FLACC: Fuzzy Logic Approach for Congestion Control

Mahmoud Baklizi
The world Islamic science and education university W.I.S.E
Department of Computer Networks Systems, Amman, Jordan

***Abstract*—The popularity of network applications has increased the number of packets travelling within the routers in networks. The movement expends most resources in such networks and consequently leads to congestion, which worsens the performance measures of networks, such as delay, packet loss and bandwidth. This study proposes a new method called Fuzzy Logic Approach for Congestion Control (FLACC), which uses fuzzy logic to decrease delay and packet loss. This method also improves network performance. In addition, FLACC employs average queue length (aql) and packet loss (PL) as input linguistic variables to control the congestion at early stages. In this study, the proposed and compared methods were simulated and evaluated. Results reveal that fuzzy logic Gentle Random Early Detection (FLGRED) showed better performance results than Gentle Random Early Detection (GRED) and GRED Fuzzy Logic in delay and packet loss and when the router buffer was in heavy congestion.**

Keywords—Congestion; Network Result Performance; GREDFL

I. Introduction

Computer networks are utilised in organizations, homes and offices. This situation motivates the movement of huge data within computer networks around the world [1-3]. The data are transferred in the form of small pieces called packets. The generated packets form current application data that are temporarily stored during their transfer in routers and switch to network resources. Every router resource has a buffer to queue and stores the packets before transferring to their destination. However, when the available space resources of the network router buffer are not all occupied and thus leave enough space for new packets, incoming packets can drop directly [4, 5]. This situation causes congestion, which happens when arriving packets at router surpass the existing resources [6, 7]. One of the oldest methods that manage and control congestion is tail-drop, which works in the router buffer [8, 9]. The tail-drop method uses first-in-first-out (FIFO) module and manages the congestion at the buffer when the buffer reaches the limit. When incoming packets arrive at the buffer at little amounts, the tail-drop method effectively operates with good performance. However, the method fails when high traffic exists, as the router buffer quickly overflows and every arriving packet can be dropped [10]. Fig. 1 illustrates the tail-drop method buffer.

Generally, the disadvantages of existing congestion control algorithms can be summarized as follows. Existing algorithms use static probability for packet dropping, and several propose an addition target value that leads to a large number of packet drops when the probability value is high and bursting traffic is present. However, the parameterization problem still exists in most dynamic methods. Bursting traffic causes a heavy congestion signal, which then leads to significant packet drops. Conversely, network performance becomes degraded when the probability of packet dropping is set too low. Specifically, Dp, PL, mql, and D increases, and T decreases. Consequently, a dynamic mechanism is required to implement packet dropping based on the congestion status. This paper proposes an enhance method, Stabilized Dynamic GRED (SDGRED), to address the aforementioned disadvantages and to improve network. When the high traffic arrive at the router buffer (Bursty Traffic) causes many limitation and problem [11]. Such as; 1) increase the queuing time delay because the packet spends a lot of time inside the router buffer; 2) Increase the packet loss especially when the heavy congestion is present; 3) Overwhelmed the link bandwidth because retransmitting the packet has been dropped from the router buffer.

Unfortunately, the tai-drop method failed to control the congestion at the buffer. Therefore, many Active Queue Management Method (AQM) have been proposed to detect and eliminate the bursty traffic problem that exists in previous method [1, 12, 13]. In addition, the main goal of the AQM methods discover and manage the congestion at early before the router buffer reaches to the limit size. On the other hand, AQM methods employed dropping probability DP to control the congestion at the buffer by dropping the packets earlier overflow. Consequently, the AQM methods decrease the packet loss and queuing delay by random dropping strategy. Many researchers have been developed and enhanced the AQM methods. Such as, random early detection (RED) [14], GRED [15], Adaptive GRED [16], Dynamic GRED [10], and ENAGRED [17]. Many AQM methods using fuzzy logic, such as GREDFL [18], FLRED, Enhanced Random Early Detection using Responsive Congestion Indicators [19] and REDFL [20]. Though, the current AQM methods cannot predict the congestion at early stage in effective manner, as a result decrease performance of the network [21, 22]. Current paper a new method proposed, to discover the congestion earlier to detect the problems that appear in trail-drop method and enhance the network performance. Especially, delay and packet loss performance measures. In addition, solve the problem of parameterization AQM methods. The proposed method uses the fuzzy logic to utilizing the router buffer by using (aql) and (PL).

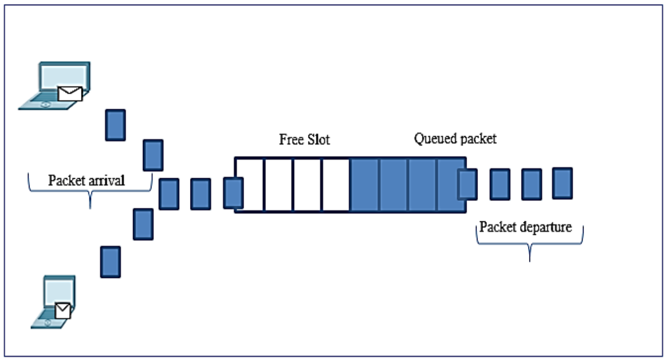

Fig. 1. Tail-Drop Router Buffer.

The current paper is planned as the following. Section 2 shows the literature review work regard to AQM method. Section 3 explains the FLACC method and stages of implementation. Section 4 displays the simulation environment and the parameter that used in the proposed method. Section 5 discuss the performance result of the proposed and compared method. Section 6 includes the conclusion of the current paper. High traffic at the router buffer (Bursty traffic) results in problems, such as 1) increased queuing time delay because packets spend time inside the router buffer; 2) increased packet loss, especially during heavy congestion; and 3) overwhelmed link bandwidth because retransmitting packets has been dropped from the router buffer [11]. Unfortunately, the tail-drop method fails to control the congestion at the buffer. Therefore, Active Queue Management (AQM) methods have been proposed to detect and eliminate the bursty traffic problem that exists in the previous method [1, 12, 13]. The main goal of AQM methods is to discover and manage the congestion early or before the router buffer reaches the limit size. AQM methods employ dropping probability (DP) to manage the congestion at the buffer by dropping the packets early in the overflow. Consequently, AQM methods decrease the packet loss and queuing delay by using the random dropping strategy. Many researchers have developed and enhanced AQM methods, such as random early detection (RED) [14], GRED [15], Adaptive GRED [16], Dynamic GRED [10] and ENAGRED [17]. Many AQM methods use fuzzy logic, such as Gentle Random Early Detection Fuzzy Logic (GREDFL) [18], FLRED and REDFL [20]. The inability of current AQM methods to predict congestion at early stages effectively decreases network performance [21, 22]. Thus, this study proposes a new method to discover the congestion early, detect the problems that appear in the trail-drop method and enhance network performance. Delay and packet loss performance measures are employed. In addition, the proposed method can solve the parameterisation problem of AQM methods. The proposed method uses fuzzy logic in utilising the router buffer by using (aql) and (PL).

II. Related Work

Congestion is a main question in networks. Thus, several scientists have developed and enhanced methods related to congestion control. RED was proposed in 1993 [14]. This method is a mechanism that controls congestion at the quick stage in the buffer and solves and overcomes limitations of the tail-drop method (see Fig. 2).

RED method uses a dropping policy to manage the congestion and discover it before the router buffer becomes full. Subsequently, RED method has two thresholds (minthreshold and maxthreshold) defined at the router buffer and employs an aql calculated value. The packet is dropped according to the following rules. When the aql is less than the minthreshold, no packets are dropped. When the aql is between the minthreshold and maxthreshold, the packets start to drop in low probability. Finally, when the packets are greater than the maxthreshold, the dropping value becomes one and every packet that arrives may overflow [14]. Hence, RED method improves the performance result better than the tail-drop method and decreases the bursty traffic problem.

GRED is another AQM method [15] that enhances RED method, solves the parameterisation problem and decreases Bursty traffic [22, 23]. Unlike RED, GRED method uses three thresholds (minthreshold, maxthreshold and doublemaxthreshold) in different positions at the router buffer to control congestion by using two different dropping probabilities (see Fig. 3 for GRED router buffer).

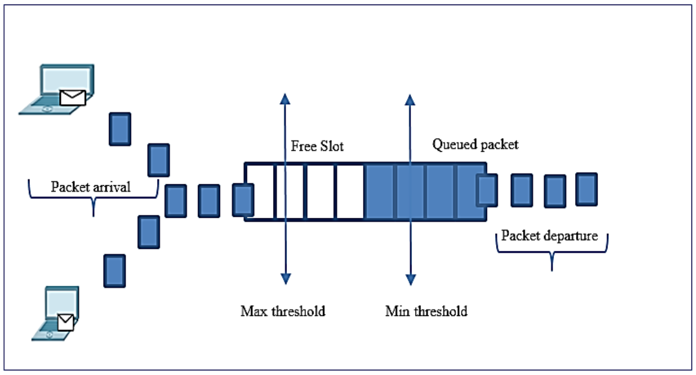

Fig. 2. RED Router Buffer.

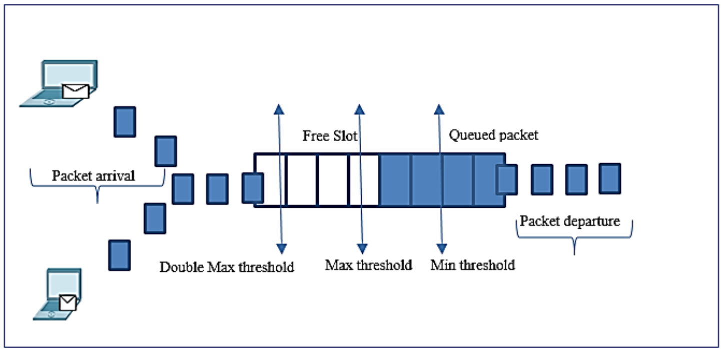

Fig. 3. GRED Router Buffer.

The dropping operates within the following rules. When the aql value is less than the minthreshold, no dropping occurs. If the aql value is between the minthreshold and maxthreshold, GRED method drops the packet according to the equation that is similar to the RED method [14]. When the aql value is between maxthreshold and double maxthreshold, GRED uses another dropping equation with high probability to prevent the router buffer from becoming full. Finally, when the aql is greater than the doublemaxthreshold, the dropping increases and reaches 1 as that of the RED method [15]. Subsequently, every packet that arrives at the router buffer overflows. The result reveals that GRED method outperforms RED, as the doublemaxthreshold has been added. DGRED method is proposed in [10]. Similar to GRED method, DGRED has the same number of thresholds. Unlike GRED method, DGRED employs adaptive threshold positions to enhance the performance and keeps the aql value at the level between minthreshold and maxthreshold by using target aqls assigned at specific levels. As revealed, all previous AQM methods used aql. Therefore, the aql value can sometimes be high but can be low in actual queue length. In addition, according to the dropping policy, AQM methods based on the aql value. Once the router buffer handles traffic quickly, the real queue length decreases. If aql is great, formerly received packets are dropped even if the actual queue length is slight [17, 24]. However, most AQM methods use thresholds and target aql, T(aql), which leads to parameterisation problems [18]. To avoid the parameterisation problems that appear in AQM methods, fuzzy logic is agreed upon as a solution for congestion in the control network [25]. Fuzzy logic does not need to deal with parameters [21]. The first use of fuzzy logic was with ATM networks, which developed network performance and decreased packet loss.

Baklizi et al. proposed GREDFL, which is an enhanced AQM GRED method. GREDFL employees' aql and D as input variables for fuzzy logic. Therefore, the GREDFL method enhances the GRED method whilst reducing the widespread of parameter dependency in it. GREDFL uses the same simulation that is used in the AQM GRED method, which uses a discrete time queue model and FIFO packet arrival probability. The performance result for GREDFL method outperforms that for GRED method.

Many other methods use fuzzy logic to improve network performance. Conversely, no method can handle the congestion at early stages to have better performance than the current AQM methods [3, 26, 27]. In this study, FLACC uses two input linguistic variables (aql and PL), unlike GREDFL that uses aql and delay PL. Packet loss is considered one of the most important measures that plays a min role in wresting network performance. Furthermore, the proposed FLACC avoids parameterisation problems and improves packet loss and delay performance measure to gain better performance results than current AQM methods.

III. Proposed FLACC Method

This part presents the design and implementation of the proposed FLACC. Fuzzy logic has been used in FLACC to solve and decrease the limitation of congestion. Fuzzy logic is mainly employed to detect and prevent parameterisation problems in most existing AQM approaches. Also, the proposed method employs aql and PL as congestion indicators to enhance the performance of AQM methods. aql and PL play main roles in expecting the congestion in the buffer. The main goal is to calculate the DP for all arriving packets in the router buffer by using a fuzzy inference process (FIP). To calculate DP, aql and PL are used. Fig. 3 illustrates FIP to calculate the dropping.

The FLACC router buffer is designed without the thresholds existing in AQM methods. Hence, FIP is used to generate the dropping probability, which employs two variables as inputs, aql and PL, to gain one output called DP (see Fig. 4).

The FLACC router buffer is designed without the thresholds existing in AQM methods. Hence, FIP is used to generate the dropping probability, which employs two variables as inputs, aql and PL, to gain one output called DP (see Fig. 5).

The proposed FLACC is designed and implemented to overcome the limitations of current AQM methods and investigate satisfying performance results. This goal is achieved in six stages. Performance measures and parameters are illustrated in Table I.

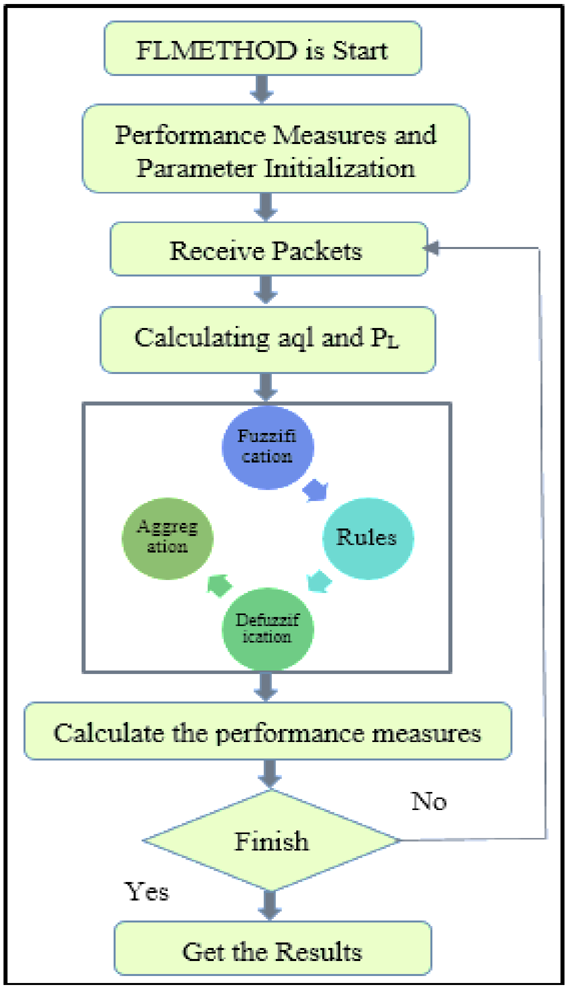
FLMETHOD is Start
Performance Measures and Parameter Initialization
Receive Packets
Calculating aql and P$_L$
Fuzzifi cation
Rules
Defuzzif ication
Aggreg ation
Calculate the performance measures
Finish
No
Yes
Get the Results

Fig. 4. The Router Buffer in FLACC Method.

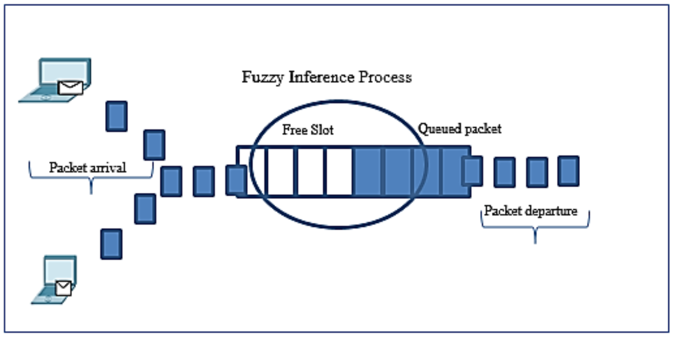

Fig. 5. The Process Stage for the Proposed Method.

TABLE. I. PARAMETERS INITIALIZATION

Parameters	Description	GRED method	GREDFL method	FLACC method
T	Throughput	zero	zero	zero
D	Delay	zero	zero	zero
PL	Packet Loss	zero	zero	zero
mql	Mean Queue Length	zero	zero	zero
DP	Dropping Probability	zero	zero	zero
PA	Packet Arrival	0.33-0.93	0.33-0.93	0.33-0.93
QW	the Weight	2/1000	2/1000	2/1000
Beta	Packet Departure	1/2	1/2	1/2
MP	Maximum Probability	0.1	0.1	0.1
S#	Number of Slot	2millions	2millions	2millions
C	Buffer Capacity	20	20	20
MT	Minimum threshold	3	----	----
MXT	Maximum MT	9	----	----
DMXT	Double MXT	18	----	----

The table is divided into two parts. Some parameters are common with all AQM methods, whereas the second part is customised. Thus, the method depends on fuzzy or AQM.

As mentioned above, on the one hand, some parameter values in Table I are set to equal 0, given that the system in the first stage is idle. On the other hand, the values of other parameters are set similar to those of GRED and FLGRED methods.

In stage two, the router buffer in the proposed method starts to generate the packets, which are moulded using discreet time queue to measure the performance result in a specific time slot. All the input-generated packets in the router are queued in FIFO. In stage three, the performance measures and packet loss are calculated. The aql value depends on whether the router buffer is empty or not. When the buffer is blank, the aql value is considered according to Equation 1.

$$aql = aql \times (1 - q_w)^{\mathrm{n}}, \qquad (1)$$

where n denotes # of packets in idle time.

When the router buffer is not empty, the aql value is calculated according to Equation 2.

$$aql = aql \times (1 - q_w) + q_w \times q\_inst, \qquad (2)$$

where q_inst represents the current queue length in the router buffer.

The value of packet loss is calculated when the router buffer becomes full or overflows. When the incoming packets reach the buffer in Bursty traffic, the outer buffer starts to build itself and reaches the limit of the router buffer size. Therefore, any arriving packet cannot accommodate space in the buffer, and every packet that arrives can drop directly. Equation 3 illustrates the packet loss.

$$PL = (1 - \mathrm{beta}) \times Pk, \qquad (3)$$

where Pk signifies the probability that the buffer is full, and beta denotes the probability of packet leaving or departure.

In stage four, the proposed FLACC fuzzification starts. FIP consists of four steps, namely, fuzzification, rule evaluation, aggregation and defuzzification.

In step 1, fuzzification uses three variables to generate the membership degree. The variables are aql, PL and DP. Each variable contains the following fuzzy set:

aql: {Conservative, Average, Extra}

PL: {Little, Medium, More}

DP: {Slight, Medium, Further}.

The triangular shape is used to represent the input and output variables (see Fig. 6). Fig. 6, 7 and 8 illustrate the membership degrees for the two inputs and one output variable, respectively.

Once the membership degrees are set, the fuzzy set for all input linguistic variables is calculated. The main goal of the fuzzification step is to form the area for each input linguistic variable. In step 2, rules are evaluated. Table II illustrates the rule evaluation for FLACC. Every part is a process that undergoes rule evaluation by obtaining the membership degree for Dp output.

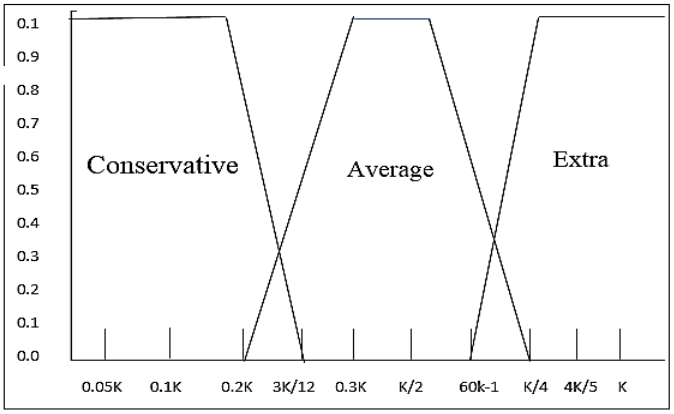

Fig. 6. Membe_rship Function of aql.

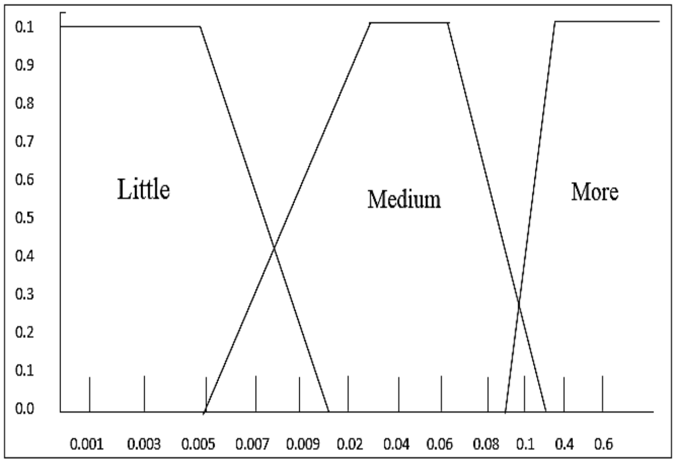

Fig. 7. Member_ship Function of PL.

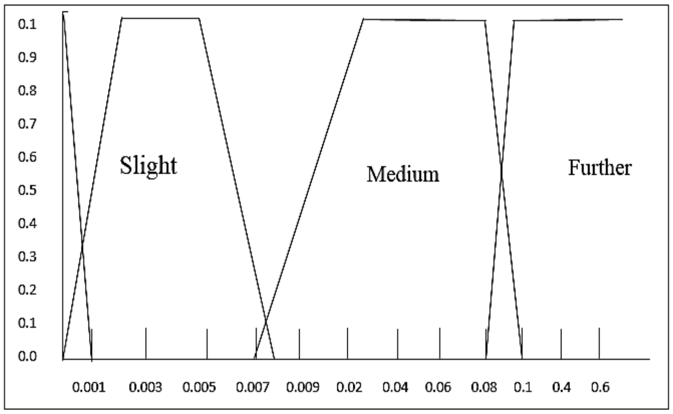

Fig. 8. Member_ship Function of Dp.

TABLE II. SET OF FUZZY LOGIC RULES

IF aql is Conservative and PL is Little THEN Dp is Zero
IF aql is Conservative and PL is Little THEN Dp is Zero
IF aql is Conservative and PL is Little THEN Dp is Zero
IF aql is Average and PL is Medium THEN Dp is Zero
IF aql is Average and PL is Medium THEN Dp is Zero
IF aql Average is and PL is Medium THEN Dp is Slight
IF aql is Extra and PL is More THEN Dp is Zero
IF aql is Extra and PL is More THEN Dp is Medium
IF aql is Extra and PL is More THEN Dp is Further

In step 3, the output rules are aggregated into a single output rule called fuzzy set, whereby the membership degrees in the previous step are combined.

In step 4, defuzzification is the final step in FIP. Crisp values for all the output variables are generated in the fuzzy set.

The centre of gravity (COG) is used in step 4 [ref mos]. The main goal of COG is to find the centre point in the fuzzy set for all output variables. Officially, COG can be set on the basis of Equation (4). Fig. 9 illustrates the algorithm of the proposed FLACC.

$$COG = \frac{\sum_x^y A(n)\times n}{\sum_x^y A(n)} \tag{4}$$

```
The Proposed FLACC Algorithm

   1.  Set Fuzzy sets
   2.  Generate Fuzzy Rules
   3.  For every packet arrive the buffer
   4.  If q is zero THEN aql equal (1-w)f(time- q_time) * aql
   5.  If q <> 0 THEN aql equal (1-w)* aql + w *q
 6. PL = ( 1-Beta ) * PK
 7. Fuzzy (aql)
 8. Fuzzy (PL)
 9. Put on Rules(Fuzzy aql, Fuzzy PL)
10. Aggregattion
11. Calculate dropping
12. Drop packet with Dp
```

Fig. 9. The Algorithm of FLACC.

IV. SIMULATION

This section explains the simulation process prototypical, compared the proposed FLACC with other methods. Discrete-time queue was used in most AQM methods to measure the performance result at any time [28, 29]. The term used was 'time slot,' which means performance measures can be calculated at any time.

The proposed FLACC, FLGRED and GRED methods were realised on a single router buffer in a simulation environment. The router buffer capacity was set to a small value that equals 20 packets. To generate heavy congestion in the router buffer, all methods were implemented as FIFO. The number of slot time used in the simulation environment was 2,000,000 [28]. The first 800,000 slots were used as warm-up and were not tallied in the quantity of the valuation.

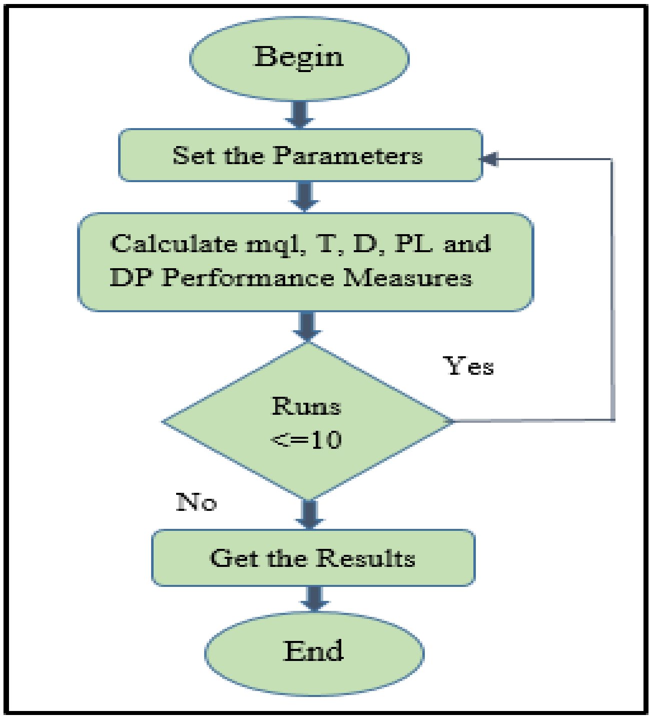

Fig. 10. Simulation Stages.

The simulation used six arrival probabilities (0.18, 0.33, 0.48, 0.63, 0.78 and 0.93), which have been used in several AQM methods [6, 30, 31]. Departure probability was set to 0.5. The proposed FLACC and the compared method were tested and implemented using Java environment under 64 bits Windows 7 (CPU i5 3.10 GHz processor and 4 GB Ram).

In stage 1, the parameters were set to particular values. In step 2, the performance measures were calculated. In step 3, if the runs were less than or equal 10, the process was repeated. Finally, the results were evaluated for all the methods. These stages are illustrated in Fig. 10.

V. Performance Results

The performance evaluation of the proposed FLACC, GREDFL and GRED methods is presented in this section. Furthermore, the proposed method is described in detail according to membership functions, perimeter values and rules set.

The FLACC, GREDFL and GRED methods were measured and compared in five different performance measures (mql, T, D, PL and Dp) to determine the best one.

A. Mean Queue Length

In this section, the performance result of the mean queue length is evaluated. Fig. 11 illustrates the output performances of the proposed FLACC, GREDFL and GRED methods. The experimental is performed using different probabilities for packet arrivals to generate congestion and non-congestion at the router buffer.

The performance measure, mql, investigates the same results for the proposed and compared methods in case the packet arrival probability is less than the packet departure. However, when the incoming packets arrive at high rate probability, such as 0.63, 0.78 and 0.93, FLACC shows a better performance than GREDFL and GRED methods. As a result, FLACC outperforms the AQM compared method when the congestion is present at the router buffer.

B. Throughput

The total number of packets passes the router at a specific time, which is called throughput. The throughput performance measure is an important criterion of computer networks. Fig. 12 illustrates the performance result analysis for the proposed FLACC, GREDFL and GRED methods.

All the methods under the probability of alpha gained the same result, either the packet arrival was less than or greater than the beta. However, all the methods obtained the same value of throughput, which was equal to 0.5. This result was due to the parameter initialisation for the packet departure, as mentioned earlier in the paper.

C. Delay

The performance result evaluation of queueing delay is explained in Fig. 12. The mean queue length in Section 5.1 plays a main role in delay calculation and evaluation. The delay is calculated according to mql and T (see Equation 5).

$$D = mql/T \tag{5}$$

The equation reveals that when the value of mql is small, that of delay is also small.

Fig. 13 illustrates that FLACC outperforms the GREDFL and GRED AQM methods in congestion or non-congestion. When the packet arrival reaches 0.48, 0.78 and 0.93, the proposed method gains enhanced performance result than the compared methods.

D. The Packet Loss

When the received packets reach at the buffer and no space is available, every packet that arrives can drop directly, as the router buffer becomes full and overflows.

Fig. 14 indicates the performance results of FLACC, GREDFL and GRED methods.

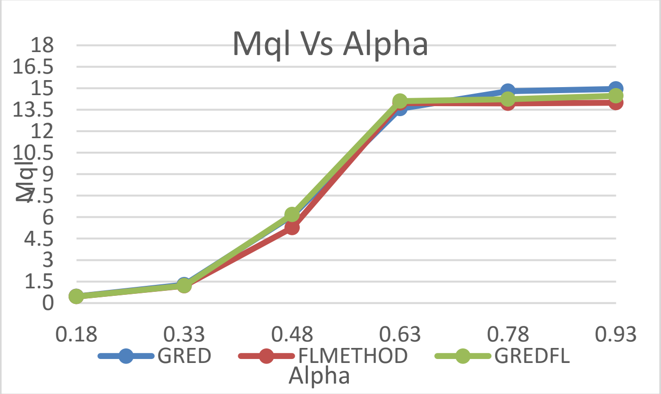

Fig. 11. Mean Queue Length Performance Results.

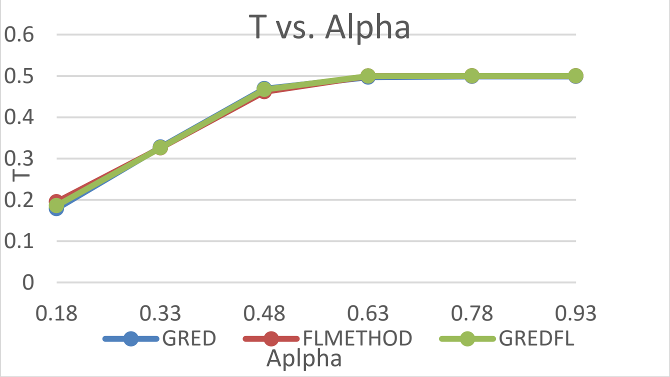

Fig. 12. Throughput Performance Results.

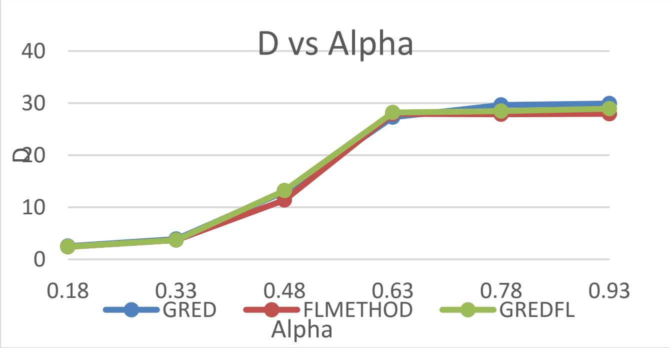

Fig. 13. Delay Performance Results.

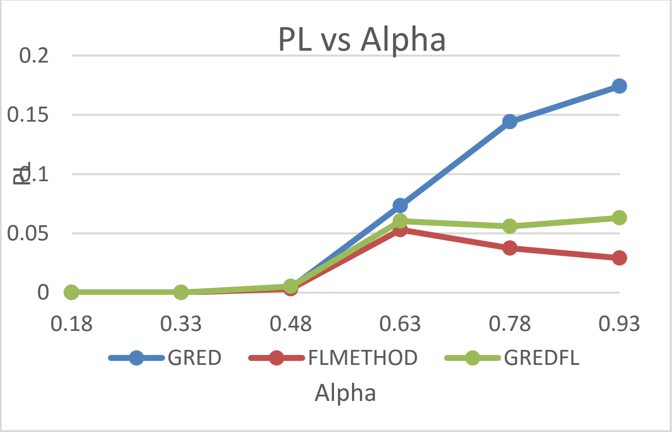

Fig. 14. Packet loss Performance Results.

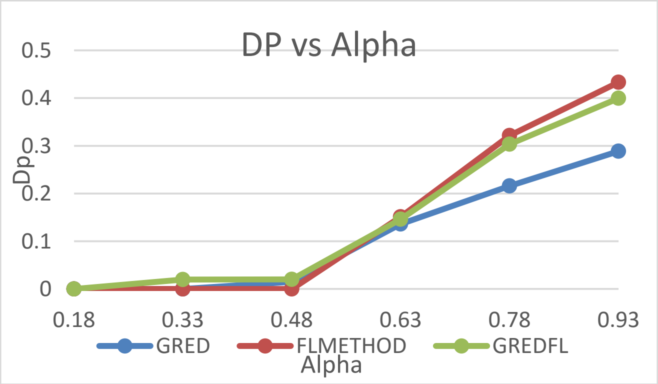

Fig. 15. Dropping Probability Performance Results.

The proposed FLACC has achieved good and least PL performance results, when heavy congestion is present. The router buffer of FLACC overflows less than that of the compared methods, GREDFL and GRED.

E. Dropping Probability

Fig. 15 shows that when congestion exists at the buffer, the FLACC drops more packets at that buffer than GREDFL or GRED methods. Hence, the value of packet arrival is greater than that of packet departure.

Note that the FLACC has lost fewer packets in the router buffer than GREDFL and GRED methods, as presented in the previous section.

VI. Conclusions

One of the main issues that affect computer networks is congestion, which causes the wresting of network resources and performance. Subsequently, the effects include increased queuing buffer, long delays and packet loss. We propose FLACC as a new congestion avoidance method. FLACC is an extension of the GRED AQM method. As parameterisation is a limitation of current AQM methods, FLACC aims to detect and avoid such a problem. aql, PL, and FIP are employed as indicators to discover and prevent congestion at early stages or before the router buffer becomes full. To compare and evaluate their performance results, the proposed FLACC, GREDFL and GRED were tested in a simulated process by using popular processes. The simulation result evaluation showed that FLACC has better performance results than GREDFL and GRED methods when heavy congestion is present. The mql, D and PL decreased with the FLACC, although all the compared methods gained the same T in congestion or no congestion situations. Furthermore, GREDFL and GRED methods dropped fewer packets (Dp) at their router buffers than the proposed FLACC.

VII. Future Work

In future work the current FLACC method applying in wireless area to mechanism the congestion in best effort. To increase the network performance, specially improve the delay and packet loss. In addition, applying the multi class queue using the proposed FLACC method solves the bursty traffic problem.

References

[1] Chandra, E. and B. Subramani, A Survey on Congestion Control. Global Journal of Computer Science and Technology 2010. 9(9): p. 82-87.

[2] Thiruchelvi, G. and J. Raja, A Survey On Active Queue Management Mechanisms. IJCSNS International Journal of Computer Science and Network Security, 2008. 8(12): p. 130-145.

[3] Bazaz, Y., S. kumar, and S. Anand, Congestion Control Mechanismusing Fuzzy Logic. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 2013. 2(3).

[4] M, K. and S. G, Congestion control approach based on effective random early detection and fuzzy logic. MAGNT, 2015. 3(8): p. 180-193.

[5] BAKLIZI, M., et al., Markov-Modulated Bernoulli Dynamic Gentle Random Early Detection. Journal of Theoretical and Applied Information Technology, 2018. 96(20).

[6] Abdel-jaber, H., F. Thabtah, and M. Woodward. Traffic Management for the Gentle Random Early Detection using Discrete-Time Queueing. in Information Management in Modern Organizations: Trends & Challenges 2008.

[7] Negnevitsky, M., Artificial Intelligence: A Guide to Intelligent Systems. Second Edition ed. 2005, England.

[8] Stanojevic, R., R.N. Shorten, and C.M. Kellett, Adaptive tuning of drop-tail buffers for reducing queueing delays. Communications Letters, IEEE, 2006. 10(7): p. 570-572.

[9] Brandauer, C., et al., Comparison of Tail Drop and Active Queue Management Performance for Bulk-Data and Web-Like Internet Traffic, in Proceedings of the Sixth IEEE Symposium on Computers and Communications. 2001, IEEE Computer Society.

[10] Baklizi, M., et al., DYNAMIC STOCHASTIC EARLY DISCOVERY: A NEW CONGESTION CONTROL TECHNIQUE TO IMPROVE NETWORKS PERFORMANCE. International Journal of Innovative Computing, Information and Control, 2013. 9(4).

[11] Welzl, M., Network Congestion Control: Managing Internet Traffic. 1 ed. 2005.

[12] Kalav, D. and S. Gupta, Congestion Control in Communication Network Using RED, SFQ and REM Algorithm. International Refereed Journal of Engineering and Science (IRJES), 2012. 12 (2): p. 41-45.

[13] Fan, X., et al. QBLUE: A New Congestion Control Algorithm Based on Queuing Theory. in High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS), 2012 IEEE 14th International Conference on. 2012.

[14] Floyd, S. and V. Jacobson, Random early detection gateways for congestion avoidance. IEEE/ACM Trans. Netw., 1993. 1(4): p. 397-413.

[15] Floyd, S. Recommendations On Using the Gentle Variant of RED. http://www.aciri.org/floyd/red/gentle.html 2000.

[16] Baklizi, M., et al., Performance Assessment of AGRED, RED and GRED Congestion Control Algorithms. Information Technology Journal, 2012. 11(2): p. 255-261.

[17] Baklizi, M. and J. Ababneh, Performance Evaluation of the Proposed Enhanced Adaptive Gentle Random Early Detection Algorithm in Congestion Situations International Journal of Current Engineering and Technology 2016. 6(5).

[18] Baklizi, m., et al., Fuzzy Logic Controller of Gentle Random Early Detection Based on Average Queue Length and Delay Rate International Journal of Fuzzy Systems, 2014. 16(1).

[19] Abu-Shareha, A.A., Enhanced Random Early Detection using Responsive Congestion IndicatorsInternational Journal of Advanced Computer Science and Applications(IJACSA), 2019. 3(1): p. 358-367.

[20] Abualhaj, M.M., A.A. Abu-Shareha, and M.M. Al-Tahrawi, FLRED: an efficient fuzzy logic based network congestion control method. The Natural Computing Applications, 2016.

[21] Mulla, A.S. and B.T. Jadhav, Fuzzy Based Queue Management Policies– An Experimental Approach. International Journal of Current Engineering and Technology 2014. 4(1).

[22] Baklizi, M., J. Ababneh, and N. Abdallah. Performance Investigations of Flred and Agred Active Queue Management Methods. in Proceedings of Academicsera 13 th International Conference. 2018. Istanbul, Turkey.

[23] Seifaddini, O., A. Abdullah, and A.H. Vosough, Red, Gred, Agred Congestion Control Algorithms in Heterogeneous Traffic Types, In International Conference on Computing And Informatics. 2013.

[24] Baklizi, M., J. Ababneh, and A Survey in Active Queue Management Methods According to Performance Measures. International Journal of Computer Trends and Technology (IJCTT), 2016. 38 (3): p. 145.

[25] Tassiulas, L., Y.C. Hung, and S.S. Panwar, Optimal buffer control during congestion in an ATM network node. Networking, IEEE/ACM Transactions on, 1994. 2(4): p. 374-386.

[26] Kusumawardani, M., Active queue management (aqm) and adaptive neuro fuzzy inference system (anfis) as intranet traffic Control. Academic Research International 2013. 4(5).

[27] Ingoley, S.N. and M. Nashipudi, A Review: Fuzzy Logic in Congestion Control of Computer Network in International Conference in Recent Trends in Information Technology and Computer Science 2012.

[28] Abdel-Jaber, H., et al., Performance evaluation for DRED discrete-time queueing network analytical model. J. Netw. Comput. Appl., 2008. 31(4): p. 750-770.

[29] Woodward, M.E., Communication and Computer Networks: Modelling with discrete-time queues. 1993: Wiley-IEEE Computer Society Press.

[30] Ababneh, j., et al., Derivation of Three Queue Nodes Discrete-Time Analytical Model Based on DRED Algorithm, in The Seventh IEEE International Conference on Information Technology: New Generations (ITNG 2010),USA.2010.2010.

[31] Guan, L., et al., Discrete-time performance analysis of a congestion control mechanism based on RED under multi-class bursty and correlated traffic. Journal of Systems and Software, 2007. 80(10): p. 1716-1725.