Stabilizing Average Queue Length in Active Queue Management Method

Mahmoud Baklizi[1]
The World Islamic Science & Education University W.I.S.E
Department of Computer Networks Systems
Amman, Jordan

***Abstract*—This paper proposes the Stabilized (DGRED) method for congestion detection at the router buffer. This method aims to stabilize the average queue length between allocated minthre_shold and doublemaxthre_shold positions to increase the network performance. The SDGRED method is simulated and compared with Gentle Random Early Detection (GRED) and Dynamic GRED active queue management methods. This comparison is built on different important measures, such as dropping probability, throughput, average delay, packet loss, and mean queue length for packets. The evaluation aims to identify which method presents better simulation performance measurement results when non-congestion or congestion situations occur at the router buffers in congestion control. The results show that at high packet arrival probability, the proposed algorithm helps provide lesser queue length values, delayed time, and packet loss compared with current methods. Furthermore, SDGRED generates adequate throughput at high packet arrival probability.**

***Keywords*—Congestion control methods; GRED; dynamic GRED; random; simulation; active queue management method**

I. Introduction

The worldwide broadcast of computer networks connects a huge number of devices, from personal computers to multi-branch organization networks [1, 2]. Enormous amounts of data are sent and received between network devices in the form of packets. When several senders send the data over the same intermediary link, packets are stored in the routers' buffer and spend a lot of time waiting for transmitted. However, in view of the buffer size disadvantages in whole network resources [3-5], incoming packets are dropped after the number of packets more than the resource size of the router buffer. Fig. 1 illustrates a possibly congested router buffer. Every packet arriving at the router buffer is considered overflow and dropped as well as causing congestion [6-8]. Of average delay (D) and mean queue length (mql) of packets in the router buffer which also decreases the amount of packets going in the router buffer (T) [9-11].

Enormous congestion control algorithms, such as Gentle Random_Early_Detection (GRED) [12], Enhanced Adaptive GRED (EAGRED)[13] and Markov-Modulated Bernoulli Dynamic GRED [6] have been proposed. However, these algorithms have failed to adjust dynamically to provide the best solution based on the mql status.

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 performance. The latter objective involves alleviating PL and obtaining more acceptable performance measurement results with regard to D and mql when heavy congestion takes place at the router buffers [14].

The paper is summarized as follows. Related work is presented in Section 2. The proposed SDGRED method is covered in Section 3. Section 4 presents the details of simulation experimental environment. The performance results of the developed simulation are discussed in Section 5. Section 6 presents the summery of the proposed paper.

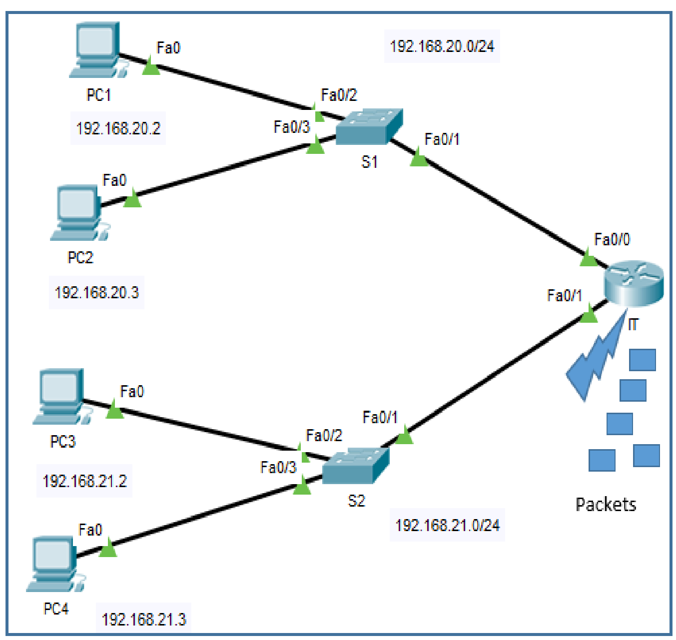

Fig. 1. Congestion in Router Buffer.

II. Related Work

Several studies have explored controlling congestion and handling the aforementioned problems [15-20]. The Drop-Tail (DT) method [21] aims to control congestion employing a stable router buffer size to optimize queuing delay. The size of the router buffers is set to a maximum and all incoming packets are dropped when the router buffers overflow. There are several disadvantages of DT. Such as, increase the packet delay, decreases the throughput (T), an increase in the packet loss rate, and global synchronization [22].

Average Queue Management Methods (AQM) methods are a solution to overwhelm drawbacks of DT method. Unlike the DT method that starts dropping packets only after the router buffers overflow [13, 14, 23], AQM methods are depend on dropping the packets in the router buffer in early stages. So, early congestion control mechanism notifies the sources sender to start decrease their transmission packets early before the buffers are occupied completely and becomes full. AQM methods control the congestion in the router buffer, so as to increase the throughput, decreases the time delay, decreases the packet loss values, and keeps mql at a lowest value. AQM emerges with an adaptable utilization buffer size. Packet droppings are initiated based on a calculated threshold value to prevent buffer overflow. AQM calculates the current value of aql according to the number of packets then compares it.

(AQM) methods are a solution to overwhelm drawbacks of DT method. Unlike the DT method that starts dropping packets only after the router buffers overflow [13, 14, 23], AQM methods are depend on dropping the packets in the router buffer in early stages. So, early congestion control mechanism notifies the sources sender to start decrease their transmission packets early before the buffers are occupied completely and becomes full. AQM methods control the congestion in the router buffer, so as to increase the throughput, decreases the time delay, decreases the packet loss values, and keeps mql at a lowest value. AQM emerges with an adaptable utilization buffer size. Packet droppings are initiated based on a calculated threshold value to prevent buffer overflow. AQM calculates the current value of aql according to the number of packets then compares it with the set threshold. When the aql value arrives to defined threshold, all the incoming packets that arrive at the router buffer are dropped with main factor called the probability in order to preventing router buffer overflow and becomes full [24-26]. Enormous methods for congestion control, such as AGRED [22], EAGRED [27], DGRED [14], and FLGRED have been built based on AQM.

Floyd [27] proposed the GRED to make less some disadvantages in random early detection (RED)[12]. Comparable to RED, the GRED method chiefly purposes to control the congestion in router buffer at an early stage. GRED implements its algorithm by stabilizing the aql at a certain level. GRED uses a familiar approach used by RED in calculating the dropping. Conversely, GRED uses minimum, maximum, and double maximum threshold. Commonly, GRED responds to the arriving packets at router buffer according to the subsequent steps (Fig. 2):

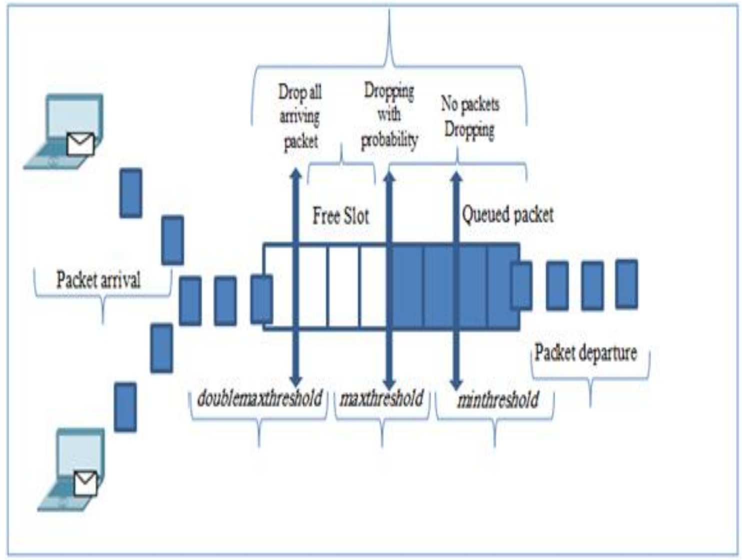

Fig. 2. GRED Buffer.

- When the current value of aql is less than the minthre_shold value, allow receiving packets.
- When the aql is greater than the minimumthreshold value and less than maxthre_shold value, the GRED method start drop the packets in the router buffer in random manner, as RED.
- When the aql is reach the maxthre_shold value and less than the doublemaxthre_shold value, the GRED method start drops the packets with based on higher probability scenario.
- Finally, if the aql value is arriving the doublemaxthre_shold value, the GRED method drops every arriving packets and the Dp is set one.

However, GRED has several disadvantages. Such as, GRED contains numerous threshold values, GRED parameters are set to exact values to gain satisfactory performance. This cusses parameterization problem. And when the current aql value is below the minthre_shold value and heavy congestion occurs in the router buffer, the aql will take a long time to modify; the result the router buffer overflows and becomes full. Therefore, no dropping for packets even with the overfull GRED router buffer.

Dynamic GRED (DGRED) is a development of GRED method. DGRED uses a dynamic maxthre_shold position and doublemaxthre_shold to control the dropping policies mechanism in the router buffer at the early time earlier it overflows[14]. This algorithm aims to stabilize the aql value at the router using an original defined value called Target aql (Taql) that is calculated and set between the minthre_shold and maxthreshold. In addition, the proposed DGRED intends to provide better performance results than other AQM methods, such as RED and two of its variants, GRED and AGRED[14]. These results are represented by the results of mql, packet loss, and delay when congestion happens at the buffer, see Fig. 3.

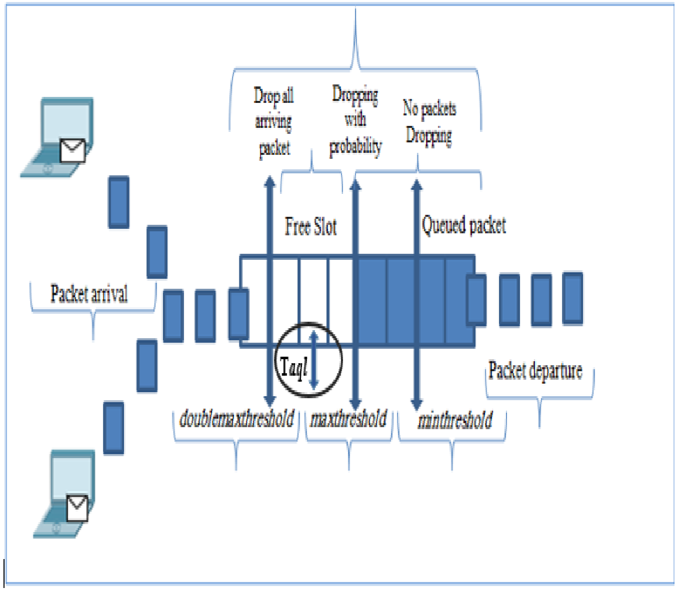

Fig. 3. DGRED Buffer.

DGRED furthermore changes the maxthre_shold and doublemaxthre_shold parameters setting at the buffer to improve performance measures. DGRED method employs minthre_shold and doublemaxthre_shold as set in GRED method.

However, DGRED has some limitations. DGRED involves several threshold values and Taql, which means using parameters to control the congestion in the router buffer (parameterization).

III. Proposed SDGRED Method

Fig. 4 shows SDGRED's processing stages. The parameter setting initialization step (Step 1) ensures that parameters are actually specified when the packets reach the router buffer. SDGRED method uses the minthre_shold and maxthre_shold values as that in DGRED method [14]. The doublemaxthre_shold in SDGRED method is considered the same value as that in DGRED[14], see Fig. 5. The initial value of aql is set zero and the counter sequence value starts from–1.

The SDGRED method then receives packets (Step 2) using a Bernoulli model, ∈ [9], n = 0,1,2,3,4,5 …., wherever n refers to the arrival packets number in the router buffer in specific slot n. the Bernoulli process is appropriate when the buffer has a static length slot.

SDGRED then observes the queue status in the router buffer (Step 3) and calculates the aql value depend on status the buffer either contain packets or not contain packets, as shown in Fig. 6. Thus, in the case of empty router buffer queue, the aql value is considered according to idle time (n) and computed using Equation (1). Meanwhile, in the case of router buffer queuing, the aql is computed using Equation (2).

$$average\ ql = average\ \mathrm{ql} \times (1 - average\ \mathrm{w})^{\mathrm{n}} \quad (1)$$

$$average\ ql = average\ ql \times (1 - \mathrm{qw}) + \mathrm{qw} \times \mathrm{q\_inst} \quad (2)$$

Next (Step 4), the SDGRED method matches the aql value with the thresholds position values and subsequently updates maxthre_shold and doublemaxthre_shold positions in the router buffer to increase network performance (Fig. 7). Both maxthre_shold and doublemaxthre_shold values set according to the aql value.

In Fig. 7, the maxthre_shold and doublemaxthre_shold values increased and decreased around the minthre_shold by Equations (3) to (5) to prevent congestion at the router buffers. Thus, the aql value stabilizes around the minthre_shold and prevents the saturation of router buffers. As a result, fewer packets are dropped. Furthermore, the calculations can cause changes in aql value in a slow mode. Therefore, the Equations (3) to (5) are derived.

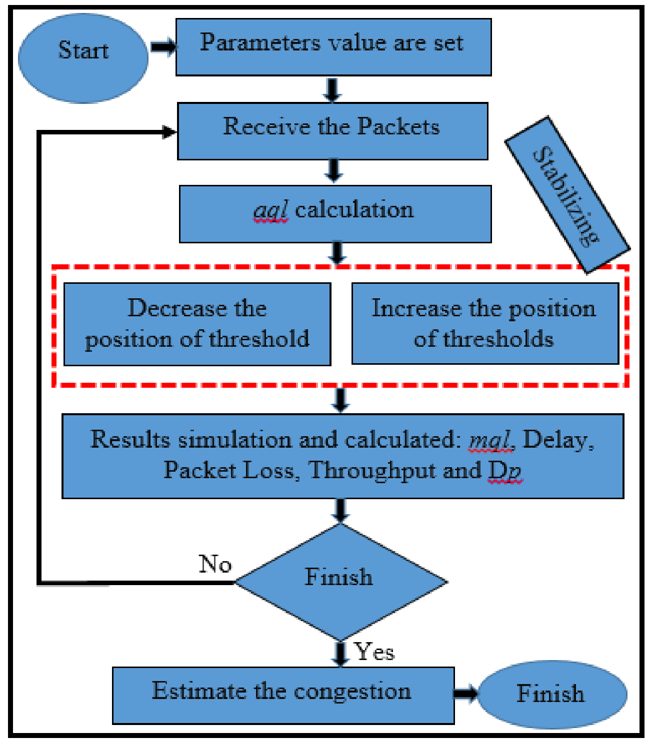

Fig. 4. SDGRED Stages.

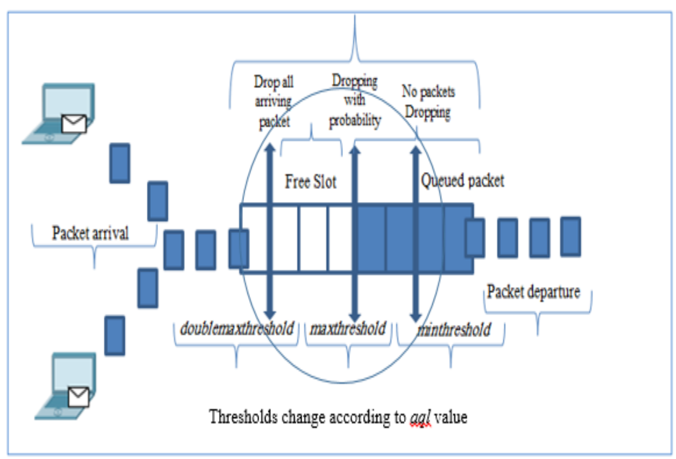

Fig. 5. The Proposed SDGRED Buffer.

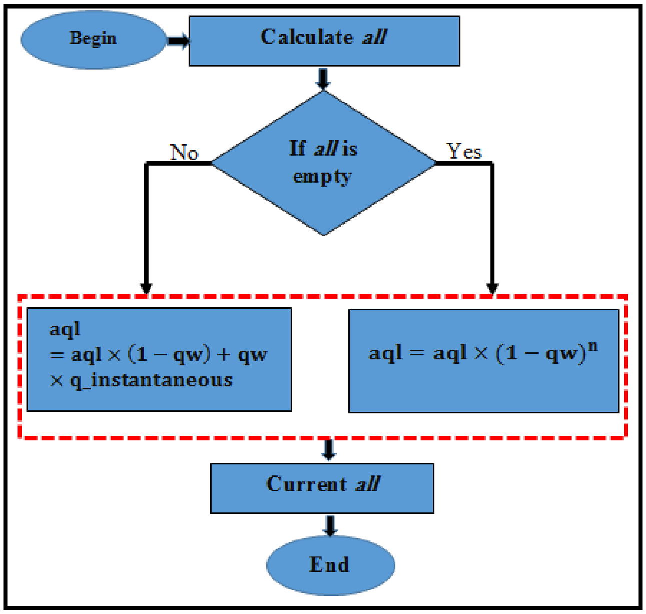

Fig. 6. Average Queue Length Status.

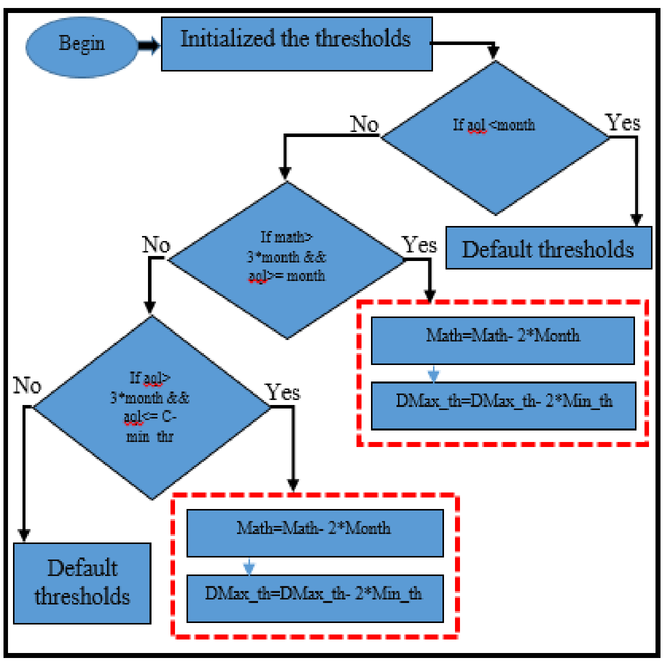

Fig. 7. Thresholds Stabilizing Stages.

As such, when the aql value is below min_threshold, no change occurs and the min_threshold, maxthre_shold and doublemaxthre_shold values will initialize [27]. Conversely, when the aql value is below the minthre_shold multiply two according Also maxthre_shold is reach greater than or equal to the position of minthre_shold multiply three, then the maxthre_shold and doublemaxthre_shold values change using Equations:

$$\text{maxthreshold} - 2 * \text{Minthre\_shold} \quad (3)$$

$$\text{Doublemaxthreshold} - 2 * \text{Minthre\_shold} \quad (4)$$

$$\text{maxthreshold} + \text{Minthre\_shold} \quad (5)$$

Thus, the aql value arises rapidly and stabilizes at the min_threshold. Also, if the aql is greater than the minthre_shold multiply three and less than or equal to the buffer capacity min_threshold, the maxthre_shold and double maxthre_shold values will set as shown by Equation. Thus, they become the same and prevent the double maxthre_shold value to go over the buffer capacity. Subsequently, the maxthre_shold and doublemaxthre_shold values increase to push the aql near the minthre_shold and decrease the probability the router buffer becomes full and over flows. In the last, in a case none of previous scenario happen, the maxthre_shold and doublemaxthre_shold is set to the same values as in the DGRED method [22].

(Step5) of the SDGRED method, the congestion is assessed and packet dropping is applied. aql plays a main role in congestion estimation according to dropping polices. In case the aql value is not reach the min_threshold, no event for congestion is presented at the SDGRED router buffer and no packet is dropped. In addition, Dp is fixed to zero and C is fixed to–1. Hence, no packet is reached to the boundary of threshold. If the aql value is between the minthre_shold and maxthre_shold values, the SDGRED router buffer operates as DGRED for dropping the arrival packets. Dropping packets based on increasing C by 1 and calculating Dp for arriving packets. If the aql value is between the doublemaxthre_shold and maxthreshold, the SDGRED router buffer starts drop the incoming packets based on DGRED method, which involves initializing the C value and set one and calculating Dp for current arriving packets. Lastly, if the aql value is reach the doublemaxthre_shold value, the proposed SDGRED router buffer drops every arriving packet with Dp equal 1 and sets C to zero. Subsequently, in case the SDGRED router buffer becomes empty, the value of idle time is set to current time directly.

IV SIMULATION

GRED, DGRED, and the proposed stabilize DGRED are simulated depend on a discrete time queue model which uses a time as a slot [28, 29]. Each slot time may contain packet arrival (alpha) and packet departure (beta). Simulation is implemented by applying the compared methods in a network environment involving a lone router buffer hop. Particularly, both packet arrival and departure are implemented in single hope on a first packet arrival first packet departure basis. GRED, DGRED, and SDGRED simulations are applied in Java with i5 processor device, 1.68 GHz and 8 GB RAM. In this simulation, the probability value for both alpha and beta for the router buffer in a specific slot time is called alpha and beta, respectively [23, 29].on the other hand, the Packet arrivals and packet departures are demonstrated using a Bernoulli process and a geometrical distribution, respectively [29].

IV. EVALUATION RESULTS

The performance results of the SDGRED method is compared with DGRED and GRED AQM methods. The performances are implemented in simulation environment 10 runs, each run getting different seeds value as an input to the random number producer. This scenario eliminates likely bias in the output performance results and yields confidence intervals value. The performance results are calculated after the system becomes stable to collect the results which means a steady state.

For the parameters are set in GRED, DGRED, and the proposed SDGRED are introduced using equal parameters at most. In order, to make congestion and non-congestion situations at the router buffer, the packet arrival was set to the following values[7].0.18,0.33, 0.48, 0.63, 0.78 and 0.93 respectively; each value of them goes to generate congestion or non-congestion station. The buffer size room was set 20 packets to guarantee the congestion at small buffer sizes. A total slot was set to 2000000 were used in the simulations. The minthre_shold is set 3, the maxthre_shold is set 9, doublemaxthre_shold is set 18, Dmax, is set 0.1 and qw is set 0.002, as recommended in DGRED[14]. Table I lists all the utilized parameters. The simulation performance results are stately using numerous performance metrics. Such as, Throughput, Delay, mql, packet loss, and dropping probability, which are discussed in the following subsection.

TABLE I. PARAMETERS SETTING

Parameter	DGRED	DGRED	SDGRED
alpha	0.18,033,048, 0.63,0.78,0.93	0.18,033,048, 0.63,0.78,0.93	0.18,033,048, 0.63,0.78,0.93
beta	1/2	1/2	1/2
Buffer size	20	20	20
Q_w	0.002	0.002	0.002
$D_{\_max}$	0.1	0.1	0.1
# of slots	2 millions	2 millions	2 millions
Mint_hreshold	3	3	3
Max_threshold	3*min	3*min	3*min
Double_maxthreshold	2*max	2*max	2*max
Target aql	parameter	------	dynamic

Mean Queue Length, Throughput, and Delay Results.

Respectively, Fig. 8, 9 and 10 explain the output performance results for GRED, DGRED, and the proposed SDGRED using different probabilities of packet arrivals as mention above. Specifically, Fig. 8 shows the mql and the probability of packet arrival.

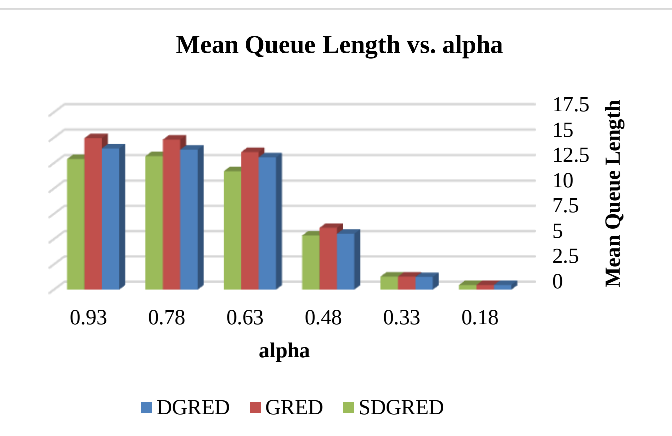

Fig. 8. Mean Queue Length.

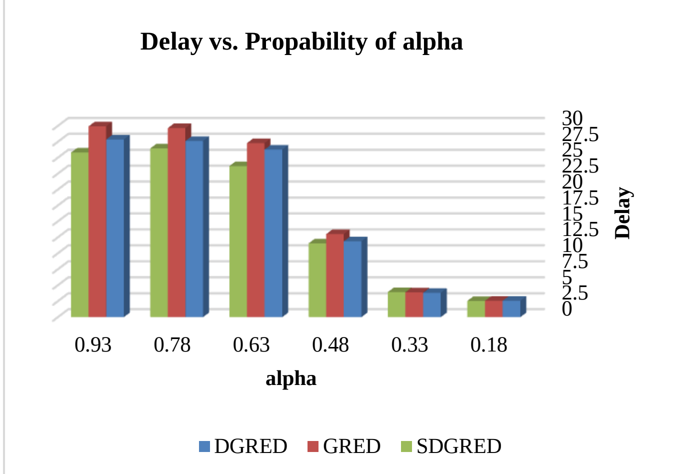

Fig. 9. Delay Performance Results.

The mean queue length for all methods and SDGRED method is the same up to a certain value of the probability of alpha 0.18, 0.33 and 0.48. Such a small probability value reasons bright congestion at most because the probability of packet departure was set 0.5 greater than that of alpha (a > B). So, all compared methods gain satisfied and stable mean queue length values. On the other hand, for a higher probability values, such as, 0.63, 0.78 and 0.93 congestion is more likely to occur at the router buffers. Thus, the mean queue length of the AQM methods arises exponentially. In such a case, the proposed SDGRED performs better than the DGRED and GRED methods because fewer packets are dropped and the router buffer space available for new packets arrival.

Fig. 9 illustrates a comparison of the delays in all algorithms. Although DGRED shows good performance in terms of the average delay, the proposed SDGRED performs better because of the fewer dropped packets in SDGRED.

Finally, Fig. 9 shows the throughput performance measure in all the packet arrival probabilities were set. The proposed SDGRED and compared methods gain the same throughput results either light congestion or heavy congestion, the packets arrival probability are set to 0.18, 0.33, and 0.48 which means lower probability or higher than that of packet departure, such as, 0.63, 0.78 and 0.93. Fig. 10 refers a probability of packet arrival arrive 0.18, 0.33 and 0.48 increases to arrive to the packet departure value. On the contrary, when the alpha arrives the value of beta, all the compared methods stabilize at the packet departure probability which equals 0.5 when congestion happens.

A. Packet Loss and Dp

The proposed SDGRED method is likewise compared with the DGRED and GRED methods in regards of PL performance measures and DP performance measures to display the amount of dropped packets in the buffer. The results of PL and DP are computed after the simulation becomes stable and steady. The method simulations are run 10 times with various random seeds and the mean is determined. The results of GRED, DGRED, and the proposed SDGRED algorithms in means of PL and DP are clarified in Fig. 11 and 12, in that order.

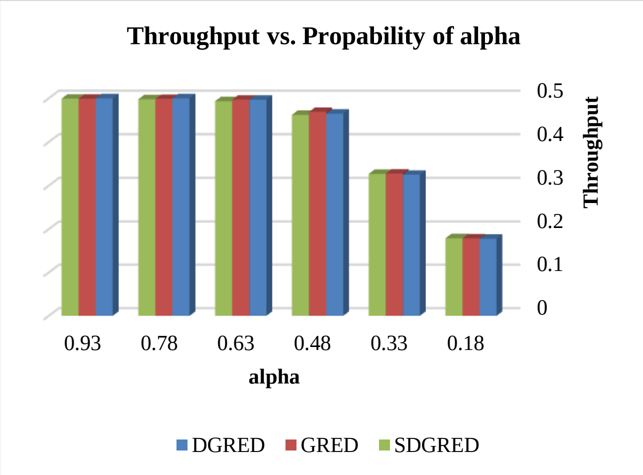

Fig. 10. The Throughput.

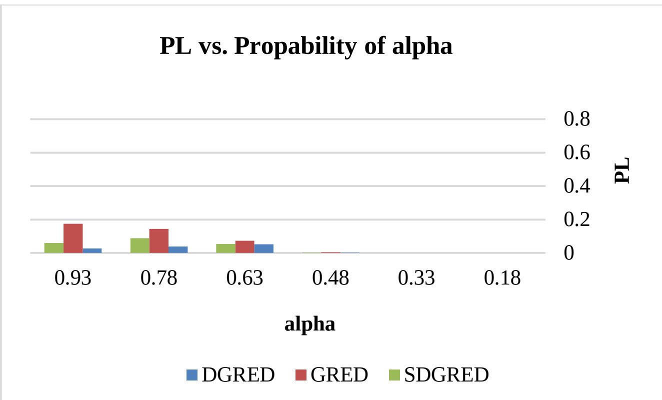

Fig. 11. Packet Loss Result.

In Fig. 11, GRED, DGRED, and the proposed SDGRED algorithm marginally produce the same PL performance result when the beta probability is greater than that of alpha. The DGRED introduces better PL performance at heavy congestion because the router buffer overflows earlier compared with those in the GRED and SDGRED methods.

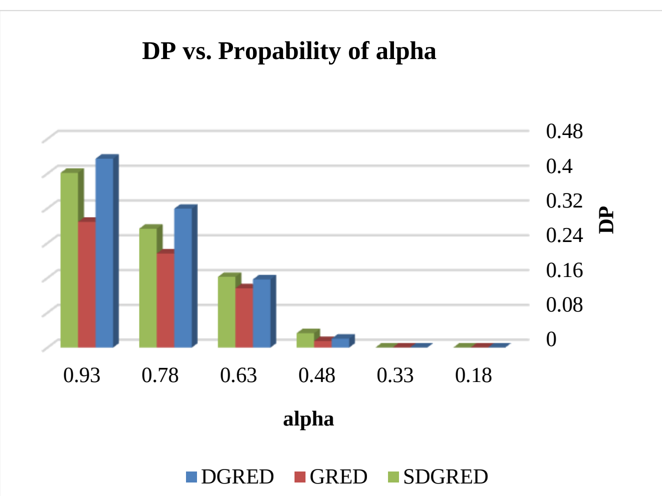

Fig. 12. Dropping Probability.

Fig. 12 shows that for the same reason, the proposed SDGRED algorithm evidently drops more packets when the beta probability is lower than that of alpha.

Therefore, the proposed method reveals the following improvements:

- GRED, DGRED, and the proposed SDGRED methods the same performance measure results when the alpha equals 0.18, 0.33, and 0.48.
- The SDGRED method offers a marginally better mql and Delay than the DGRED, and GRED methods when the alpha arrive to the 0.63, 0,78 and 0.93. In addition, when the alpha less than 0.5, the GRED, DGRED, and the proposed SDGRED methods gain similar T performance results.
- DGRED method slightly outperforms the SGRED and GRED methods for PL when heavy congestion. Moreover, at such values of packet arrival probability, SDGRED drops fewer packets (Dp) at their router buffers.

V. Conclusions

The current paper proposed an enhanced AQM method depend on the DGRED called the SDGRED. The proposed SDGRED aims to keep the aql between the minthre_shold position and doublemaxthre_shold position by changing the maxthre_shold and doublemaxthre_shold positions according to current aql value. This aql change helps stabilize the aql at minthre_shold position.in order to prevent the congestion. SDGRED employs maxthre_shold and doublemaxthre_shold positions in adaptive manner to keep the aql value arround the minthre_shold value, which may lead to fewer packet losses and queuing delay. The SDGRED technique is compared with the GRED and DGRED methods with the following performance measures such as, T, mql, D, PL, and Dp, to present which method offers better performance result in regards of packet arrival probability. The results show the SDGRD method is competitive to the compared methods.

References

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

[2] Adeeb alsaaidah, et al., Markov-Modulated Bernoulli-Based PErformence Analysis for Gentle Blue and Blue Algorithms under Bursty and Correlated traffic. Journal of Computer Sciences, 2016. 12(6): p. 289-299.

[3] Senthilkumaran, T. and V. Sankaranarayanan, Dynamic congestion detection and control routing in ad hoc networks. Journal of King Saud University - Computer and Information Sciences, 2013. 25(1): p. 25-34.

[4] ABDEL-JABER, H., et al., Random Early Dynamic Detection Approach for Congestion Control Baltic J. Modern Computing, 2014. 2(1): p. 16-31

[5] 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.

[6] BAKLIZI, M., et al., MARKOV-MODULATED BERNOULLI DYNAMIC GENTLERANDOM EARLY DETECTION. Journal of Theoretical and Applied Information Technology, 2018. 96(20).

[7] 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.

[8] 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).

[9] NSKI, A.C. and Ł. CHRÓST, ANALYSIS OF AQM QUEUES WITH QUEUE SIZE BASED PACKET DROPPING. Int. J. Appl. Math. Comput. Sci, 2011. 21(3): p. 567–577.

[10] 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.

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

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

[13] 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).

[14] Baklizi, M., et al., DYNAMIC STOCHASTIC EARLY DISCOVERY: A NEWCONGESTION CONTROL TECHNIQUE TO IMPROVENETWORKS PERFORMANCE. International Journal of InnovativeComputing, Information and Control, 2013. 9(4).

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

[16] Abdel-jaber, H., et al. Traffic Management for the Gentle Random Early Detection Using Discrete-Time Queueing. in Proceedings of the International Business Information Management Conference 2008. Marrakech, Morocco.

[17] Abdel-Jaber, H., et al., Performance evaluation for DRED discrete-time queueing network analytical model. Journal of Network and Computer Applications, 2008. 31(4): p. 750-770.

[18] Abdel-jaber, H., et al. Modelling BLUE Active Queue Management Using Discrete-Time Queue. in Proceedings of the 2007 International Conference of Information Security And Internet Engineering (ICISIE'07). 2007. London.

[19] Moarefianpour, A. and V.J. Majd, Input-to-State Stability in Congestion Control Problem of Computer Networks with Nonlinear Links. International Journal of Innovative Computing Information and Control 2009. 5(8): p. 2091-2105.

[20] Chen, X., T. Liu, and J. Zhao, A LOGIC-BASED SWITCHING CONTROL APPROACH TO ACTIVE QUEUE MANAGEMENT FOR TRANSMISSION CONTROL PROTOCOL. International Journal of Innovative Computing, Information and Control 2008. 4(7): p. 1811—1820.

[21] 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.

[22] Floyd, S. and V. Jacobson. Random early detection gateways for congestion avoidance. in Networking, IEEE/ACM Transactions on. 1993.

[23] 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.

[24] Bitorika, A., et al. A Comparative Study of Active Queue Management Schemes. in Proceddings of IEEE ICC 2004, Congestion Control Under Dynamic Weather Condition 103. 2004.

[25] Ryu.S, Active Queue Management (AQM) based Internet Congestion Control. October 1, 2002: University at Buffalo.

[26] Salim, J.H., U. Ahmed, and J. 2000, Performance evaluation of explicit congestion notification (ECN) in IP networks, N.W. Group, Editor. 2000, RFC 2884.

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

[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.