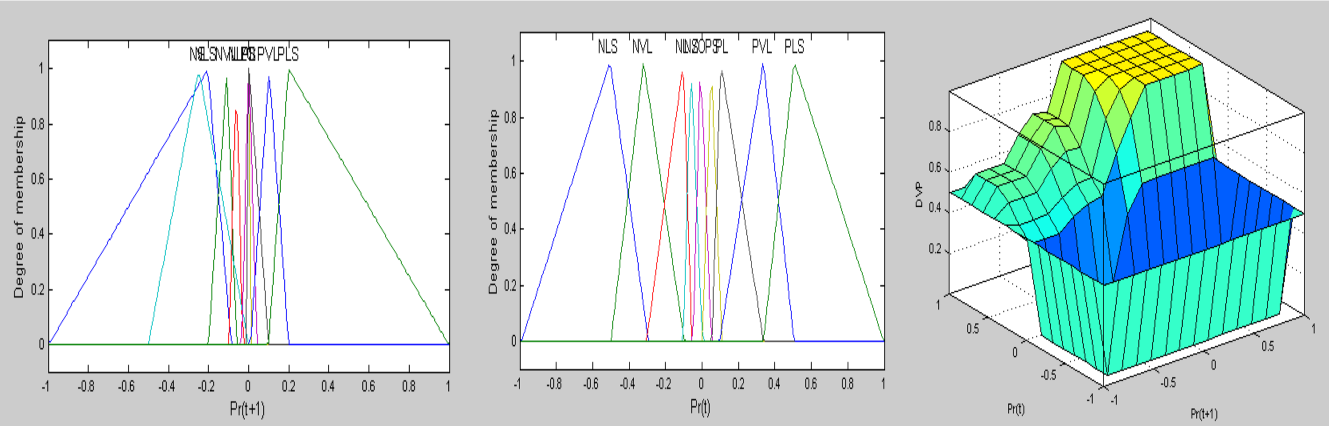

Figure 3. Values of the membership input function and the inferred curved surface after training

V. SIMULATION RESULTS FOR EVALUATING AQM ALGORITHMS

In this section, the authors use NS-2 simulator to study the performance of the proposed AQM in comparison with other methods such as REM, PI, ARED. Controlling values of parameters in AQM is established by the proposals by authors in [2] and [4]. This is to produce a fair comparison that uses the establishments proposed by those authors. The network configuration for the simulation is presented in Figure 5. With the use of TCP/Newreno, AQM algorithm within the queue of all core links from Router A to outer F is set up. Simple Droptail is used for all other access links. Traffic and delay are as follows: (C1, d1) = (C8, D8) = (C9, d9) = (100Mbps, 5ms), (C2, d2) = (C4, d4) = (C6, d6) = (15Mbps, 10ms), (C3, d3) = (15Mbps, 60ms), (C5, d5) = (15Mbps, 30ms), (C7, d7) = (C10, d10) = (C11, D11) = (200Mbps, 5ms). That Flow N1 ends at destination 1, Flow N2 ends at destination 2 and Flow 3 ends at destination 3 generate a cross traffic. The number of flows is initially set up as N1=100, N2=50 and N3=100. Next, the number of sources is made up to N1=500, N2=200 and N3=300. The traffic is changed by pausing a half of sources at 40 s and resuming at 70 s to measure the response of the algorithms.

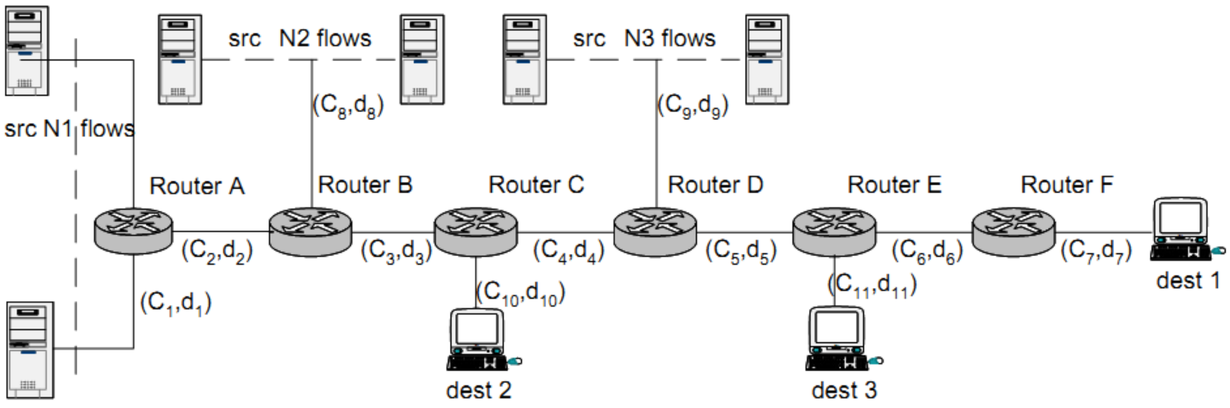

Figure 4. The network topology for simulation

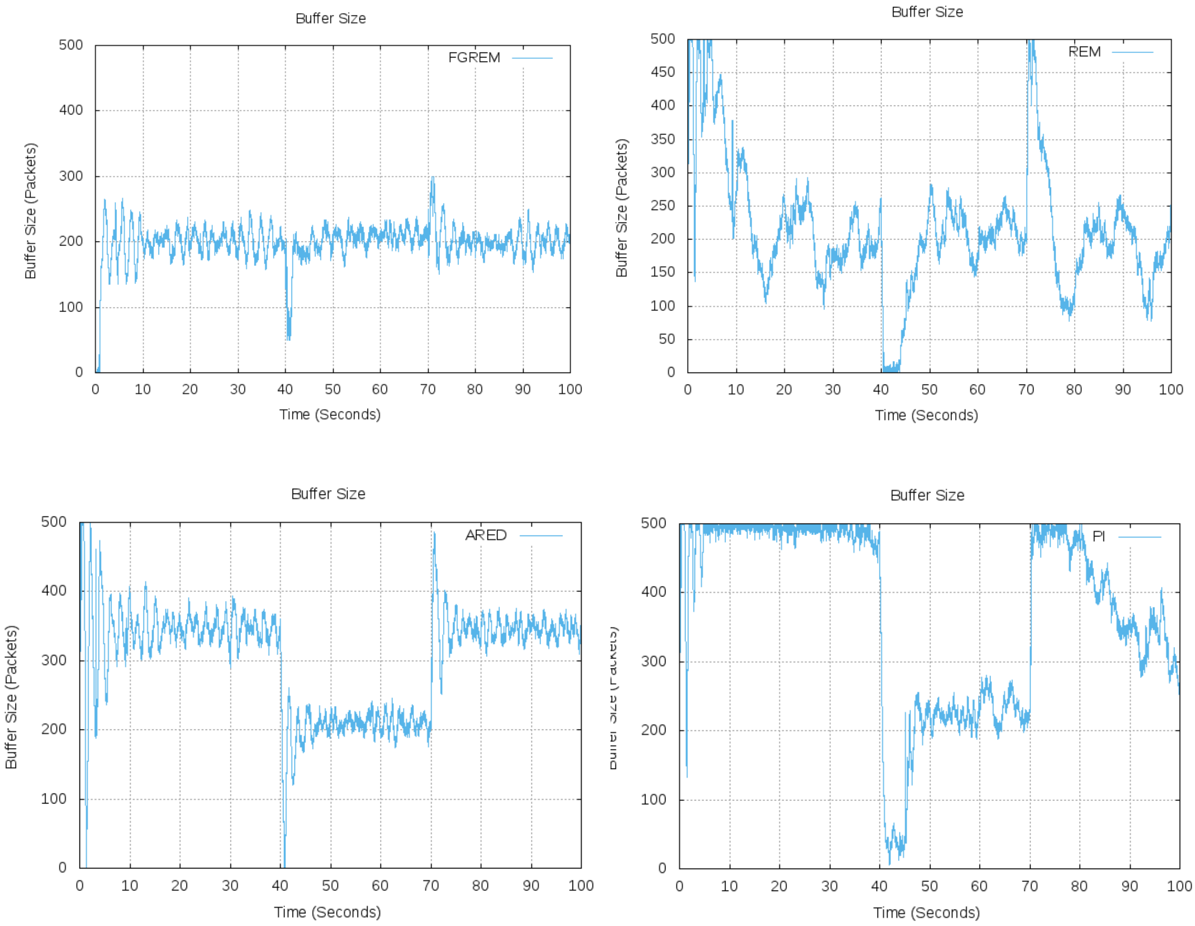

Figure 5: Queue lengths of AQM schemes at N=800,RTT=30

Figure 5 indicates the average length of the queue corresponding to the alternately AQM algorithms from left to right and downward as FGREM, REM, ARED and PI. It is clearly that FGREM control proves to be more effective than any other. Within the first 10 seconds, the variation of the mean length of the queue corresponding to other AQM algorithms remains high especially ARED and REM). However, for FGREM, the variation remains smallest and only extends for the first 3 seconds due to its parameters being taken from preceding research models. Furthermore, within 100 seconds of simulation, FGREM's mean length of queue is most stable. Specifically, when the traffic load changes at 40s and 70s, the variation of FGREM's mean length of the queue remains smallest. This result is contrasting to that from conventional AQM algorithms such as PI, ARED and REM.

Figure 6 illustrates the changes of the parameters during the operation of the TCP/IP network with the number of inputs ranging from 250 to 800. From left to right in turn are packet loss ratio, the utilization, mean queuing delay and delay variation compared to traffic load. When the traffic load increases, FGREM was possible to reach the maximum utilization of the route, along with minimum packet loss ratio and smallest delay variation, due to FGREM can still maintain the queue around a specified target values. Other AQM algorithms present a slow reaction in adjusting the queue; thus, show less effectiveness such as: ARED produces highest packet loss ratio and lowest utilization, REM produces high delay and delay variations, both of which affects the quality of services.

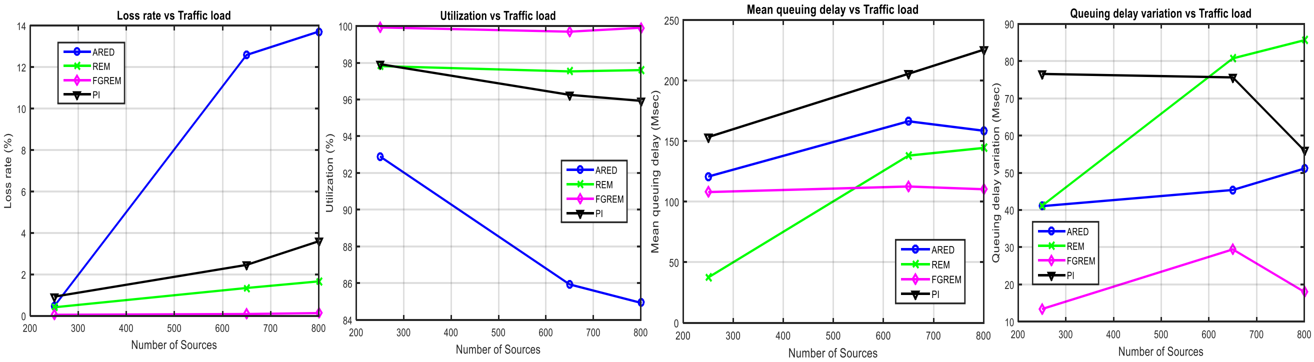

Figure 6. The changes of network parameters when the number of loads chanae

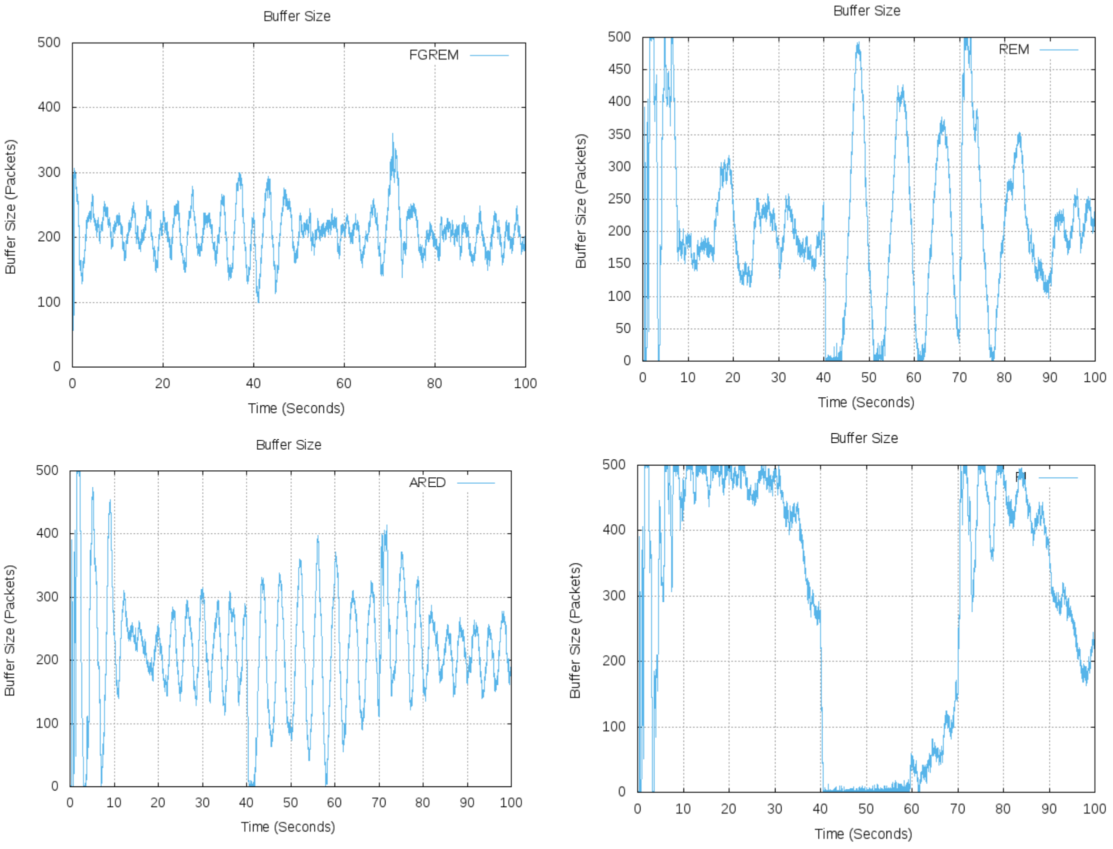

Figure 7. Queue lengths of AQM schemes at N=800, RTT=200

Next, an investigation into the operations of AQM algorithms is performed through delay transmissions linked to the bottlenecks. Vary the RTT values from 30 ms to 120 ms then 200 ms. The mean lengths of the queue are presented in Figure 7. The changes of operating parameters are described in Figure 8.

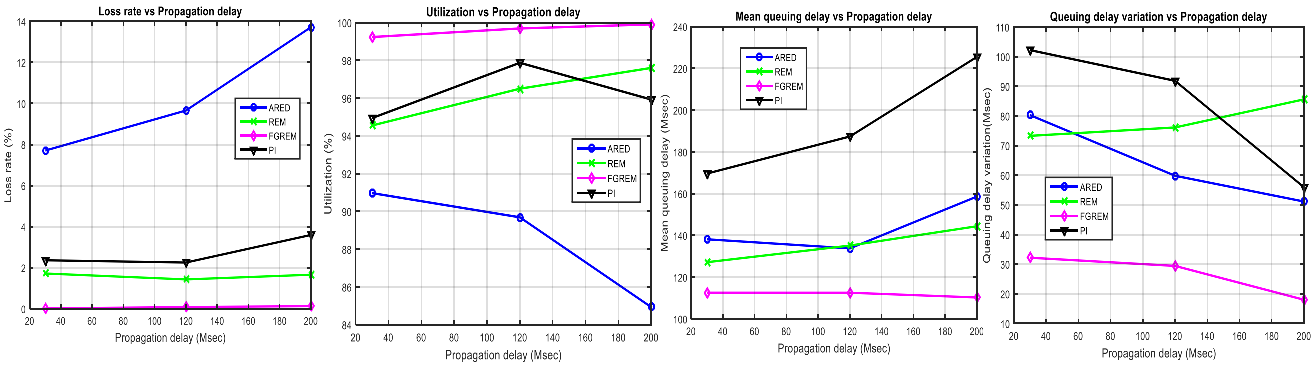

Figure 8. The changes of network parameters when RTT change

As is illustrated by Figure 7, the mean length of the queue sharply fluctuates when RTT value changes. However, FGREM least fluctuates, thus, suffers least packet loss ratio (see Figure 8). The result shows that although FGREM does maintain the minimum mean, it possesses minimum delay variation. On probability of packet dropping, ARED and PI proves to be least effective due to its basis on the length of the queue which is an indirect element; followed by REM due to its basis on incoming speed of traffic streams.

VI. Conclusion

The article proposed a new model of genetic fuzzy system to improve the AQM algorithm based on the queue length and load. Naturally, this is an improvement of REM algorithm presented in [2]. REM algorithm for the management of the queue is based on a parameter called "cost" which is the combination between the length of the queue and the load speed. However, the parameters are constant and dependent on expert knowledge. On the other hand, these parameters are non-updated upon the network dynamic status; thus, they are not adaptive to the network's real circumstances. FGREM solves the mentioned drawbacks. It is advisable to build the structure of Sugeno fuzzy system to design genetic fuzzy system; followed by GA with real values to assist the adjustment of the fuzzy system. The operation of the system with AQM algorithm has been reviewed by conventional method namely REM, ARED, PI. Proposed parameters through simulations in published results demonstrate a good performance of FGREM algorithm. The length of referred queuing is produced in a short period of time and rarely affected by network circumstance variations such as changes in traffic, the number of inputs, and RTT.

References

[1] Aoyama. T, A new generation network: Beyond the Internet and NGN, *Communications Magazine, IEEE*, Volume: 47 (5): 82 – 87, 2009.

[2] Athuraliya S., Lapsley D. E., Low S. H, Random early marking for Internet congestion control. *IEEE/ACM Transactions on Networking*, Vol. 15(3): 48-53, 2001.

[3] Cong Huu Nguyen, Thanh Nga Thi Nguyen, Huy Phuong Nguyen, Research on the application of genetic algorithm combined with the "cleft-overstep" algorithm for improving learning process of MLP neural network with special error surface. *The 7th International Conference on Natural Computation and the 8th International Conference on Fuzzy Systems and Knowledge Discovery*, Vol 2: 222-227, 2011.

[4] Chrysostomou. C., Fuzzy Logic Congestion Control in TCP/IP Tandem Networks , *ISCC'06 Proceedings 11th IEEE Symposium on Computers and Communications*, pp.123 – 129, 2006.

[5] Jyh Shing Roger Jang, Chuen Tsai Sun, Eiji Mizutani, *Neuro fuzzy and Soft Computing*, Prientice Hall International, Inc, 2002.

[6] S. H. Low, F. Paganini, and J. C. Doyle, Internet congestion control, *IEEE Control Systems Magazine, Vol.22(1):28-39,2002.*

[7] *Sapna Bagde, Poonam Sinha,* Ashish Jain, Survey of Performance based Transmission Control Protocol in MANET, *International Journal of Computer Applications & Information Technology*, Vol. 2(1): 9-16, 2013.

[8] Thiruchelvi, G. and J. Raja, A survey on active queue management mechanisms, *Int. J. Comput. Sci. Network Secur.,* Vol 8: 130-145, 2008.

Dr. Phuong Huy Nguyen is PhD in Telecommunication Engineering at Hanoi University of Technology, Vietnam. He is working at Faculty of Electronics Engineering, Thainguyen University of Technology, Thai nguyen City, Vietnam. Research interests: Digital Signal Processing, Soft computing, Fuzzy control.

MSc Thi Mai Thuong Duong is working at Faculty of Information Technology, University of Information and Communication Technology, Thai nguyen City, Vietnam. Research interests: Computer science.

MSc Thu May Duong is working at Faculty of Information Technology, University of Information and Communication Technology, Thai nguyen City, Vietnam. Research interests: Computer science, Computer networking