Fuzzy Active Queue Management for Congestion Control in Wireless Ad-Hoc

Essam Natsheh, Adznan B. Jantan, Sabira Khatun, and Shamala Subramaniam
Department of Computer and Communication Systems, University Putra Malaysia, Malaysia

Abstract: *Mobile ad-hoc network is a network without infrastructure where every node has its own protocols and services for powerful cooperation in the network. Every node also has the ability to handle the congestion in its queues during traffic overflow. Traditionally, this was done through Drop-Tail policy where the node drops the incoming packets to its queues during overflow condition. Many studies showed that early dropping of incoming packet is an effective technique to avoid congestion and to minimize the packet latency. Such approach is known as Active Queue Management (AQM). In this paper, an enhanced algorithm, called Fuzzy-AQM, is suggested using fuzzy logic system to achieve the benefits of AQM. Uncertainty associated with queue congestion estimation and lack of mathematical model for estimating the time to start dropping incoming packets makes the Fuzzy-AQM algorithm the best choice. Extensive performance analysis via simulation showed the effectiveness of the proposed method for congestion detection and avoidance improving overall network performance.*

Keywords: *Active queue management, ad-hoc networks, fuzzy systems, intelligent networks, network congestion.*

Received September 24, 2005; accepted March 28, 2006

1. Introduction

Mobile ad-hoc network is a network without infrastructure where every node can work as a router. Every node has protocols and services to request and provide services to other nodes with the congestion handling capability. Traditionally, the congestion handling is done through Transmission Control Protocol (TCP). This protocol sends congestion signal (drop incoming packets) when the node's queue is full (queue length is maximum). Some studies [4, 11] showed that early dropping of incoming packet before reaching the maximum queue length is an effective technique to avoid congestion and to minimize the packet latency, e. g., Active Queue Management (AQM) drops incoming packets before the queue is full in contrast to traditional queue management which starts dropping only when the queue in overflowed.

Mobile ad-hoc networks suffer high network congestion due to high Bit Error Rate (BER) in the wireless channel, increased collisions due to the presence of hidden terminals, interference, location dependent connection, uni-directional links, frequent path breaks due to mobility of nodes and the inherent fading properties of the wireless channel [20]. This substantiates the need for high adaptive AQM algorithms with adapting capabilities to high variability and uncertainty for these types of networks. The proposed fuzzy logic based AQM, called Fuzzy-AQM, is such types of algorithms to overcome the above shortcoming in ad-hoc networks. The application of fuzzy logic to the problem of congestion control allows us to specify the relationship between queue parameters and packets dropping probability using “if...then...” type of linguistic rules. The fuzzy logic algorithm would be able to translate or interpolate these rules into a nonlinear mapping.

In this study, the focus is to investigate the impact of the traditional and Fuzzy-AQM algorithms on the ad-hoc network. The considered strategy is as follows: When the congestion is detected, the node uses one of the AQM policies to drop the incoming data packets. Meanwhile, it allows the control packets to pass to the queue using Drop-Tail policy. Therefore, the data packets are dropped first when the packets drop probability exceeds a certain threshold while the control packets are still acceptable until the queue is full.

Control messages are preferred to pass to the queue during congestion time for the following reasons:

1. Control messages are used to update the changes of the network topology. Therefore, they prevent data packet to be transmitted through broken paths.
2. Data packets are “connection oriented”, that is, guaranteed delivery to their destinations by TCP. In contrast, control messages are “connectionless”; that is, the dropped message will not be retransmitted again.
3. Control message size is very small compared to data packet. Normally in ad-hoc routing protocols, control message size is 64 bytes while data packet is 512 bytes, i. e., the control message takes small space in the queue and fast processing time in the node.

The rest of this paper is organized as follows. Section 2 summarizes related work on the common AQM polices issues and focuses on previous implementations of fuzzy AQM policies. Followed by congestion in ad-hoc networks, the fuzzy dropping algorithm as a new AQM policy (Fuzzy-AQM), performance analyzes of the proposed algorithm, and finally the conclusions.

2. Related Work

The most famous AQM algorithm is Random Early Detection (RED) [11]. The RED algorithm manages the queue in an active manner by randomly dropping packets with increasing probability as the average queue size increases. It maintains two thresholds that determine the rate of packet drops: A lower threshold (denoted by min_{th}) and an upper threshold (denoted by max_{th}). For each packet k arrives to the queue, the drop probability for that packet $p_d(k)$ is given by:

$$p_d(k) = \begin{cases} 0 & \text{if } q_c < \min_{th} \\ \dfrac{avg - \min_{th}}{\max_{th} - \min_{th}} \max_p & \text{if } \min_{th} \le avg < \max_{th} \\ 1 & \text{if } avg \ge \max_{th} \end{cases} \tag{1}$$

Where q_c is current queue size, avg is current average queue size and max_p is maximum drop probability.

Some previous studies showed the difficulties of choosing the RED parameters [13, 18, 19]. Other studies showed that there is no significant benefit to RED over Drop-Tail for the web traffic [5, 6, 13]. Those drawbacks are the main reasons to default disable of the RED function (or some vendor-specific variant of RED, e. g., Cisco's Weighted RED (WRED) [9]) in most of the available routers today. To overcome these drawbacks, extensions of the RED algorithm had been proposed to make it more robust and/or adaptive, for example, Stabilized RED (SRED) [22], Flow RED (FRED) [3], Dynamic RED (DRED) [16] *etc*. The most famous dynamic configured RED is the Adaptive RED (ARED) algorithm proposed by Floyed *et al*. [12]. In ARED, the max_p is configured dynamically to keep the average queue size avg within a target range.

Many studies used the fuzzy logic system to dynamically calculate the drop probability behavior of AQM policy. Wang *et al*. [30] proposed Adaptive Fuzzy-based RED (AFRED) algorithm to calculate the drop probability using the current queue size as the only input for the fuzzy system. Some other studies calculate the drop probability based on Fuzzy Explicit Rate Marking (FERM) algorithm using two queue state inputs: The current queue size 'q_c' and its rate of change 'Δq_c'. The FERM was implemented in [25] for ATM networks, while in [8, 28] it was implemented for differentiated services (Diff-Serv) networks.

In [1, 2, 7, 10, 17, 27, 32], the authors calculate the drop probability using Fuzzy Proportional Derivative Controller (FPDC) with two inputs: The error 'e' (which is the difference between the current queue size and the desired queue length) and the change of the error 'Δe' (which is the difference between the current error and the previous error). A conventional fuzzy controller use (e, Δe) as inputs to observe the controlled system response and its parameters. These parameters are overshoot, rise-time and settle-time. This set of parameters is not only used to evaluate the stability, but the performance of a system as well, and often is given in specification. Using the same inputs (e, Δe) to calculate the drop probability of AQM is meaningless and the fuzzy "if...then..." rules will not accurately represent the queue system behavior.

Li *et al*. [15] have used the current average queue size 'avg' and its variance 'Δavg' as the input for the Fuzzy Logic Adaptive RED (FLARED) algorithm to adaptively modifying the changes of step-size of the parameter max_p. This scheme tune only one parameter of ARED algorithm and its drawback is the lack to tune other ARED parameters.

In this study, we have used fuzzy logic system to calculate the drop probability in ad-hoc networks using: The current queue size and the number of neighboring nodes. This scheme can be generalized to be used in any network where the number of neighbors' nodes represents the number of communication links, or precisely number of TCP sessions. Table 1 compares various schemes to design fuzzy AQM algorithms.

Table 1. The fuzzy AQM schemes.

Fuzzy AQM Scheme	Congestion Metric	Optimized Variable
AFRED [30]	Current queue size 'q_c'	Drop Probability
FERM [8, 25, 28]	q_c and its change 'Δq_c'	Drop Probability
FPDC [1, 2, 7, 10, 17, 27, 32]	Error 'e' and its change 'Δe'	Drop Probability
FLARED [15]	Average queue size 'avg' and its change 'Δavg'	Δmax_p
Our scheme: Fuzzy-AQM	q_c and node neighbors density	Drop Probability

3. Congestion in Ad-Hoc Networks

In ad-hoc networks, congestion control is handled through transport layer protocols. The connection-oriented transport layer protocol used in ad-hoc networks is Transmission Control Protocol (TCP) [14]. The objectives of this protocol include the setting up of an end-to-end connection, end-to-end delivery of data packets, flow control and congestion control. TCP uses window-based flow control mechanism. The sender maintains a variable size window whose size limits the number of packets the sender can send. The destination sends ACKnowledgment (ACK) for packets that are received. When the window size is exhausted, the sender must wait for an ACK before sending a new packet based on a sliding window principle. This waiting time is known as Retransmission TimeOut (RTO) period. If the ACK does not arrive within the RTO period, then the sender will assume the packet is lost. The loss of packet is due to the congestion in the network which will yield TCP to start the congestion control mechanism.

Mobile ad-hoc networks experience dynamic changes in the network topology due to unrestricted mobility of nodes. The topology changes lead to frequent changes in the connectivity of wireless links and hence routes reestablishment may be repeated very often. This route reestablishment process takes a significant amount of time. The route reestablishment time is a function of transmission range of the nodes, distance between the source and destination, number of intermediate nodes between the source and destination and node's velocity. If the route reestablishment time is greater than RTO period of the source node, then it will not receive the ACK and assumes congestion in the network, followed by retransmission of the lost packets and initiation of the congestion control mechanism [20]. A schematic illustration of congested ad-hoc network is shown in Figure 1. The source sends its data packets through node A, which passes those packets to node B then to the destination. As soon as the link between the source and node A is broken, it starts route reestablishment process and creates a direct link with node B. If this processing time is less than RTO, the source will receive the ACK and send other data packets, or it will resend the previous lost packets.

4. Fuzzy-AQM Algorithm

In this section, concepts and rules of the proposed Fuzzy-AQM algorithm for ad-hoc networks are introduced. In the following two subsections, we studied the effect of some node parameters on packets drop probability. These parameters are used in subsection C to create the rules of the fuzzy system. Method to design their membership functions is presented in the later subsection. Overall system design and its implementation complexity are presented in subsection E and F. Compatibility of the proposed algorithm with other conventional algorithms discussed in the last subsection.

4.1. Effect of q_c on Drop Probability

Current queue size q_c is the most used indicator in AQM policy for estimating the probability of dropping the incoming packets. The drop probability p_d can be calculated as [26]:

$$p_d = \frac{2N^2}{\left(CT_p + q_c\right)^2} \tag{2}$$

Where N is a load factor, C is a transmission capacity (in packets/seconds) and T_p is a propagation delay (in seconds). Assuming a 10 Mbps (2500 packets/sec) transmission capacity with a 100 msec propagation delay, Figure 2 shows the relation between the drop probability and the load for various queue sizes. It is evident that the probability of a packet dropping increases as the load increases. More packets in the queue wait for processing as load increases. Thus, it can be stated that when the used space of the queue is high, the drop probability of incoming packets is also high and vice versa. Consequently, the following rules are proposed:

R1: If q_c is low then p_d ought to be low.
R2: If q_c is medium then p_d ought to be high.
R3: If q_c is high then p_d ought to be high.

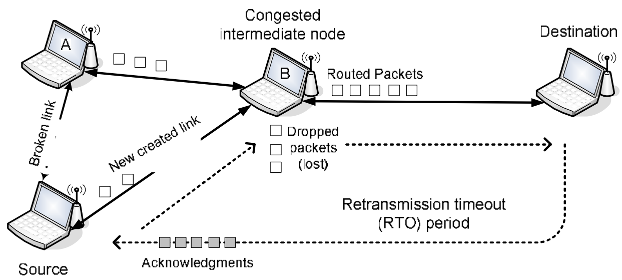


Figure 1. Congestion in ad-hoc networks.

4.2. Effect of Node Neighborhood Density on Drop Probability

In ad-hoc networks, the traffic is categorized as: Data packets and control messages. The control messages are used to continuously update the nodes about the topology changes (new created or lost links). For example, if a node has two neighbors that means it will receive two hello messages every second from them. Besides, receiving a route request messages, a route breaks messages, or data packets. If that node has ten neighbors, this means it will receive, in every second, ten hello messages beside bulk amount of control messages and data packets. Hence, it is clear that the traffic pass through the nodes with few neighborhoods is less than the others with many neighbors. In equation 2, the load N can be written as:

$$N = \sum_{i=0}^{n} \lambda_i \tag{3}$$

Where λ_i denote flow's rate from the neighbor node i and n is the number of neighbors. The congestion will happen at:

$$p_d = 1 \qquad if \ \sum_{i=0}^{n} \lambda_i > C \ and \ q_c = q_m \tag{4}$$

Where q_m is the maximum queue size. Hence, if the neighbors' density of a node's is high, the node's queue will be full quickly and increases the probability of congestion and vice versa. Consequently, the following rules are proposed:

R4: If neighbors' density is low then p_d ought to be low.
R5: If neighbors' density is medium then p_d ought to be high.
R6: If neighbors' density is high then p_d ought to be high.

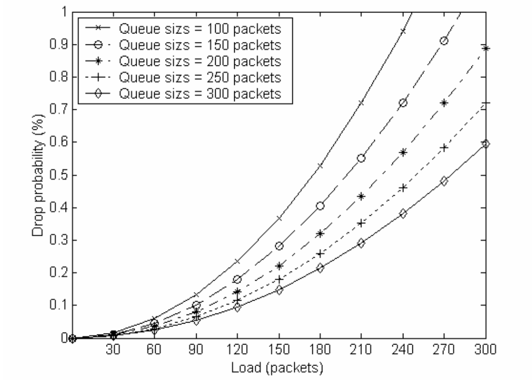


Figure 2. Drop probability for the coming load.

4.3. The Rule-Base for Fuzzy Drop Probability

To fulfill the fuzzy sets theory, the previous six rules (R1 to R6) can be combined within a 2-dimensional rule-base to control the drop probability adaptively as presented in Table 2. For example, according to Table 2 the first rule is:

If q_c is *Low* and neighbors' density is *Low* then p_d is *Low*

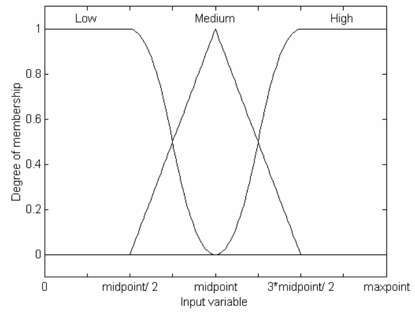
Table 2. Fuzzy-AQM rules for drop probability.

		Neighbors' Density		
		Low	**Medium**	**High**
q_c	**Low**	Low	Low	Low
	Medium	Low	High	High
	High	High	High	High

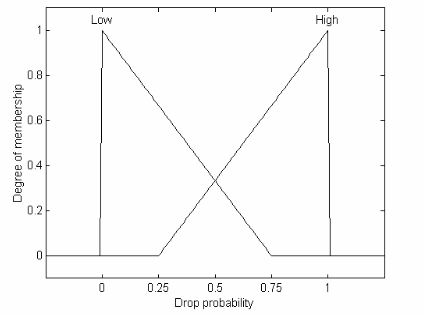
4.4. Membership Functions For Fuzzy Variables

After defining the fuzzy linguistic 'if-then' rules, the Membership Function (MF) corresponding to each element in the linguistic set should be defined. For example, if the queue size is 5 k bytes and q_c equal to 2 k bytes, using conventional concept, it implies q_c is either 'low' or 'medium' but not both. In fuzzy logic, however, the concept of MFs allows us to say the q_c is 'low' with 80% membership degree and 'medium' with 20% membership degree.

The MFs we propose to use for the fuzzy inputs (q_c, neighbors' density) and the fuzzy output (p_d) are illustrated in Figure 3. These MFs are used due to their economic value of the parametric and functional descriptions. In these MFs, the designer needs only to define one parameter; *midpoint*. These MFs mainly contain the *triangular* shaped MF [23]. The remaining MFs are as follows: Z-shaped membership to represent the whole set of low values and S-shaped membership to represent the whole set of high values.



(a) MFs used for the input variables.



(b) MFs used for the output variable.

Figure 3. Membership functions used for the fuzzy variables.

Maxpoint is the maximum queue size in q_c–MF (Table 2), and it is the number of the network's nodes in the neighbors' density MF. *Midpoint* of q_c–MF is a threshold that indicates whether the queue is going to be full soon. The threshold is simply set to 60% of the queue size. The optimal value for this variable depends in part on the maximum average delay that can be allowed by the nodes. Tseng *et al.* [29] argue about the cost-effectiveness to have large ad-hoc networks. They proved by simulation that practical sizes of ad-hoc networks would range within about five nodes. Therefore, for neighbors' density MF, *midpoint* should be equivalent to five nodes.

4.5. Fuzzification, Inference and Defuzzification

The fundamental diagram of the fuzzy system is presented in Figure 4. Fuzzification is a process where crisp input values are transformed into membership values of the fuzzy sets (as described in the previous section). After the process of fuzzification, the inference engine calculates the fuzzy output using the fuzzy rules described in Table 2. Defuzzification is a mathematical process used to convert the fuzzy output to a crisp value; that is, p_d value in this case.

There are various choices in the fuzzy inference engine and the defuzzification method. Based on these choices, several fuzzy systems can be constructed. In this study, the most commonly used fuzzy system, *Mamdani* method, is selected; for further details on this system see [31].

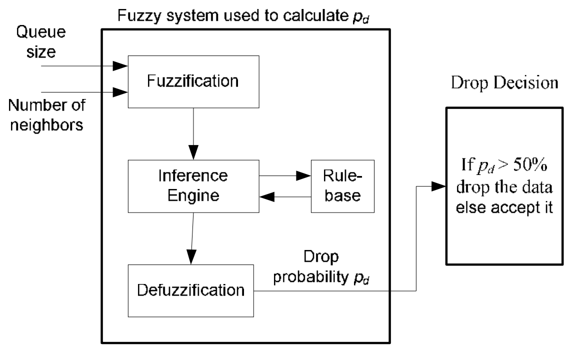


Figure 4. Block diagram for the basic elements of the fuzzy-AQM.

4.6. Implementation Complexity

Using fuzzy logic system with AQM, we may achieve comparable or better run-time computation than purely conventional methods. This can be achieved using *lookup table*. The input-output relationship of the fuzzy reasoning engine for Fuzzy-AQM is illustrated in Figure 5. This relationship can be stored as a lookup table which will result in a very fast execution.

5. Performance Analysis of the Proposed Fuzzy-AQM

5.1. Simulation Environment

Simulation of the proposed AQM design was done using *OMNeT++* version 2.3 with *Ad-Hoc simulator* 1.0 [21]. The *OMNeT++* is a powerful object-oriented modular with discrete event simulator tool. Each mobile host is a compound module which encapsulates the following simple modules: An application layer, a routing layer, a MAC layer, a physical layer, and a mobility layer.

- *Application Layer:* This module produces the data traffic that triggers all the routing operations. In all scenarios, 15 nodes are enabled to transmit. The traffic is modeled by generating a packet burst of 64 packets sent to a randomly chosen destination that stays the same for all the burst length. The rate of each burst sending packets is 3 packets/sec. The time elapsed between two application bursts is normally distributed in [0.1, 3] sec. The packet size is 512 bytes.
- *Routing Layer:* The routing model is the heart of the simulator. This model depicts the Ad-hoc On-demand Distance Vector (AODV) routing protocol, all of its functions, parameters and their implementation [24].
- *MAC Layer:* The simple implementation for this layer has been used. The outgoing messages (from routing layer) are let pass through to the physical layer. The incoming one (from physical layer) instead is delivered to the routing layer with an *MM1 queue* policy with queue size 5k bytes. When an incoming message arrives, the module checks a flag that indicate if the routing layer is busy or not. If so, the message will be saved in the queue using Drop-Tail, Adaptive RED, or Fuzzy-AQM algorithm. Note that Drop-Tail is a special case of AQM with the following condition:

$$p_d = \begin{cases} 1 & \text{if } q_c = q_m \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

  The parameters of Adaptive RED (see notation in [12]) are set at min_{th} = 1.5k bytes, max_{th} = 3k bytes, max_p = 0.01, w_q = 0.002, $\alpha = max_p/4$, and $\beta = 0.9$. When the routing layer is not busy, the MAC module picks the first message from the queue and sends it upward.
- *Physical Layer:* It deals with the on-fly creation of links that allow the exchange of messages among the nodes. Every time a node moves from its position, an interdistance check on each node is performed. If a node gets close enough (depending on the transmission power of the moving nodes) to a new neighbor, a link is created between the two nodes with the following properties: Channel bandwidth is 11 Mb/s (IEEE 802.11a) and delay is 10 μ s. Each node has a defined transmission range chosen from a uniformly distributed number between [90, 120] m.
- *Mobility Layer:* The *random waypoint* model was adopted for the mobility layer. It is one of the most used mobility pattern in the ad-hoc network simulations. This is because of its simplicity and its quite realistic mobility pattern. In this mobility model, a node randomly selects a destination. On reaching the destination, another random destination is targeted after 3 seconds *pause time*. The speed of movement of individual nodes range between [11, 16] m/sec. The direction and magnitude of movement was chosen from a uniformly distributed random number.

Three different network sizes are modeled: 700m×700m map size with 25 and 35 nodes and 800m×800m map size with 45 nodes. Each simulation run takes 300 simulated seconds. Multiple runs were conducted for each scenario and collected data was averaged over those runs.

5.2. Performance Metrics

The following metrics were used for measuring performance:

- *Drop Ratio:* The percentages of packets that are dropped from the queue due to overflow (congestion) to the total arrival in the queue.
- *Invalid Route Ratio*: Calculated as follows:

$$\text{Invalid Route Ratio} = \frac{\sum_{i=1}^{n} \text{Number of invalid routes}}{\sum_{i=1}^{n} \text{Number of valid routes}}$$

Each time a route is used to forward a data packet, it is considered as a valid route. If that route is unknown or expired, it's considered as invalid route.

- *Average End-to-End Delay*: Average packet delivery time from a source to a destination. First, for each source-destination pair, average delay for packet delivery is calculated. Then the whole average delay is calculated from average delay of each pair. End-to-end delay includes the delay in the send buffer, the delay in the interface queue, the bandwidth contention delay at the MAC layer, and the propagation delay.
- *Routing Overhead:* Calculated as follows:

$$\text{Overhead} = \frac{\sum_{i=1}^{n} \text{Number of SentCtrlPkt by source}}{\sum_{i=1}^{n} \text{Number of received data by destination}}$$

Where n is number of nodes in the network and *SentCtrlPkt* is control packets used by AODV and described in Table 3. This metric can be employed to estimate how many transmitted control packets are used for one successful data packet delivery. We use it to study the effect of AQM algorithms on the efficiency and scalability of the routing protocol.

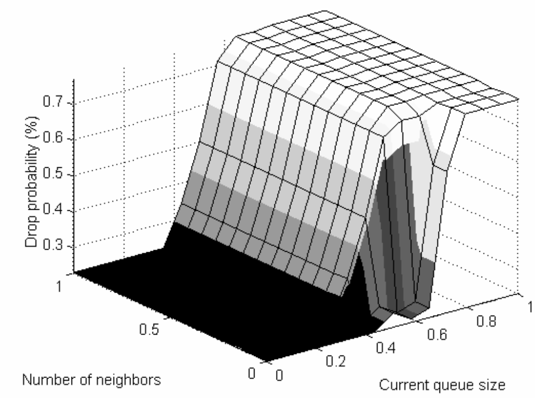


Figure 5. The input-output relationship of the fuzzy-AQM.

Table 3. Control packets used by AODV.

Message	Description
RREQ	A Route Request message
RREP	A Route Reply message
RERR	A Route Error containing a list of the invalid destinations
RREP_ACK	A *RREP* acknowledgment message

6. Simulation Results and Evaluations

6.1. Drop Ratio Details

The average control messages drop ratio for the proposed Fuzzy-AQM algorithm is less than other conventional algorithms as shown in Figure 6-a. The percentage of Fuzzy-AQM improvement compared to Drop-Tail and Adaptive RED algorithms is: 93.9% and 74.5% for 25 nodes, 65.8% and 33.5% for 35 nodes, and 75.1% and 49.7% for 45 nodes, respectively.

This improvement of the fuzzy algorithm is a result of choosing the neighbors' density parameter to estimate the size of incoming traffic and hence start the early dropping policy as needed. Despite the data packets drop ratio of Fuzzy-AQM is little bit higher than adaptive RED, as shown in Figure 6-b, this is enough to produce a higher enhancement in the control messages drop ratio. This enhancement is a result of the wide difference between the size of data packets (512 bytes) and control messages (64 bytes). Consequently, at congestion time, dropping one data packet allows the queue to accept eight control messages.

Drop-Tail algorithm doesn't have any mechanism to distinguish between data and control packets like other AQM algorithms. Moreover, the number of control messages in ad-hoc network is much higher than data packet; to provide continuous update of topology changes. Those two reasons affect a high control messages drop ratio for the Drop-Tail algorithm as shown in Figure 6-a.

6.2. Invalid Route Ratio Details

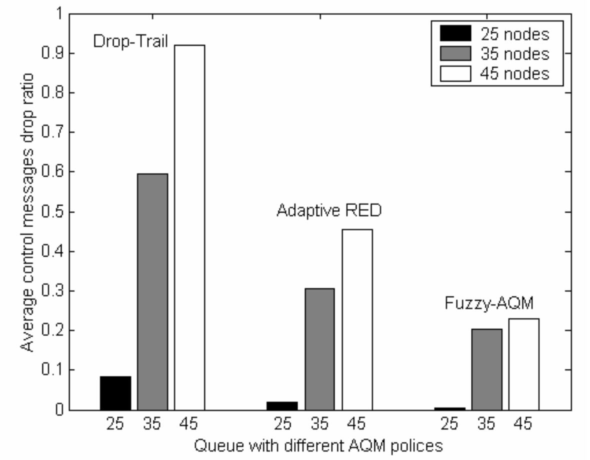
The Fuzzy-AQM algorithm has less average invalid route ratio compared to other conventional AQM as shown in Figure 7. This decrement of the proposed algorithm is about: 20.3% and 23.1% for 25 nodes, 31.1% and 14.6% for 35 nodes, and 22.4% and 12.9% for 45 nodes, respectively.

Information about route breaks is broadcasted as an RERR message. The Fuzzy-AQM algorithm allows more control messages to pass the queue to the upper routing layer as shown in Figure 6. This increased number of received control messages helps the nodes with Fuzzy-AQM to be more accurate to topology changes and have precise updated routing tables, hence, have less invalid routes.

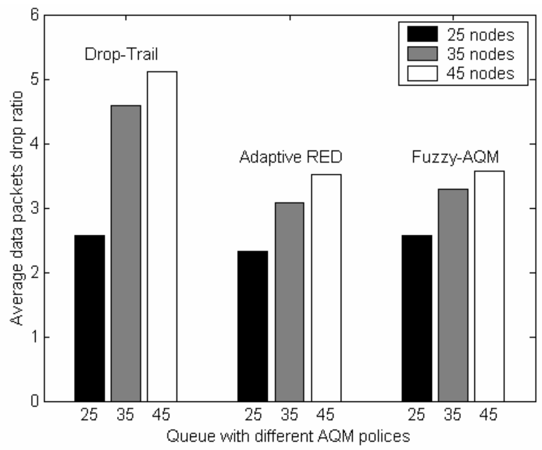
6.3. Average End-to-End Delay Details

Figure 8 indicates that the proposed Fuzzy-AQM algorithm has lower average end-to-end delay compared to other conventional algorithms. This decrement is approximately: 17.2% and 6.3% for 25 nodes, 24.1% and 11.6% for 35 nodes, and 33.6% and 21.6% for 45 nodes, respectively.

The nodes that have conventional AQM algorithms have higher invalid route ratio as shown in Figure 7, therefore they suffer longer routing delay to recover from broken paths and discover new ones. To recover a broken path, an RERR message must first be launched from the intermediate nodes to tell the source node about the broken link. The source node deletes the corresponding entry from its routing table. The RREQ must then be broadcasted from the source to the destination, and an RREP consequently has to be transmitted back to the source. Data packets are buffered at the source node during this process and the duration of their buffering adds more time delay to the end-to-end delay. The nodes with Fuzzy-AQM algorithm, on the other hand, have reliable routing tables that minimize the need to this recovery process.



(a) Control messages drop ratio comparison.



(b) Data packets drop ratio comparison.

Figure 6. Drop ratio comparison.

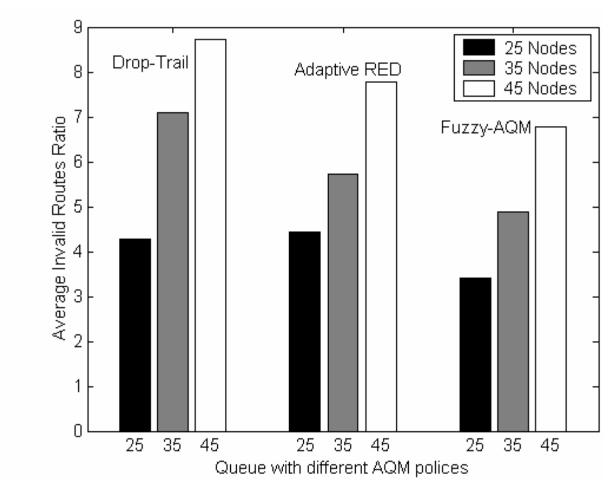


Figure 7. Invalid route ratio comparison.

6.4. Routing Overhead Details

As expected, the AQM algorithms don't have major effect on the routing protocol efficiency or scalability as shown in Figure 9. These algorithms maximize the number of 'received' control messages, meanwhile they have no effect on 'sent' control messages (see equation 10). This is because the control messages used in AODV are broadcast messages; that is, they will not be resent if they are dropped or lost.

The Drop-tail algorithms has worst routing overhead ratio as the number of node increase as a result of increasing data packets drop ratio which is clear in Figure 6-b. Meanwhile, the data packets dropping ratio is nearly the same for adaptive AQM algorithms (ARED and Fuzzy-AQM) that results in no major difference in routing overhead ratio.

6.5. Drop Probability Values

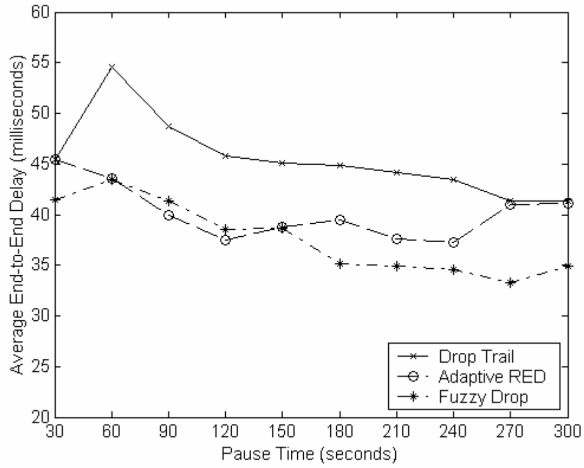
In Drop-Tail algorithm, p_d always take a static value of 1 to start packet dropping at overflow. In Adaptive RED algorithm, p_d increases linearly between the two thresholds min_{th} and max_{th} in dependent on the average queue size '*avg*'. Some studies [26] showed that using linear p_d function can result in forced drops when q_c exceeds max_{th} or link under-utilization when q_c decreases to zero. This is an evident that the original linear drop function does not perform well within a wide range of loads.

The p_d values used by the proposed Fuzzy-AQM for randomly chosen node in the 25 nodes simulated network are shown in Figure 10. It is evident that the drop function is non-linear and a high load requires a disproportionately higher p_d than a low load to keep the queue size in the same range. Non-linearity of p_d function is also clear in the input-output relation as shown in Figure 5.

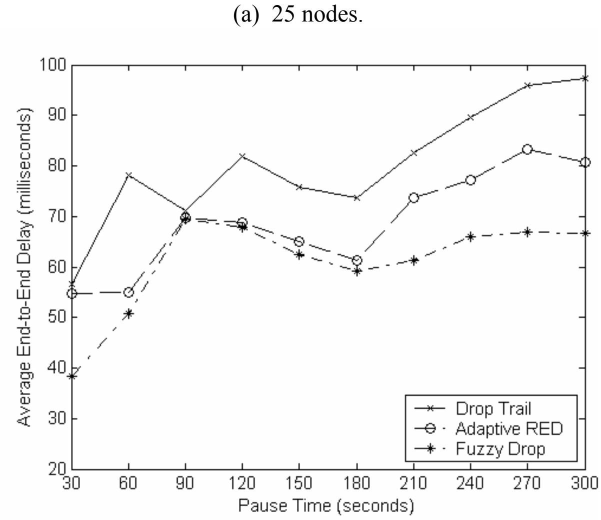
The comparison between the average p_d values used by every node in the 25 nodes and the 35 nodes networks is shown in Figure 11. Due to higher neighbors' density, 35 nodes network have higher p_d values than 25 nodes network. This is a result of increasing neighbors' density which will also increase the number of control messages.

7. Conclusions

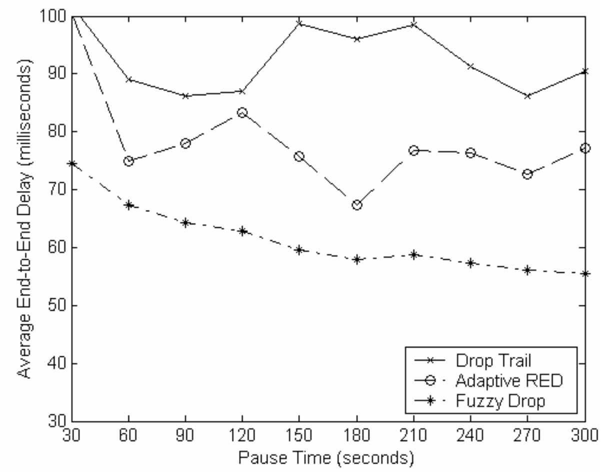
In this study, a novel AQM algorithm (Fuzzy-AQM) based on fuzzy logic system was suggested. This algorithm for early packets dropping is implemented in wireless ad-hoc networks in order to provide effective congestion control by achieving high queue utilization, low packet losses and delays. The proposed scheme is contrasted with a number of well-known AQM schemes through a wide range of scenarios. From the simulation results, the efficiency of the proposed fuzzy AQM policy in terms of routing overhead, average end-to-end delay and average packet losses are pronounced than other AQM polices, with capabilities of adapting to high variability and uncertainty in the mobile ad-hoc networks.



(a) 25 nodes.



(b) 35 nodes.



(c) 45 nodes.

Figure 8. Average end-to-end delay comparison.

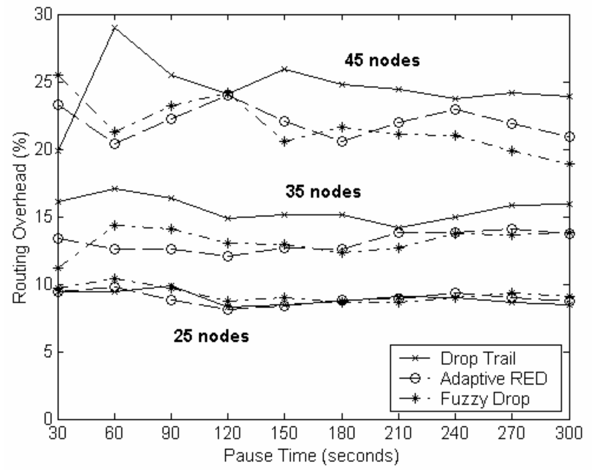


Figure 9. Routing overhead comparison.

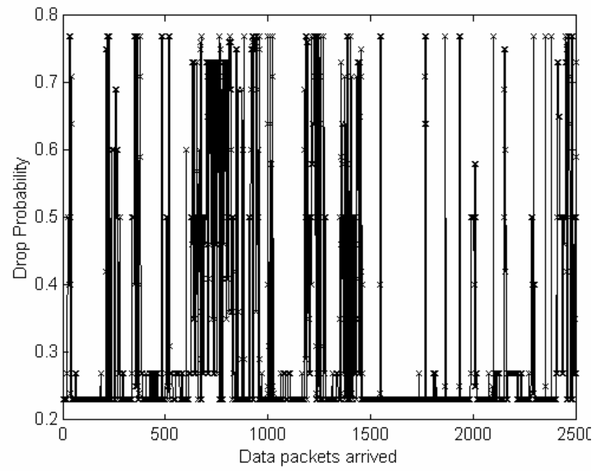


Figure 10. Drop probability values used by a node.

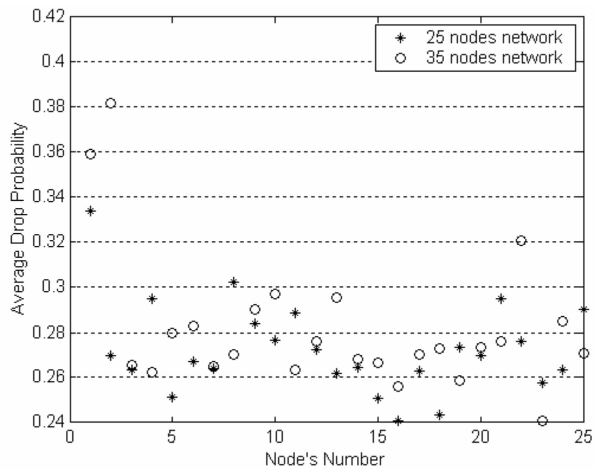


Figure 11. Average p_d values used by 25 and 35 nodes networks.

References

[1] Al-Frihat J., "Advanced Queue Management Algorithms for Computer Networks," *Studies in Informatics and Control Journal*, vol. 14, no. 2, pp. 111-116, June 2005.

[2] Aoul Y. H., Nafaa A., Negru D., and Mehaoua A., "FAFC: Fast Adaptive Fuzzy AQM Controller for TCP/IP Networks," *in Proceedings*

of IEEE Global Telecommunications Conference, vol. 3, pp. 1319-1323, November 2004.

[3] Aweya J., Ouellette M., and Montuno D. Y., "A Control Theoretic Approach to Active Queue Management," *Computer Networks*, vol. 36, no. 2-3, pp. 203-35, July 2001.

[4] Braden B., Clark D., Crowcroft J., Davie B., Deering S., Estrin D., Floyd S., Jacobson V., Minshall G., Partridge C., Peterson L., Ramakrishnan K., Shenker S., Wroclawski J., and Zhang L., "Recommendations on Queue Management and Congestion Avoidance in the Internet," *Request for Comments (RFC) 2309*, April 1998.

[5] Brandauer C., Iannaccone G., Diot C., Ziegler T., Fdida S., and May M., "Comparison of Tail Drop and Active Queue Management Performance for Bulk-Data and Web-like Internet Traffic," *in Proceedings of the 6th IEEE Symposium on Computers and Communications*, Hammamet, July 2001.

[6] Christiansen M., Jeffay K., Ott D., and Smith F. D., "Tuning RED for Web Traffic," *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 249-264, June 2001.

[7] Chrysostomou C., Pitsillides A., Hadjipollas G., Sekercioglu Y. A., and Polycarpou M., "Fuzzy Logic Congestion Control in TCP/IP Best Effort Networks," *in Proceedings of the Australian Telecommunications, Networks and Applications Conference (ATNAC'03)*, Melbourne, Australia, December 2003.

[8] Chrysostomou C., Pitsillides A., Rossides L., Sekercioglu Y. A., and Polycarpou M., "Congestion Control in Differentiated Services Networks Using Fuzzy-RED," *IFAC Journal Control Engineering Practice*, vol. 11, no. 10, pp. 1153-1170, October 2003.

[9] Cisco Systems, *Weighted Random Early Detection on the Cisco 12000 Series Router*, March 2002.

[10] Di Fatta G., Hoffmann F., Lo Re G., and Urso A., "A Genetic Algorithm for the Design of a Fuzzy Controller for Active Queue Management," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 33, no. 3, pp. 313-324, August 2003.

[11] Floyd S. and Jacobson V., "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions Networking*, vol. 1, no. 4, pp. 397-413, August 1993.

[12] Floyd S., Gummadi R., and Shenker S., "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management," available at *http://www.icir.org/floyd/red.html*, 2005.

[13] Iannaccone G., May M., and Diot C., "Aggregate Traffic Performance with Active Queue Management and Drop from Tail," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 3, July 2001.

[14] Jacobson V., "Congestion Avoidance and Control," *in Proceedings of ACM SIGCOMM'88*, vol. 18, no. 4, pp. 314-329, August 1988.

[15] Li Z., Zhang Z., Addie R., and Clerot F., "Improving the Adaptability of AQM Algorithms to Traffic Load Using Fuzzy Logic," *in Proceedings of the Australian Telecommunications, Networks and Applications Conference (ATNAC'03)*, Melbourne, Australia, December 2003.

[16] Lin D. and Morris R., "Dynamics of Random Early Detection," *in Proceedings of ACM SIGCOMM'79*, France, pp. 127-137, 1997.

[17] Lin W., Wong A., and Dillon T., "A Novel Fuzzy Logic Controller (FLC) for Shortening the TCP Channel Roundtrip Time by Eliminating User Buffer Overflow Adaptively," *in Proceedings of the 28th Australasian Computer Science Conference (ACSC2005)*, vol. 38, pp. 29-38, Newcastle, Australia, January 2005.

[18] May M., Bolot J., Diot C., and Lyles B., "Reasons not to Deploy RED," *in Proceedings of the 7th International Workshop on Quality of Service (IWQoS'99)*, pp. 260-262, June 1999.

[19] Misra V., Gong W. B., and Towsley D., "Fluid-Based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED," *ACM SIGCOMM Computer Communication Review*, pp. 151-160, 2000.

[20] Murthy C. S. and Manoj B. S., *Ad-Hoc Wireless Networks: Architectures and Protocols*, Prentice Hall, 2004.

[21] *OMNeT++* web site, available at: http://www.omnetpp.org, 2005.

[22] Ott T. J., Lakshman T. V., and Wong L., "SRED: Stabilized RED," *in Proceedings of IEEE INFOCOM*, vol. 3, pp. 1346-1355, 1999.

[23] Pedrycz W., "Why Triangular Membership Functions?," *Fuzzy Sets System*, vol. 64, pp. 21-30, 1994.

[24] Perkins C., Royer E. M., and Das S. R., "*Ad-Hoc On-Demand Distance Vector (AODV) Routing*," available at: http:// www.draft-ietf-manet-aodv-13.txt, Febraury 2003.

[25] Pitsillides A., Sekercioglu Y. A., and Ramamurthy G., "Effective Control of Traffic Flow in ATM Networks Using Fuzzy Logic Based Explicit Rate Marking (FERM)," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 2, pp. 209-225, February 1997.

[26] Plasser E., Ziegler T., and Reichl P., "On the Non-Linearity of the RED Drop Function," *in Proceedings of the 15th International Conference on Computer Communication*, India, vol. 1, pp. 515-534, August 2002.

[27] Ren F., Ren Y., and Shan X., "Design of a Fuzzy Controller for Active Queue Management," *Computer Communications*, vol. 25, no. 9, pp. 874-883, June 2002.

[28] Rossides L., Chrysostomou C., Pitsillides A., and Sekercioglu Y. A., "Fuzzy Logic Controlled RED: Congestion Control in TCP/IP Differentiated Services Networks," *Soft Computing Journal*, vol. 8, no. 2, pp. 79-92, December 2003.

[29] Tseng Y., Li Y., and Chang Y., "On Route Lifetime in Multihop Mobile Ad-Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, pp.366-376, October 2003.

[30] Wang C., Li B., Sohraby K., and Peng Y., "AFRED: An Adaptive Fuzzy-Based Control Algorithm for Active Queue Management," *in Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks (LCN'03)*, pp. 12-20, October 2003.

[31] Yager R. R. and Filev D. P., *Essentials of Fuzzy Modeling and Control*, John Wiley & Sons, pp. 109-153, 1994.

[32] Yanfei F., Fengyuan R., and Chuang L., "Design an Active Queue Management Algorithm based on Fuzzy Logic Decision," *in Proceedings of IEEE International Conference on Communication Technology (ICCT'03)*, vol. 1, pp. 286-289, April 2003.

Essam Natsheh obtained his MSc in computer engineering from the Arab Academy for Science and Technology in Egypt, in 1999. He subsequently worked as a lecturer at Al-Alamiah Institute for Computer and Technology, SA, in 2002. He worked as a lecturer also at the Information Systems Department of the King Faisal University, SA, from 2002 to 2003. Since 2003, Natsheh has been a member of a research group headed by Dr. Jantan A. at the University Putra Malaysia, which investigates issues related to the design and analysis of ad-hoc wireless networks.

Adznan B. Jantan obtained his MSc in digital systems from Cranfield Institute of Technology, UK, in 1982 and his PhD in speech recognition systems from the University College of Swansea, UK, in 1988. Since 2002, he is an associate professor at the Department of Computer and Communication System at the University Putra Malaysia, where he has been conducting research in computer networking, pattern recognition, and digital systems design.

Sabira Khatun received her BSc (Hons.), MSc in applied mathematics and PhD on hydromagnetic stability from the University of Rajshahi, Bangladesh in 1988, 1990, and 1994, respectively. She received her second PhD in communications and networking from University Putra Malaysia in 2003. She became a lecturer at the Department of Computer Science and Engineering, University Khulna, Bangladesh in 1991, and promoted to assistant professor in 1994. She joined the Department of Computer & Communication Systems Engineering, University Putra Malaysia in 1998. She is an active researcher of Teman project and MyREN Research Community. She is a member of IEEE. Her research interest spans broadband and wireless communications, and network management, including software defined radio and IPv6.

Shamala Subramaniam completed her PhD from University Putra Malaysia in 2002. Currently, she is a lecturer at the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, University Putra Malaysia. Her research interest includes scheduling algorithms, congestion control, real-time systems, modeling, and simulation.