BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **21**, No 2
Sofia • 2021 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2021-0017

Fuzzy-Logic Based Active Queue Management Using Performance Metrics Mapping into Multi-Congestion Indicators

Mosleh M. Abualhaj[1], *Mayy M. Al-Tahrawi*[2], *Abdelrahman H. Hussein*[2], *Sumaya N. Al-Khatib*[1]

[1]*Department of Networks and Information Security, Al-Ahliyya Amman University, Amman, Jordan*
[2]*Department of Computer Science, Al-Ahliyya Amman University, Amman, Jordan*

E-mails: m.abualhaj@ammanu.edu.jo mtahrawi@ammanu.edu.jo a.husein@ammanu.edu.jo sumayakh@ammanu.edu.jo

Abstract**:** *The congestion problem at the router buffer leads to serious consequences on network performance. Active Queue Management (AQM) has been developed to react to any possible congestion at the router buffer at an early stage. The limitation of the existing fuzzy-based AQM is the utilization of indicators that do not address all the performance criteria and quality of services required. In this paper, a new method for active queue management is proposed based on using the fuzzy logic and multiple performance indicators that are extracted from the network performance metrics. These indicators are queue length, delta queue and expected loss. The simulation of the proposed method show that in high traffic load, the proposed method preserves packet loss, drop packet only when it is necessary and produce a satisfactory delay that outperformed the state-of-the-art AQM methods.*

Keywords**:** *Congestion control, network performance, active queue management, fuzzy logic.*

1. Introduction

Congestion at the router occurs when the buffer to which the transmitted packets are to be accommodated, is overflowed. Congestion is mainly occurs because of the bursty traffic that increases and decreases suddenly [1-4]. As the traffic load increases with low departure rate, more packets are accommodated in the buffer over time, which eventually leads to a buffer overflow. At this point, the arrival packets cannot be accommodated and lead to increase packet loss, worsen the delay and decrease throughput significantly. As the network is connected with the routers, the consequences at the router are propagated to the entire network. Drop-Tail (DT) technique can be implemented to avoid congestion and its consequences. Using DT, the packets are dropped as the queued packets reach a threshold. Yet, the problem of the DT technique is the inability to deal with sudden congestion, which eventually leads to packet loss. Active Queue Management (AQM) is used to monitor the router buffer actively, avoid congestion and ease the serious consequences of the bursty traffic [5]. Compared to DT, AQM methods applies different packet dropping policies to eliminate any possible congestion, as illustrated in Fig. 1.

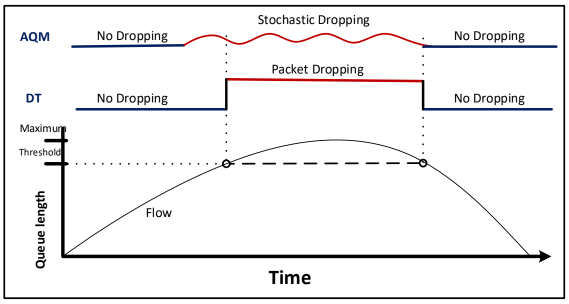


Fig. 1. DT vs AQM packet dropping policies

Based on the performance indicator, AQM decides whether to accommodate or drop the arrival packet to avoid buffer overflowing. AQM does not depend on a fixed threshold, as compared to the DT; instead, AQM stochastically drops packets based on the network load. Performance indicators are used for the monitoring purpose and to calculate a Dropping probability (Dp). The goal of the Dp is to set a stochastic mechanism for packet dropping to avoid global synchronization. Accordingly, the value of the Dp depends on the characteristic of the performance indicator [6].

Random Early Detection (RED) has been the first method in the AQM technique and has been developed based on queue monitoring through the Average Queue Length (AQL) [5]. RED has ben then extended using various performance indicators and through variation of the dropping policies [7, 8]. The goal of the proposed methods was to improve the calculation of the Dp to improve network performance. Accordingly, various indicators have been utilized, such as AQL, the instance queue length (Q), Arrival-Rate, Load-Rate, Estimated Delay, etc., Fuzzy-based AQM uses input variables that are equivalent to the performance indicators used in the non-fuzzy methods. The performance indicator AQL is mapping Q and ΔQ input variables [9]. The existing methods that belong to both techniques (conventional and fuzzy-based) have been focused on reducing the packet loss and avoid unnecessary packet dropping [10, 11].

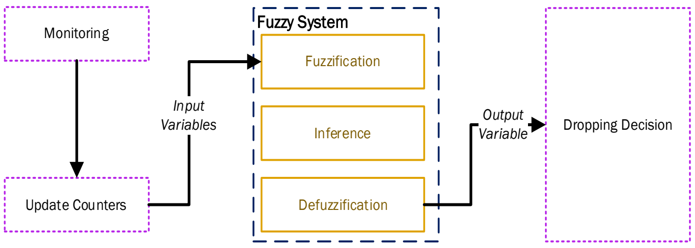


Fig. 2. Fuzzy-based AQM technique

Generally, the consequences at the router are propagated to the network as a whole. Accordingly, the quality of services and the desired performance embedded in increasing the throughput and decreasing the delay and loss requires multiple performance indicators that take into consideration the delay, dropping and loss at the router buffer. Existing AQM methods utilize one or two performance indicators at most, which cannot capture the desired performance. Besides, a single performance indicator in the conventional technique requires multiple counters and thus, multiple input variables in the fuzzy-based AQM. Thus, there is a limitation in the utilized indicators, input variables, which influence the network performance [5, 12-14].

In this paper, a fuzzy-based AQM method is developed based on three input variables that are extracted from the performance metrics. Accordingly, first, the desire performance is analyzed; the counters are identified then. These counters are used as inputs to a fuzzy-logic system. The fuzzification process, membership functions and fuzzy sets are defined to convert the input variables into linguistic terms with membership degree. The rules are built based on a three-dimensional matrix with all possible inputs and the linguistic term of the desired output. The rules are then used to map the input into the output variable. Following the rule implementation, the aggregation and defuzzification processes are implemented to obtain the crisp value of the output variable, Dp. Accordingly, the results of this paper are organized as follows: Section 2 presents a literature review on the related work for both conventional and fuzzy-based AQM. Section 3 presents the proposed method, its components, calculation and steps, while Section 4 represents the results of the simulation in comparison with the existing and up-to-date AQM methods. Finally, Section 5 presents the conclusion and future work.

2. Related works

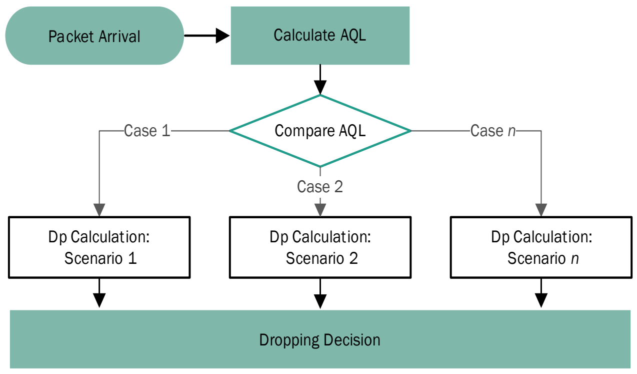
The RED Algorithm has been proposed based on maintaining the value of the AQL and using the calculated AQL value for calculating Dp and making the dropping decision. The algorithm has been built based on case-based reasoning with various parameters. Besides predicting congestion, avoid loss and unnecessary packet dropping, RED has been built to avoid global synchronization and avoid, as much as possible, dropping of consequent packets. The complex process implemented by the RED and the fulfilled objectives followed by an adaptation of RED by the Internet Engineering Task Force (IETF) in RFC 2309 [5]. There are various parameters and thresholds that have been identified and utilized to address these multiple objectives, as given in Algorithm 1. The parameters are initialized at line 1. Then, the AQL is calculated at lines 3-4. The stochastic dropping is implemented at lines 5-9. Full dropping is implemented at line 10 and no-dropping takes place at line 11. Ideal time updating is implemented at lines 12-13.

Algorithm 1. RED Algorithm

```
Step 1. INITIALIZATIONS: AQL:=0, Count = –1
Step 2. FOR-EACH Arrival-Packet($a$) DO
Step 3.     IF (Q==0) THEN AQL $\coloneqq \left(1-w_q\right)^{f(\mathrm{cTime}-\mathrm{iTime})} * \mathrm{AQL}$
Step 4.     ELSE  AQL $\coloneqq \left(1-w_q\right) * \mathrm{AQL} + w_q * \mathrm{AQ}$
Step 5.     IF ($\mathrm{min}_{\mathrm{th}} \leq \mathrm{AQL} < \mathrm{max}_{\mathrm{th}}$)  THEN
Step 6.         Count ++
Step 7.         $\mathrm{Dp}' = D_{\max} * (\mathrm{AQL} - \mathrm{min}_{\mathrm{th}}) / (\mathrm{max}_{\mathrm{th}} - \mathrm{min}_{\mathrm{th}})$
Step 8.      $\mathrm{Dp} = \mathrm{Dp}'/(1 - \mathrm{Count} * \mathrm{Dp}')$
Step 9.      IF (Drop(Dp) == TRUE) THEN Drop Packet (a), Count:= 0
Step 10.   ELSE IF ($\mathrm{AQL} > \mathrm{max}_{\mathrm{th}}$) THEN Drop Packet (a), Count:= 0
Step 11.   ELSE Count:= -1
Step 12.  IF ($Q$==0 && Idle :=FALSE) THEN iTime = cTime, Idle:=TRUE
Step 13.   ELSE Idle := FALSE
```

Count: A counter to reflect the recent accommodation value
AQL: Average Queue Length
Q: Instance Queue Length
cTime: Current Time
iTime: Idle Time
w: Weight value
D_{max}: maximum probability value
$\mathrm{Min}_{\mathrm{th}}$: minimum threshold
$\mathrm{Max}_{\mathrm{th}}$: maximum threshold
Idle: A variable to denote the idleness of the traffic

With the growth of the distributed and remote applications and shared resources, the utilization of the network resources has been growing, which creates new demands for managing the queue at the router buffer. Thus, a need to reduce packet dropping without consequences on packet loss has emerged. Besides, quick response to sudden congestion and an increase in the traffic load is required. Accordingly, various algorithms have been proposed to improve the performance of the RED method. Among these, Gentle RED (GRED) [15], Adaptive RED (ARED) [16], Dynamic RED (DRED) [17] and Double Slop RED (DSRED) [18] focused on using AQL while changing the decision making process and the Dp calculation. The goal of these methods is to maintain the ability of the AQL in avoiding false congestion while enhancing the reaction to true congestion by adapting a more fixable approach compared to the semi-static approach utilized in the RED method. A general form of using AQL in AQM is utilized in Fig. 3.



Packet Arrival
Calculate AQL
Case 1
Compare AQL
Case n
Case 2
Dp Calculation: Scenario 1
Dp Calculation: Scenario 2
Dp Calculation: Scenario n
Dropping Decision

Fig. 3. AQL-based AQM Algorithm

Adaptive Virtual Queue (AVQ) [19] used both Arrival rate and Load rate together with a virtual queue that is linked with the actual queue at the router buffer. The actual queue is synchronized with a virtual queue with capacity that is less than the actual one. Accordingly, congestion is detected when the virtual queue is overflowed. Moreover, the virtual link is used and linked with the monitoring of the actual link. When the traffic load increase at the virtual queue, the Dp is increased and vice versa. Stabilized AVQ (SAVQ) [20] and Enhanced AVQ (EAVQ) [21] were built as an extension to AVQ. Random Exponential Marking (REM) [22] used queue length (*q*) and the transmission rate as performance indicators. Proportional Integral controller (PI) [23] used *q* and the packet loss as performance indicators. BLUE method [24] used packet loss as a performance indicator and adaptive calculation of the Dp that differ completely from the way by which RED is implemented. Accordingly, Dp is not calculated but adaptively increased/decreased based on the indicator and the link idleness. Similarly, Multi-level RED (MRED) [25] used the loss as a performance indicator. Effective RED (ERED) [26] extended RED and used AQL and *q* as congestion indicators. While Self-tuning RED, [27] used AQL with adaptive technique as similar to BLUE. Stabilized RED (SRED) [28] used *q* as a performance indicator. B a k l i z i et al. [29] and B r i s c o e [3] used AQL as similar to RED.

Table 1. Summary of the fuzzy-based AQM methods

Reference	Inputs		Output		Rule Set*
	Set (Length)	Function	Set (Length)	Function	
FRED	Queue (3) & Δ Queue (3)	Trapezoidal	Dp (4)	Trapezoidal	Customized
FEM	Queue (3) & Δ Queue (3)	Trapezoidal	Dp (4)	Triangular	Customized
FB	Queue (3) & Δ Queue (3)	Trapezoidal	Dp (4)	Trapezoidal	Full
Fuzzy AQM	Arrival rate (3) & Arrival factor (3)	Trapezoidal	ΔDp (5)	Trapezoidal	Full
FCRED	Actual-Target *Q* (7) & Δ Actual-Target *Q*(5)	Triangular	ΔDp (9)	Triangular	Full
Fuzzy-logic Controller-based RED	AQL (3) & Loss (3)	Trapezoidal	Dp (5)	Trapezoidal	Full
GREDFL	AQL (3) & Delay (3)	Trapezoidal	Dp (4)	Trapezoidal	Full
Fuzzy RED	Queue (3) & Δ Queue (3) & Delay (3)	Triangular	Dp (4)	Triangular	Full
Fuzzy ERED	AQL (4) & Queue (3)	Triangular	Dp (4)	Triangular	Full
FLRED	AQL (4) & Delay (4)	Trapezoidal	Dp (4)	Trapezoidal	Full

* ***Full***: *Rules covering all possible combination for the terms in the input sets, such as each rule cover exactly a single term from each input.* ***Customized***: *Rules does not cover all cases or using a set with individual rules; each may cover multiple terms.*

As more complex performance indicators are utilized, such as the transmission rate and estimated loss, more parameters are utilized. Parameter initialization problem grew and there was a need to ease the problem and face the challenges of improving the network performance. Fuzzy-RED (FRED) [9] used AQL with fuzzy inference problem to ease the problem of parameter settings. Similarly, Fuzzy BLUE (FB) [30] used queue size and packet loss as performance indicators. These methods produced a single output variable, which is Dp. Fuzzy Controller Random Early Detection (FCRED) (Sun et al., 2007) used differences between the actual and target queue length and the change in such difference as input variables to produce a control value that is used to calculate the value of Dp. Generally, the existing fuzzy-based AQM methods differ in five aspects, these are: 1) input variables; 2) output variable; 3) membership function; 4) fuzzy sets; 5) fuzzy rule set. Table 1 summarizes the existing fuzzy-based methods based on these aspects.

Overall, fuzzy-based AQM used common input variables, such as queue, AQL, delay and loss, to produce the Dp or a variable that used to calculate Dp using an equation. These inputs are the indicators that were mapped from the conventional-based AQM and calculated before they can be used with the fuzzy systems. There are some common characteristics among these variables, such as using a queue to calculate AQL, delay and arrival rate. The utilized combination of these variables does not cover all performance metrics. Thus, the input variables are not independent and not comprehensive. A summary of the utilized indicators and their intrinsic variables are summarized in Table 2.

Table 2. Summary of the utilized indicators

Indicator	Variable, s
Q	Instance queue length
AQL	Q and Previous AQL
ΔQ	Q and Previous Q
Arrival-Rate	Number of arrived packets
Departure-Rate	Number of departed packets
Load	Arrival-Rate & Departure-Rate
Delay	Q and ΔQ

Accordingly, to optimize the performance of the AQM, the input variables should be analyzed, and their combination should cover all desired performance metrics. Besides, except for Fuzzy RED, the existing fuzzy-based used two input variables, which are not enough to cover the desired performance metrics. In this paper, the performance metrics will be analyzed, and the core components of these variables will be used as input to a fuzzy system. Using trapezoidal allows for more flexibility in the process and can be easily mapped into triangular, yet triangular cannot be mapped into trapezoidal. Finally, recent studies focus on covering all possible combination of the input set, each with individual rules to avoid any complexity. Thus, in the proposed method, trapezoidal membership function and full rule-set will be used.

3. Proposed work

The proposed AQM method is built to optimize the network performance by covering the on-demand performance metrics. The framework for building the proposed method, as illustrated in Fig. 4, consists of the following steps: 1) identify the input variables by analyzing loss, delay and throughput criteria; 2) identify the fuzzification components, which are the linguistic sets and the membership functions; 3) set up the rule set based on trial-and-error while adjusting the membership function of the output variables; 4) define the aggregation process; 5) set up the simulation environment.

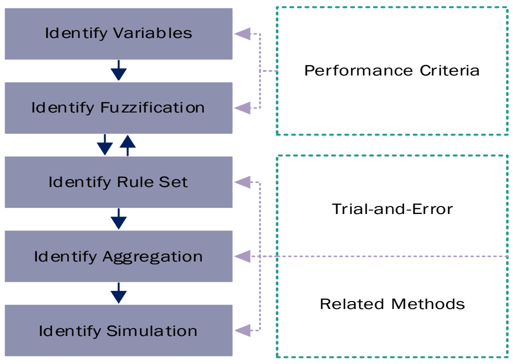


Fig. 4. The framework of the proposed method

3.1. Input variables

The input variables are extracted based on analyzing three performance metrics; these are delay, throughput and loss. The delay can be estimated based on the transmission rate and the queue length. Delay is proportional to the queue length as the arrival packet has to wait in the FIFO (First-In-First-Out) queue model that is commonly implemented in the router buffer. Besides, the delay is proportional to the transmission rate as the increase of the departure λ_{out} decrease delay and the increase of the arrival rate λ_{in} increase delay subject to the queue length. The difference between the λ_{out} and λ_{in} can be estimated by the changes in the queue length over time. Accordingly, the estimated delay can be denoted as

(1) $$d_{\text{queue}} \propto q * \Delta q,$$

where d_{queue} is the delay at the queue. The Packet Loss (PL) is inversely proportional to the remaining capacity (v) of the router buffer. As the remaining capacity decreases, the probability of packet loss increases and vice versa. Accordingly, the estimated loss can be denoted as

(2) $$\text{PL} \propto 1/v.$$

The throughput can be estimated based on the dropping and the loss. Dropping is inversely proportional to the queue length as AQM is responsible for early dropping of packets in a heavily loaded network. Loss as mentioned depends on the remaining capacity (v). Accordingly, the estimated Throughput (T) can be denoted as

(3) $$T \propto v/q.$$

Accordingly, three variables are extracted from the performance indicators, these are the queue length (q*), the changing in the queue (Δq*) and the remaining capacity (v*). The values of these variables are calculated and normalized in the range [0, 1]. The variable q, which is a counter of the number of packets, is divided by the buffer capacity (c). The value of the Δq is calculated as the difference between two consequent captured lengths, the current and the previous and then normalized in the range [0, 1]. As such if the difference is positive, the calculated value will be in the range (0.5, 1] and the range [0, 0.5) will be for negative differences. The value 0.5 means that the difference is zero. The variable v, which is a counter remaining positions, is divided by the buffer capacity (c). These variables are used as input variables for the proposed method based on the description of that is summarized in Table 3.

Table 3. Summary of the Input Variables

Variable	Calculation	Description
Normalized queue length (q*)	$q^*{}_t = q_t/c$	The length of the queue based on the accommodated packets
Queue change (Δq*)	$\Delta q^*{}_t = (q_t - q_{t-1} + c)/(2 * c)$	The differences in queue length between the current time and the previous time
Normalized remaining capacity (v*)	$v^*{}_t = (c - q_t)/c$	The maximum length of the queue based on the packets that can be accommodated simultaneously

3.2. Fuzzification

The value of the input variables is converted into linguistic term with membership degree using the associated fuzzy set and the membership function. According to the space partitioning approach for fuzzy system construction [31], the value range of the input variable is divided into 2N+1 equal regions. The value of N is set to 1 for the input variables. Thus, the linguistic set of the input variables is set as {low, moderate, high}. The membership function is unified for the input variables with trapezoidal, as illustrated in Fig. 5.

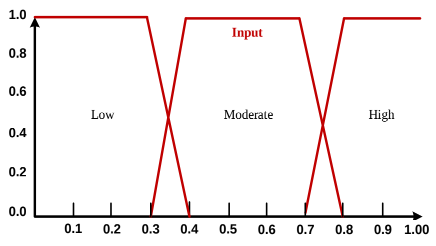


Fig. 5. Membership functions for the input variables in the proposed method

Each region in the function is characterized by four points {a, b, c, d}, where a and b form the first line and the second line is constructed by c and d. The value of these points for the input function are as follows: low 0, 0, 0.3, 0.4; moderate 0.3, 0.4, 0.6, 0.7; high 0.6, 0.7, 1.0, 1.0. Accordingly, in the fuzzification process, the crisp value of the input variable is converted into a term or multiple terms based on the value range, at which the value occurs. A membership degree (μ) with each term is calculated as

$$\mu(x) = \begin{cases} 0 & \text{if } x < a,\ x > d, \\ (x-a)/(b-a) & \text{if } a \le x \le b, \\ 1 & \text{if } b \le x \le c, \\ (d-x)/(d-c) & \text{if } c \le x \le d. \end{cases} \tag{4}$$

Accordingly, for the value of each input variable, one or more linguistic term is extracted. Based on the membership function utilized and the boundary of the regions, the extracted terms are associated with membership degree.

3.3. Output variable

The output variable Dp is associated with the membership function with equal regions as similar to the input variables. The value of *N* is set to 2 for the output variable and thus, the linguistic set for the output is {zero, low, moderate, high, extreme}. Based on previous work [11] and trial-and-error, the membership boundaries of the Dp is modified as illustrated in Fig. 6. The value of the boundary points for the output functions are modified to suits the desired output, as follows: zero 0, 0, 0.005, 0.01; low 0.005, 0.01, 0.4, 0.5; moderate 0.4, 0.5, 0.6, 0.7; high 0.6, 0.7, 0.8, 0.9; extreme 0.8, 0.9, 1.0, 1.0. In the membership function of the output, a variable is used in the defuzzification process, in which the linguistic term is converted to a crisp value, as will be discussed in the following subsections.

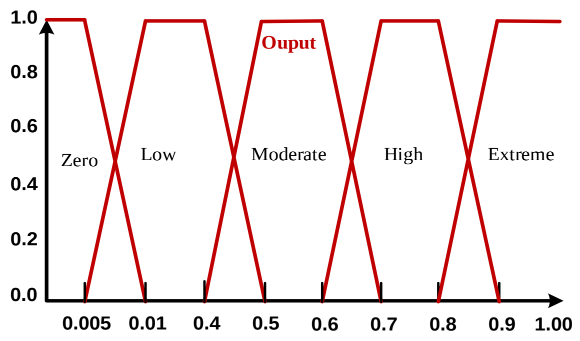


Fig. 6. Membership functions for the output variable in the proposed method

3.4. Rule-set

The trial-and-error identifies the rule-set with the expertise in the field together with those rules that can be concluded from [11]. The fuzzy rules that are built as a full-set, which means that each possible combination of the terms in the input variables is associated with a single rule. Accordingly, for the three input variables with three possible terms, 27 rules are generated in the cross matrix as given in Table 4.

As given in Table 4, an example of a rule from the rule-set is given as follows: *if Q is low and ΔQ is low, and v is high, then Dp is moderate*. The rest of the rules are extracted from the cross matrix similarly.

In the rule evaluation step, based on the terms of the output, a rule is applied, and the A confidence degree is associated with the THEN-part of the rule is produced as an output. A confidence degree for each input is obtained using AND operator. Accordingly, the minimum value of the membership degrees of the input is selected as the confidence degree of the output term.

Table 4. Rule-set cross matrix

Q	Low			Moderate			High		
Δ*Q* / *v*	Low	Moderate	High	Low	Moderate	High	Low	Moderate	High
High	zero	zero	zero	low	low	high	extreme	extreme	extreme
Moderate	zero	zero	zero	low	Moderate	high	extreme	extreme	extreme
Low	zero	zero	low	Moderate	Moderate	high	extreme	extreme	extreme

3.5. Aggregation and defuzzification

In the aggregating step, similar output terms are aggregated. As such, based on the output of the rules evaluation, multiple rules may be evaluated into a similar term, each of which with a confidence degree. These terms with their confidence degrees are aggregated into a single term and a single degree. For example, given that the rule evaluation produced two outputs of the term low with two confidence values, the output of the aggregation step will be the term low with a single confidence value. Overall, the aggregation is implemented over the confidence degrees of the THEN-part of the rules with identical terms.

In the defuzzification step, the terms of the Dp with their aggregated confidence degree are converted into crisp value. The utilized function for this purpose is the Centre Of Gravity (COG), which forms an averaging over the membership function [32].

3.6. Simulation environment

The simulation is implemented in NetBeans Integrated Development Environment (IDE) and Java Development Kit (1.6). The simulated network consists of input links, router and output links. The router is associated with a buffer of capacity of 20 packets, similar to the simulation in [10, 11, 29, 33]. The discrete-time queue is used to model the traffic flow over the network being simulated. Accordingly, the arrival and departure rates depend on probability values. The values are enumerated to create different traffic load on the simulated network [34]. The arrival rates are chosen to be of the values [0.3, 0.5 and 0.95] and the departure of the value 0.5, which were used in recent AQM evaluations [7, 8, 11, 35]. Three scenarios are created with such arrival and departure probabilities, light traffic, moderate and heavy traffic. The packet arrival process is simulated as a Bernoulli process, and the departure is simulated as geometrically distributed. The simulation components are given in Fig. 7.

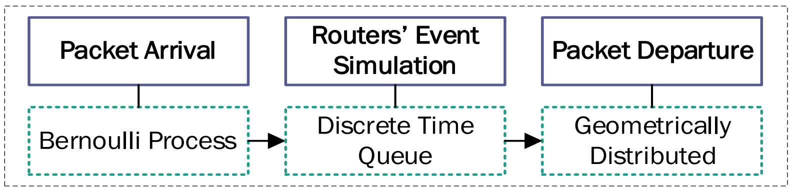
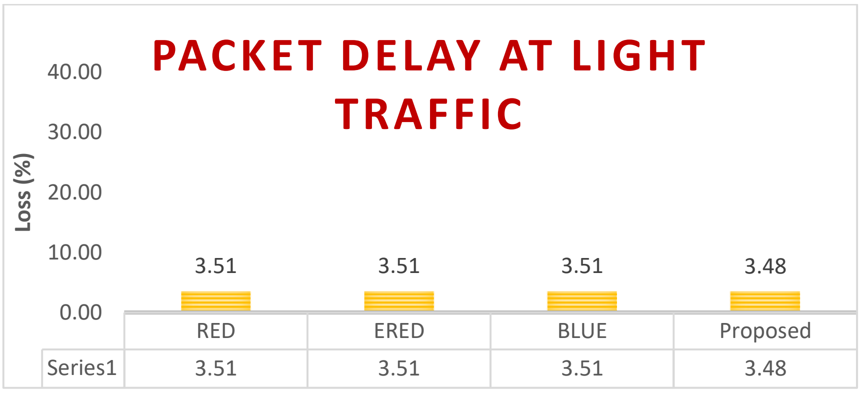


Fig. 7. Simulation components

4. Experimental results

The proposed and compared methods are evaluated in three traffic loads; these are light traffic, moderate and heavy traffic. The comparison is implemented based on delay, dropping rate and packet loss, besides retransmitted packets can give a better indication in the method comparison. Accordingly, it was used as another criterion for method comparison.

In light traffic, as illustrated in Fig. 8, the proposed and compared methods performed similarly with reasonable delay. There was no packet loss nor dropping as the queue is never overflowed with such traffic, and packet dropping is not required. At this sort of traffic, there was no loss or dropping implemented by the proposed and compared methods.

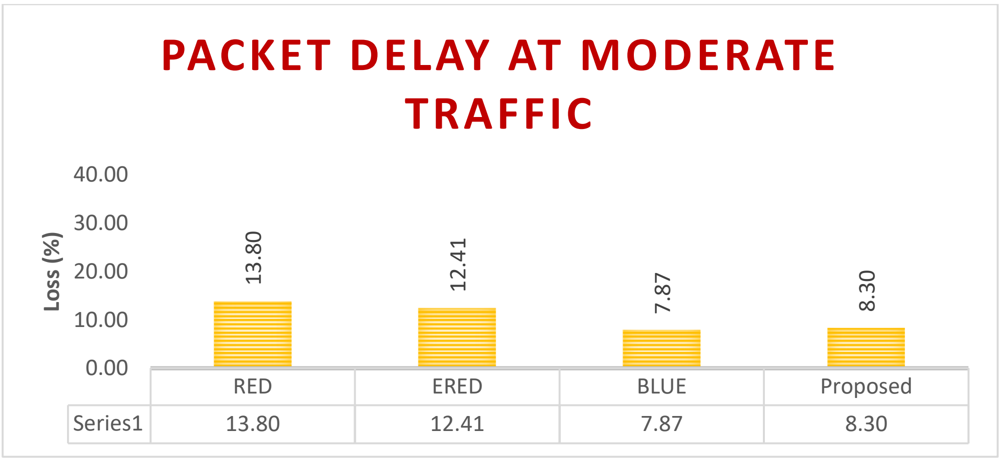


	RED	ERED	BLUE	Proposed
Series1	3.51	3.51	3.51	3.48

Fig. 8. Delay-based comparison for the compared methods in light traffic

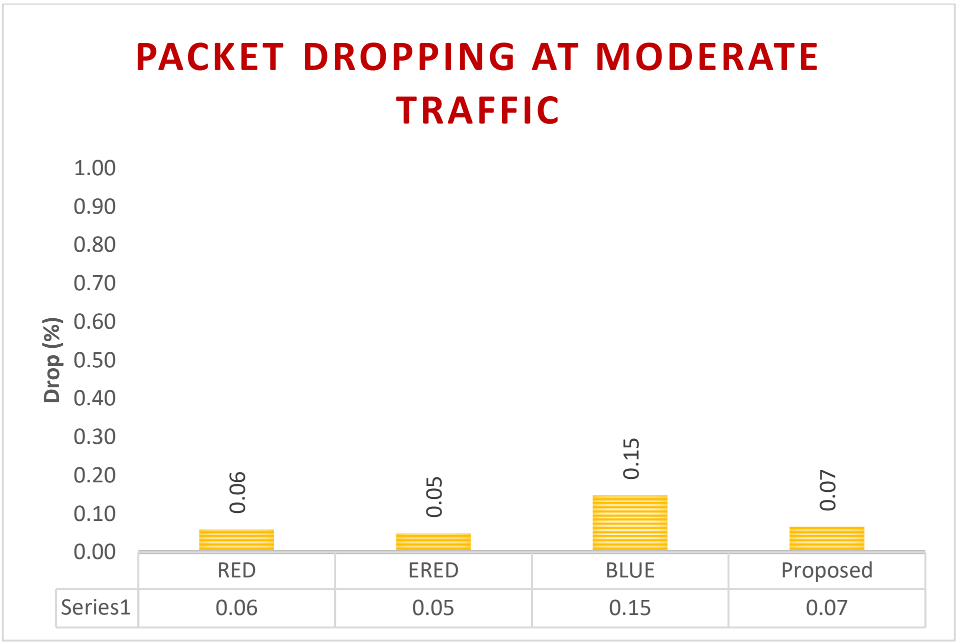
In moderate traffic, the proposed method produced better delay compared to RED and ERED and slightly worse than BLUE, as illustrated in Fig. 9. Yet, BLUE did not produce better performance in total, as BLUE reached this queuing delay because of dropping packets unnecessarily as illustrated in Fig. 10. Similar to the light traffic, there was no packet loss because of using any of the compared methods. Accordingly, the proposed method outperformed the compared methods by dropping a suitable amount of packets to prevent loss and preserve delay. At the same time, RED and ERED dropped slightly fewer packets and sacrificed delay. BLUE drop almost double number of packets compared to the proposed method and produced slightly and insignificantly better delay.

In heavy traffic, as major concern of the AQM methods, the results are close to those in moderate traffic but with clearly distinguished details. The proposed method produced better delay compared to RED and ERED and worse than BLUE, as illustrated in Fig. 11. ERED produce a very high delay that cannot be resisted in some commonly utilized applications like video conferencing. BLUE produce a reasonable delay, yet did not produce better performance in total, as BLUE reached this queuing delay as a result of dropping packets unnecessarily as illustrated in Fig.12. ERED loses a significant amount of data, as illustrated in Fig. 13. Using retransmission as criteria, it is clear that the proposed method is better than BLUE.



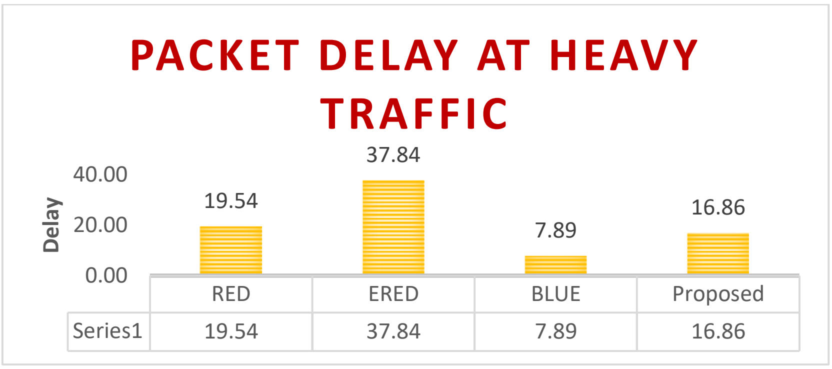
	RED	ERED	BLUE	Proposed
Series1	13.80	12.41	7.87	8.30

Fig. 9. Delay-based comparison for the compared methods in moderate traffic



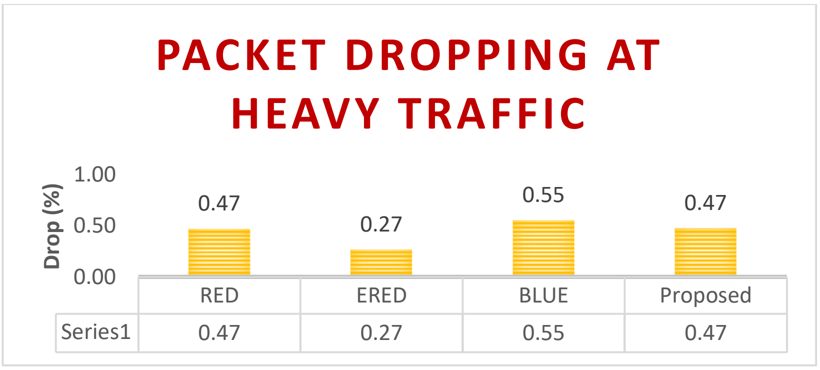
	RED	ERED	BLUE	Proposed
Series1	0.06	0.05	0.15	0.07

Fig. 10. Dropping-based comparison for the compared methods in moderate traffic



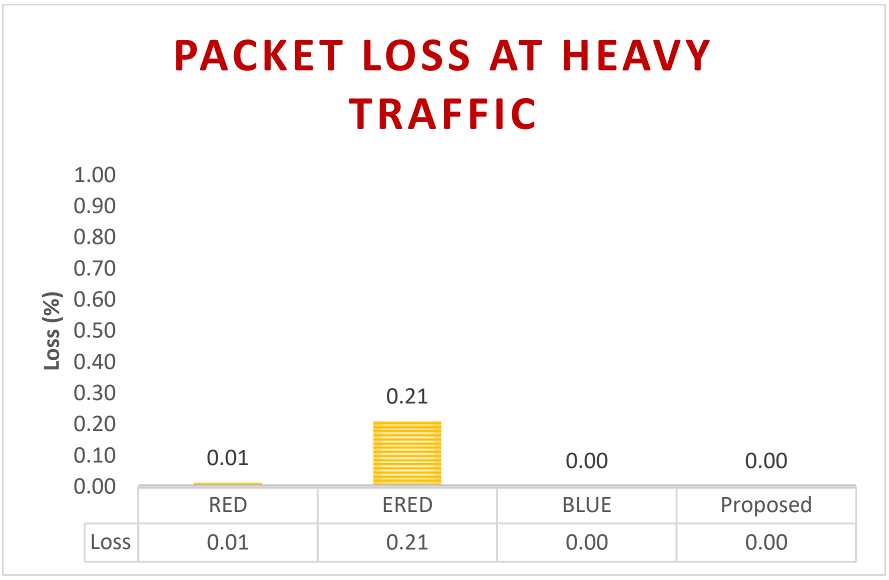
	RED	ERED	BLUE	Proposed
Series1	19.54	37.84	7.89	16.86

Fig. 11. Delay-based comparison for the compared methods in heavy traffic



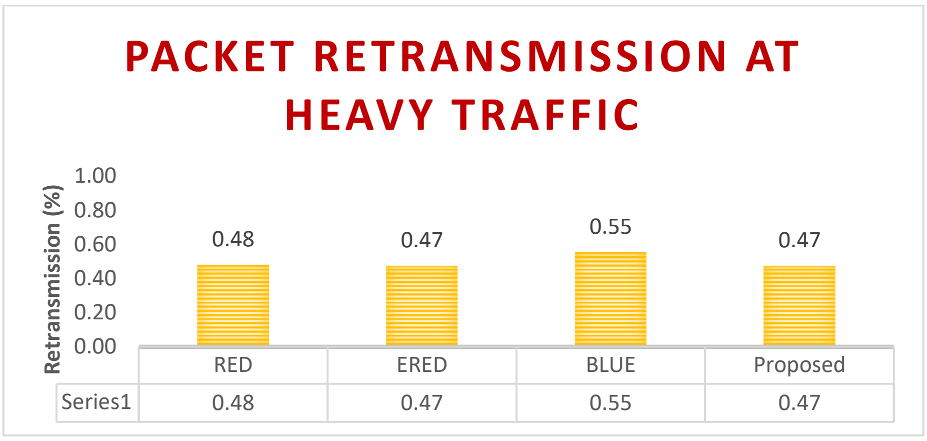
	RED	ERED	BLUE	Proposed
Series1	0.47	0.27	0.55	0.47

Fig. 12. Dropping-based comparison for the compared methods in heavy traffic



	RED	ERED	BLUE	Proposed
Loss	0.01	0.21	0.00	0.00

Fig. 13. Loss-based comparison for the compared methods in heavy traffic



	RED	ERED	BLUE	Proposed
Series1	0.48	0.47	0.55	0.47

Fig. 14. Retransmission-based comparison for the compared methods in heavy traffic

Accordingly, as given in the results, the proposed method preserve packet loss, drop packet only when it is necessary to avoid high dropping rate and drop packet as early as possible to avoid high delay. Compared to the other methods, the proposed method is better than RED, as it produces better results by predicting congestion at an early stage and dropping packets to avoid delay. Besides, the proposed method is better than ERED as it avoids loss by dropping packets only when it is required based on the utilized controllers. Finally, the proposed method is better than BLUE as it avoids high dropping rate while preserving loss by dropping packets and avoid false congestion, which cannot be avoided in BLUE as results suggest. Having high dropping indicates that BLUE is sensitive to false congestion that leads to high packet dropping.

5. Conclusion

A new multi-indicators Active Queue Management (AQM) method is proposed in this paper. The goal of the proposed method is to optimize the performance of the AQM according to the commonly utilized performance metrics. Various steps have been implemented to achieve the intended goals starting with analyzing the performance metrics and identifying the input variables. The fuzzy system is implemented based on space partitioning approach, full-set of rules and center of gravity. The results illustrate that the proposed method preserve packet loss, drop packet only when it is necessary to avoid high dropping rate and drop packet as early as possible to avoid high delay. The proposed method also detects congestion at an early stage and recognize false congestion. The future work will focus on improving the performance based on the quality of demands. Different applications require different demands, such as reducing delay, reducing loss, etc. As such, the input variables will be given different weight by adjusting the rule-set. The system will be reconstructed based on the newly constructed rule-set, and the results will be compared with the results obtained in this paper.

Acknowledgments**:** We appreciate the Research Deanship of Al-Ahliyya Amman University (AAU) – Jordan, for providing research resources, equipment and various research programs to encourage research among faculty members.

References

1. Zhao, Y., Z. Ma, X. Zheng, X. Tu. An Improved Algorithm of Nonlinear RED Based on Membership Cloud Theory. – Chinese Journal of Electronics, Vol. **26**, 2017, No 3, pp. 537-543.
2. Yu-hong, Z., Z. Xue-feng, T. Xu-yan. Research on the Improved Way of RED Algorithm S-RED. – International Journal of u-and-e- Service, Science and Technology, Vol. **9**, 2016, No 2, pp. 375-384.
3. Briscoe, B. Insights from Curvy Random Early Detection (RED). 2015.
4. Jamil, S., N. Alipasandi, B. Alipasandi. An Improvement over Random Early Detection Algorithm: A Self-Tuning Approach. – Journal of Electrical and Computer Engineering Innovations (JECEI), Vol. **2**, 2014, No 2, pp. 57-61.
5. Floyd, S., V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. – IEEE/ACM Trans. Netw., Vol. **1**, 1993, No 4, pp. 397-413.
6. Sharma, N., S. S. Rajput, A. K. Dwivedi, M. Shrmali. P-RED: Probability Based Random Early Detection Algorithm for Queue Management in MANET. – In: Advances in Computer and Computational Sciences. Springer, 2018, pp. 637-643.
7. Abu-Shareha, A. A. Enhanced Random Early Detection Using Responsive Congestion Indicators. – International Journal of Advanced Computer Science and Applications (IJACSA), Vol. **10**, 2019, No 3, pp. 358-367.
8. Abu-Shareha, A. A. Controlling Delay at the Router Buffer Using Modified Random Early Detection. – International Journal of Computer Networks & Communications (IJCNC), Vol. **11**, 2019, No 6, pp. 63-75.
9. Lin, D., R. Morris. Dynamics of Random Early Detection. – In: Proceeding of the ACM SIGCOMM'97 Conference on Applications Technologies, Architectures, and Protocols for Computer Communication, 1997, pp. 127-137.
10. Baklizi, M., H. Abdel-Jaber, A. A. Abu-Shareha, M. M. Abualhaj, S. Ramadass. Fuzzy Logic Controller of Gentle Random Early Detection Based on Average Queue Length and Delay Rate. – International Journal of Fuzzy Systems, Vol. **16**, 2014, No 1, pp. 9-19.
11. Abualhaj, M. M., A. A. Abu-Shareha, M. M. Al-Tahrawi. FLRED: An Efficient Fuzzy Logic Based Network Congestion Control Method. – Neural Computing and Applications, Vol. **30**, 2018, No 3, pp. 925-935.
12. Chen, W., S. H. Yang. The Mechanism of Adapting RED Parameters to TCP Traffic. – Computer Communications, Vol. **32**, 2009, No 13, pp. 1525-1530.
13. Seifaddini, O., A. Abdullah, H. Vosough. RED, GRED, AGRED Congestion Control Algorithms in Heterogeneous Traffic Types. – In: International Conference on Computing and Informatics, 2013.
14. Ismail, A. H., A. El-Sayed, Z. Elsaghir, I. Z. Morsi. Enhanced Random Early Detection (ENRED). – International Journal of Computer Applications, Vol. **92**, 2014, No 9.
15. Floyd, S. Recommendations on Using the Gentle Variant of RED. 2000. **http://www.aciri.org/floyd/red/gentle.html**
16. Floyd, S., R. Gummadi, S. Shenker. ICSI Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. – AT&T Center for Internet Research at ICSI, 2001.
17. Aweya, J., M. Ouellette, D. Y. Montuno. A Control Theoretic Approach to Active Queue Management. – Computer Networks, Vol. **36**, 2001, No (2-3), pp. 203-235.
18. Zheng, B., M. Atiquzzaman. DSRED: An Active Queue Management Scheme for Next Generation Networks. – In: Proc. of 25th Annual IEEE Conference on Local Computer Networks. LCN 2000, IEEE, pp. 242-251.

19. K u n n i y u r, S., R. S r i k a n t. End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks. – IEEE/ACM Transactions on Networking, Vol. **11**, 2003, No 5, pp. 689-702.
20. L o n g, C. N., B. Z h a o, X. P. G u a n. SAVQ: Stabilized Adaptive Virtual Queue Management Algorithm. – IEEE Communications Letters, Vol. **9**, 2005, No 1, pp. 78-80.
21. Y a n p i n g, Q., L. X i a n g z e, L. Q i, J. W e i. A Stable Enhanced Adaptive Virtual Queue Management Algorithm for TCP Networks. – In: 2007 IEEE International Conference on Control and Automation, 2007, pp. 360-365.
22. A t h u r a l i y a, S., V. H. L i, S. H. L o w, Q. Y i n. REM: Active Queue Management. – Teletraffic Science and Engineering, Vol. **4**, 2001, pp. 817-828.
23. H o l l o t, C. V., V. M i s r a, D. T o w s l e y, W. B. G o n g. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. – In: Proc. of IEEE INFOCOM 2001, Conference on Computer Communications, Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No 01CH37213), Vol. **3**, pp. 1726-1734.
24. F e n g, W. C., D. K a n d l u r, D. S a h a, K. S h i n. BLUE: A New Class of Active Queue Management Algorithms. Technical Report CSE-TR-387-99, University of Michigan, 1999.
25. K o o, J., B. S o n g, K. C h u n g, H. L e e, H. K a h n g . MRED: A New Approach to Random Early Detection. – In: Proc. of 15th International Conference on Information Networking, IEEE, 2001, pp. 347-352.
26. A b b a s o v, B., S. K o r u k o g l u. Effective RED: An Algorithm to Improve RED's Performance by Reducing Packet Loss Rate. – Journal of Network and Computer Applications, Vol. **32**, 2009, No 3, pp. 703-709.
27. C h e n, J., C. H u, Z. J i. Self-Tuning Random Early Detection Algorithm to Improve Performance of Network Transmission. – Mathematical Problems in Engineering, Vol. **2011,** 2011.
28. O t t, T. J., T. V. L a k s h m a n, L. H. W o n g. Sred: Stabilized Red. – In: IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. 11th Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No 99CH36320), Vol. **3**, 1999, pp. 1346-1355.
29. B a k l i z i, M., H. A b d e l - J a b e r, M. M. A b u - A l h a j, N. A b d u l l a h, S. R a m a d a s s, S. A. A L m o m a n i. Dynamic Stochastic Early Discovery: A New Congestion Control Technique to Improve Networks Performance. – International Journal of Innovative Computing, Information and Control, Vol. **9**, 2013, No 3, pp. 1118-1126.
30. Y a g h m a e e, M. H., H. A m i n T o o s i. A Fuzzy Based Active Queue Management Algorithm. – Simulation Series, Vol. **35**, 2003, No 4, pp. 458-464.
31. Y u a n, Y., M. J. S h a w. Induction of Fuzzy Decision Trees. – Fuzzy Sets and Systems, Vol. **69**, 1995, No 2, pp. 125-139.
32. N e g n e v i t s k y, M., A. I n t e l l i g e n c e. A Guide to Intelligent Systems. – In: Artificial Intelligence. 2nd Edition. Pearson Education, 2005.
33. A b d e l -j a b e r, H., J. A b a b n e h, F. T h a b t a h, A. M. D a o u d, M. B a k l i z i. Performance Analysis of the Proposed Adaptive Gentle Random Early Detection Method under Noncongestion and Congestion Situations. – In: International Conference on Digital Enterprise and Information Systems, Berlin, Heidelberg, Springer, 2011, pp. 592-603.
34. K h a t a r i, M., G. S a m a r a. Congestion Control Approach Based on Effective Random Early Detection and Fuzzy Logic. – arXiv preprint arXiv:1712.0424, 2017.
35. M o h a m m e d, H., G. A t t i y a, S. E l - D o l i l. Active Queue Management for Congestion Control: Performance Evaluation, New Approach, and Comparative Study. – International Journal of Computing and Network Technology, Vol. **5**, 2017, No 2, pp. 37-49.

Received: 17.11.2020; Second Version: 13.01.2021; Accepted: 23.02.2021