CONTROLLING DELAY AT THE ROUTER BUFFER USING MODIFIED RANDOM EARLY DETECTION

Ahmad Adel Abu-Shareha

Information Technology and Computing Department,
Arab Open University (AOU), Riyadh, Saudi Arabia

ABSTRACT

Active Queue Management (AQM) methods are used to manage the buffer of the network routers and avoid the problems caused by network congestion, especially packet loss. Among various AQM methods that have been proposed in the literature, Random Early Detection (RED)method has proved to stabilize the network performance under various traffic loads. However, as the primary concern of RED is to avoid loss when the router buffer overflowed, RED harms the delay at the router and increases the latency. Given that reducing delay is critical to some applications, such as online conferencing and broadcasting, RED needs to be adjusted to ease the delay problem. In this paper, RED is improved by monitoring the delay at the router buffer and implementing packet dropping to handle the issue of network delay and enhance the network performance. Accordingly, the modified method calculates and used a delay parameter with the RED to reduce the delay while maintaining the desirable RED's characteristics. The experimental results showed that the proposed Delay-Controller Random Early Detection (DcRED) improved network performance under various traffic loads. Compared to RED, DcRED results in less delay, while maintaining the loss and dropping rates.

KEYWORDS

Delay, Congestion, Random Early Detection, Active Queue Management

1. INTRODUCTION

Active Queue Management (AQM) methods are responsible for managing the router buffer and avoid congestion and its consequences, which are the increment of the packet loss and the decrement of the throughput (Figure 1). Buffer congestion occurswhen the amount of the transferred data exceeds the capacity of the buffer. Because congestion has terrible consequences on the network performance, various AQM methods were proposed in the literature, each of which has a specific target in defeating one of these consequences. Random Early Detection (RED) is the first AQM method, which was proposed to overcome the limitation of the passive approach that was used to manage the buffer, which is called a tail drop [1-3].

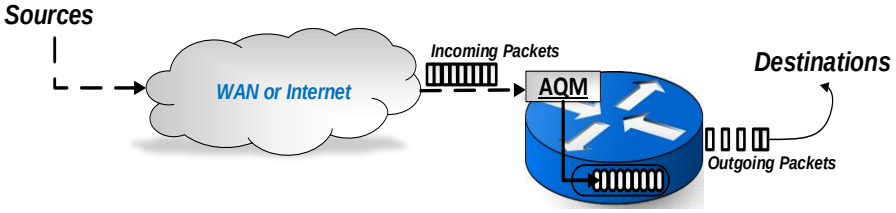


Figure 1. AQM at the Router Buffer

DOI: 10.5121/ijcnc.2019.11604

RED is formed of two elements, the dropping technique, and the utilized congestion indicator. The dropping technique prevents packet loss by dropping packets with early signs of congestion and prevents, to considerable extents,dropping packets unnecessarily. The utilized congestion indicator, which is the average queue length (aql), refers to the estimated length of the queue of the router buffer [4, 5]. Using *aql* has the advantages of recognizing false congestion, when heavy, yet temporary traffic, passing by the network. The ability to identify false congestion depends on the slow variation of *aql*values over time by computing the average length while givinga high weight for the previous queue lengths and less weight for the current queue length. Accordingly, when temporary heavy traffic is passed by the router, the value of *aql* does not change significantly and thus prevent unnecessary dropping [3, 6, 7].

The characteristics and advantages of the RED can be summarized in the following: 1) RED dropspackets early to avoid buffer overflowing and router congestion, by other means actively manages the buffer and avoid the congestion [8]. 2) RED drops packets randomly to avoid global synchronization. 3) RED drops packets stochastically with each packet arrival and keeps the buffer monitored [4, 5, 9-12]. These characteristics encourage its broad usability as it has been used for a long time in distributed devices all over the world. These characteristics allow stabilizing the network performance under various traffic loads. Another advantage of RED is the ability to modify the desired performance by altering the values of some parameters, which leads to multiple forms of RED called Weighted RED (WRED). WRED allows creating multiple virtual queues with different settings to provide various desired performance (Figure 2). Given these advantages, RED is approved by the Internet Engineering Task Force (IETF) in RFC 2309 [13-15].

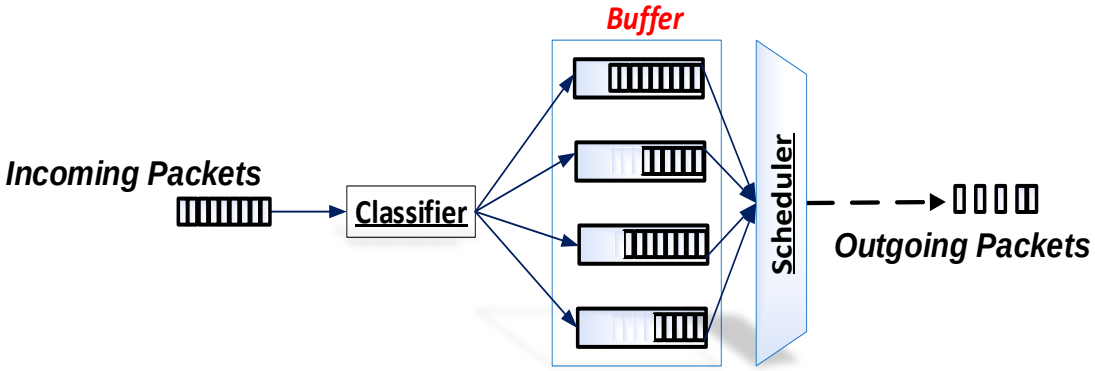


Figure 2. Weighted Random Early Detection

Although RED is proofed to provide sufficient throughput in different network status, some limitations have been found due to the busty nature of the real network traffic, which increased and decreased unsystematically and suddenly [8]. To enhance the performance of RED, variousAQM methods were proposed by modifying the utilized dropping technique and the congestion indicator. Extending the congestion indicator, together with the dropping technique have been implemented according to different directions, such as adapting different scenarios for packet dropping, calculating the dropping probability using different equations or using the fuzzy system, adding more constraints to the calculation of the dropping probability and using other indicators. These extensions provided a better performance under certain network statuses, such as light traffic or heavy traffic, while scarifying the performance under different statuses. Thus, these methods did not gain the same consideration as RED. Accordingly, RED remained the most commonly utilized technique for a long time [16].

Improving RED to enhance delay without sacrifices dropping is an urgent desire becauseavoiding delay is significant formany real-time applications, such as online conferencing and broadcasting.

In this paper, RED is extended to consider delay improvement, while preservingthe characteristics of the original method, maintainingstable performance and allowing a weighted version of the proposed method. The proposed method augmented the original method with delay indication that is calculated with each packet arrival and is used for calculating the dropping probability. The rest of the paper is organized as follows: RED mechanism and the related RED's extension are discussed in Section 2. Section 3 presents the details of the proposed method. Section 4 presents the simulation and parameter settings. The results are discussed in Section 5. Finally, the conclusion is given in Section 6.

2. RELATED WORK

Random Early Detection (RED) method was proposed by Floyd and Jacobson [1] to overcome the limitations of the traditional technique for queue management, which is the Drop-Tail (DT). DT is implemented merely by dropping all packets as the queue length reached a specific threshold or the buffer overflowed. On the other hand, RED performs router management by estimating the queue length of the router buffer with each packet arrival using *aql*, calculating the dropping probability of the packet with reference to the computed *aql* and stochastically making a decision to accommodate or drop the packet in order to avoid congestion, maximize the buffer utilization and avoid global synchronization. RED operationsareillustrated in Figure 3, which are described as follows: As a packet arrives at the router, RED calculates the value of the *aql*, compares the computed value of *aql* to some predetermined thresholds, and selects the proper decision among the dropping scenarios, which are packet accommodation, stochastically dropping and firmdropping. If the stochastic dropping is the choice, then, the dropping probability *Dp* is calculated, and the packet is dropped randomly based on the calculated probability. The scenarios involved in the RED technique links the estimated status of the buffer with the decision to be made. Accordingly, no dropping decision is made for a small queue size when there is no indication of congestion. Firmdropping decision is made for a full or almost full queue, which indicates that the buffer is congested. Finally, stochastic dropping is made when there is a sign of congestion, and the dropping probability is increased as the queue length is increased.

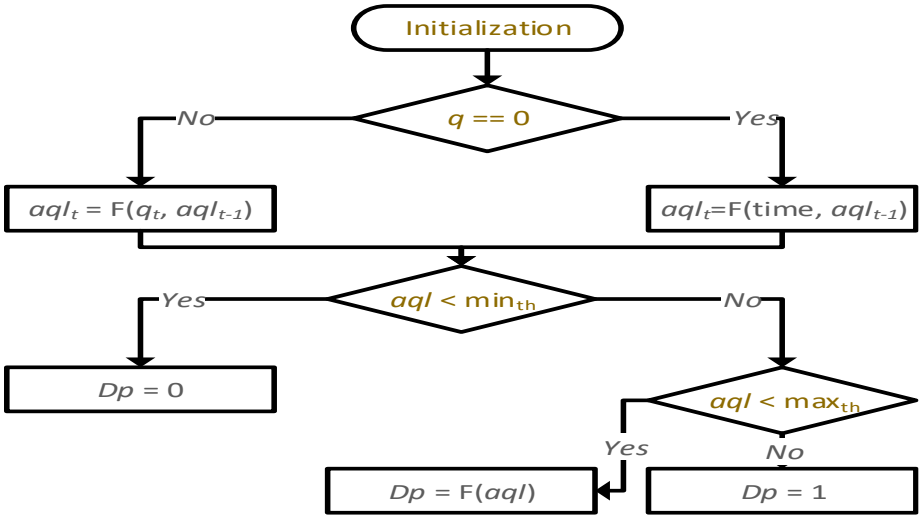


Figure 3. Flowchart of the RED Method

The following AQM methods extend RED to achieve different objectives. Particularly, to reduce packet loss, Effective Random Early Detection (ERED) [6], Adaptive Queue Random Early Detection (AQMRD) [17], Research on Improving Random Early Detection (S-RED) [4],

Enhanced RED (EnRED) and Time-window Augmented RED (Windowed-RED) [8]were proposed. Similarly, various methods were proposed to reduce packet droppings, such as Cloud-based Random Early Detection (CRED) [5], Adaptive Threshold RED [11], and Adaptive AQMRD [18]. For a fair resource allocation objective, which implemented by dropping packets based on the shared bandwidth consumed by each flow, Stabilized RED (SRED) [19] and Probability-based RED (P-RED) [20] methods were proposed. Various methods were proposed for adaptive parameterestimation, such as: AQMRD [17], CRED [5], Adaptive Threshold RED [11], Adaptive- AQMRD [18], Adaptive Tuning of Drop-Tail (ADT) [21], Adaptive BLUE [22], Self-Tuning Price-based Congestion Control (SPC) [7], New Adaptive RED (NARED) [23] and Queue-Threshold RED (LTRED) [12]. To ease the parameter sensitivity, various AQM methods used fuzzy logic with queue management. Examples of fuzzy-based AQM are FuzzyRED[13], DeepBlue[24], Fuzzy logic for GRED (GREDFL) [2], and FLRED [3]. To reduce the delay, various AQM methods were proposed, these are Gentle RED (GRED) [25], Adaptive RED (ARED) [26], Dynamic RED (DRED) [27], PI [28], AVQ [29] and DQRED [30]. Table 1 summarizes the objectives of the existing AQM methods.

Table 1. AQM Methods' Objective

Goal	Methods
To reduce packet loss	ERED [6], AQMRD [17], S-RED[4]EnRED) and Windowed-RED [8].
To reduce packet dropping	CRED [5], Adaptive Threshold RED [11], and Adaptive AQMRD [18].
To implement fair resource allocation	SRED [19] and P-RED [20].
To adaptively estimate the parameter settings	AQMRD [17], CRED [5], Adaptive Threshold RED [11], Adaptive-AQMRD [18], Adaptive Tuning of Drop-Tail (ADT) [21], Adaptive BLUE [22], SPC [7], NARED [23] and LTRED [12].
To ease the parameter sensitivity	FuzzyRED[13], DeepBlue[24], Fuzzy logic for GRED (GREDFL) [2], and FLRED [3].
To reduce the delay	GRED [25], ARED [26], DRED [27], PI [28], AVQ [29] and DQRED [30]

GRED [25] add more scenarios to those implemented in RED and implements packet dropping as the length of the queue increases. ARED [26] and DRED [27] increase the dropping rate concerning the new parameter that determined the target aql value. Generally, GRED [25], ARED [26], and DRED [27] solve the problem by implementing more aggressive dropping techniques compare to RED. Accordingly, as more packets are dropped, the queue is maintained in manageable length, and the delay is reduced significantly. However, network performance was degraded. Besides, the ability to create multiple weighted queues to distinguish between real-time and non-real time was lost.

Among the existing AQM methods, some methods are dedicated to enhancing delay adaptively to preserve the network performance. PI [28] used delay estimation in the calculation of Dp. Two equations are used to calculate two probabilities, one concerning *aql* and the other based on the estimated delay. The two probabilities are then used to calculate the final dropping probability, which balances between loss and delay. AVQ [29] used a novel technique that calculates *Dp* based on the estimated load. A virtual queue with a specific capacity and target utilization is created and synchronized with the real queue. The dropping probability is calculated based on the utilization of the virtual queue. Accordingly, to reduce the delay, the desired utilization is set to a minimum value. DQRED [30] used multiple virtual queues to accommodate different traffic classes and implement aggressive packet dropping to reduce the delay for real-time traffic. Overall, similar to other objectives, reducing delay was addressed in the literature by various

AQM methods. However, the performance of the network was degraded, and some of the desired characteristics were lost. Table 2 summarizes the related work that aimed at reducing the delay.

Table 2. AQM Methods with Delay Reduction Objective

Method	Indicator	Summary
GRED **[25]**	*aql*	Added a scenario and a new dropping probability equation to RED.
ARED **[26]**	*aql*	Maintained the queue length at a target value *Taql*.
DRED **[27]**	*aql*	Maintained the queue length at a target value *Taql*.
PI **[28]**	Load rate &*q*	Calculated *Dp* based on load rate and queue length.
AVQ **[29]**	Load rate	Calculated *Dp* based on load rate.
DQRED **[30]**	*aql*	Used multiple virtual queues, each for a specific traffic class.

3. PROPOSED WORK

To enhance the delay while preserving the original characteristics of RED, a new AQM method is implemented based on the original technique with an augmented delay parameter. Delay is calculated with each packet arrival, and the resulted value is used as a reference for packet dropping. Because the delay conflicted with the other desirable features, such as throughput and dropping rate, a balancing between throughput and delay is maintained. In order to balance between delay and other performance aspects, the utilized delay parameter is formed using the average queue length (aql) that is utilized initially with RED to decrease loss and increase the throughput.

3.1. Delay Parameter Calculation

An estimated delay parameter is used with the proposed Delay-Controller Random Early Detection (DcRED) method to determine the dropping rate. The calculated delay adheres to the common delay sources at the router buffer, which can be referred back to the arrival rate, the departure rate, and the queue size. The arrival and departure rates together form the buffer's load rate. Accordingly, in DcRED, delay at the router buffer is estimated based on two variables, these are 1) Load rate at the router buffer(l), which is proportional to the delay at the router. 2) Instance queue size (q), which is also proportional to delay. However, although the load is proportional to delay, having a high load with a smallqueue size does not significantly increase the delay, while a high load with large queue size are indications forboth delay and packet loss. Accordingly, the combination of these parameters together affects the delay at the router. Thus, the delay is calculated as the multiplication of these variables, as given in Equation 1.

$$D = l * q \tag{1}$$

As mentioned, the load rate is influenced by the estimated arrival and departure rates. Accordingly, the load at the buffer is calculated as the differences between the arrival rate and departure rate. In order to avoid false instance for arrival and departure rate, both of these variables are calculated as the average of multiple values at different time. Accordingly, the proposed method uses the average arrival rate (A) and the average departure rate (Dt) to calculate the load as given in Equation 2.

$$l = Minimum\ (A - Dt, 0) \tag{2}$$

where, *A* is is the value of arrival rate and, *Dt* is the departure rate.

The value of *A* is calculated as a weighted average of the current and previous arrival values, as given in Equation 3. The value of the weight *w* is set to a value of less than 0.5, to give more weight for the previous arrival rate.

$$A_t = w(A_{instance}) + 1 - w(A_{t-1}) \quad (3)$$

where, *w* is the weight, A_t is the value of instance arrival at the current time, $A_{t-1}$is the calculated average arrival rate at the previous time.

The current arrival value is calculated as the inverse of the duration between the current and previous packet arrival, as given in Equation 4.

$$A_{instance} = 1/(Time_i - Time_{i-1}) \quad (4)$$

where$Time_i$ is the time value at which the latest arrival occurs, $Time_{i-1}$is the time value at which the previous arrival occurs. Accordingly, $Time_i$- $Time_{i-1}$ represents the period between the last two subsequent arrivals.

For the departure rate, because AQM methods monitor the arrival process only, the departure process cannot be measured directly. Accordingly, the average departure rate (Dt) is calculated based on the differences between the arrival and the queue rate, as given in Equation 5.

$$Dt_t = A_{instance} - (\frac{aql}{BufferSize}) \quad (5)$$

where*aql* is the average queue length, BufferSize is the capacity of the utilized buffer. Solving Equation 1 to Equation 5 leads to derive a new calculation for the delay, as given in Equation 6.

$$Delay = \frac{aql*q}{BufferSize} \quad (6)$$

Accordingly, the delay is calculated as the ratio between the average and current queue and the buffer size. As either the average or instance queue size increases, the delay increase accordingly. However, to give weight for the desired delay, a weighted version of Equation 6 is implemented by giving weights for Buffer size parameters, as given in Equation 7. As less delay is desired, less weight should be given to the value of *w* parameter.

$$Delay = \frac{aql*q}{w_b* BufferSize} \quad (7)$$

3.2. DcRED Mechanism

The proposed DcRED is implemented by extending the original RED using the calculated delay parameter. To keep the modification on the original RED as minimum as possible with the purpose to preserve the characteristics of the RED, only *Dp* calculation is modified. Algorithm 1 represents the proposed DcRED method.

Algorithm 1: DcRED

1 PARAMETER SETTING: w_q, w_b, Th_{min}, Th_{max}, D_{max}
2 VARIABLE INITIALIZATION:*aql*:= 0, *count*:= -1
3 FOR-EACH *A*
4 IF (*q* equal to 0) →*aql*:=$(1\text{-}w)^{f(cTime\text{-}iTime)} * aql$
5 IF (*q* not equal to 0) →*aql*:= (1-w)* *aql* + w_q **q*
6 *d*=(*aql* * *q*)/w_b * BufferSize
7 IF ($min_{th} \leq aql < max_{th}$)
8 *counter*++, *Dp'*= D_{max}* (*d* -Th_{min})/(Th_{max} - Th_{min})
9 *Dp* = *Dp*'/ (1- *counter***Dp'*)
10 IF (Drop(*Dp*) is TRUE) → Drop-A, *count* := 0
11 ELSE IF (*aql*>max_{th}) → Drop-A, *counter*:= 0
12 ELSE →*counter*:= -1
13 IF (*q* equal to 0 &&*Idel* equal to False)
14 iTime=cTime, Idel = TRUE
15 ELSE Idel = False

The parameters and variables are initialized in line 1 and line 2. The value of *aql* is calculated as given in line 4 and line 5. Then, the value of the delay parameter is calculated in line 6. The *Dp* calculation is implemented based on three scenarios, as given in line 7 to line 12. As similar to RED, the *aql*, which is considered as the queue monitoring parameter is compared to two thresholds, as given in line 7 and line 11:when *aql* value is within the two thresholds, *Dp* is calculated as given in line 8 and line 9, with reference to the values of the delay, a counter parameter that avoids global synchronization, the maximum probability parameter, and the two thresholds. Accordingly, the arriving packet A is dropped based on the calculated value in line 10. In line 11, if *aql* is greater than the maximum threshold, then the packet is dropped firmly. Finally, in line 12, if *aql* is less than the minimum threshold, then the packet is accommodated. Line 13 and line 15 trace the empty buffer and save the time value that is used to calculate the *aql*.

4. SIMULATION AND PARAMETER SETTINGS

The simulation process consists of simulating the network, simulating the router buffer, the AQM methods, and the evaluation process. Network simulation is implemented as a discrete-time queue model, which consists of slots, in each slot; packet arrival and packet departure occur stochastically based on the predetermined arrival and departure rates, α and β, respectively[31]. The values of α are set to {0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 0.95}. These values are tested to create different traffic load ranging from light, moderate, and heavy traffic. While the values of β are set to {0.3 and 0.5}. The number of slots that is used for the simulation is 2,000,000, the first 800,000 slots are used as a warm-up period, while the rest is used for collecting the results. The buffer is simulated as a memory space of a predetermined number of packets that are denoted by a parameter, and it is modelled as FIFO (First-In-First-Out). The buffer size is set to the value of 20. The proposed and compared AQM methods are simulated as decision making for every arriving packet. A set of parameters is initialized, based on the values recommended by the original methods [2, 4, 10]. These parameters are: queue weight is set to the value of 0.002, the maximum probability is set to the value of 0.1, minimum and maximum thresholds are given the values 3 and 9, respectively. Finally, the buffer weight is assigned the value of 0.1, which empirically proofed to be the best.

5. RESULTS

The performance of the proposed DcRED is evaluated and compared to the original RED method, ERED method and BLUE method, which is known to reduce delay compared to the state of the art methods [24]. As these methods are simulated, the results are captured using a set of measurements; these are a delay, dropping, and packet loss. Delay is measured as the average number of slots in which each packet is accommodated in the buffer. Dropping is simply reported as the number of dropped packets and loss is the number of lost packets. The total number of packets arrived at the buffer depends on α and the number of slots.The simulation is run several times based on various α and β values. The first run aimed at creating a heavy load at the router, with the values of 0.9 and 0.3 for α and β, respectively. Accordingly, this runs expected to create heavy congestion as the arrival rate is triple the departure rate. The second run aimed at creating a heavy load, but lighter compared to the first run, with the values of 0.9 and 0.5 for α and β, respectively. Similar to the first run, this run is expected to create congestion as the arrival rates are almost double the departure rate. The third run aimed at creating a moderate load using the values of 0.5 for both α and β. This run is expected to create limited light congestion at some points while the network is running. The fourth run aimed at creating a light load, with the values of 0.3 and 0.5, for α and β, respectively. This run should not create any congestion at the router, as the departure rate is greater than the arrival rate. Finally, the results of eight runs with values of {0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 0.95} and 0.5 for α and β, are discussed.

Figure 4 illustrates the delay comparison between the proposed and compared methods under various traffic statuses.

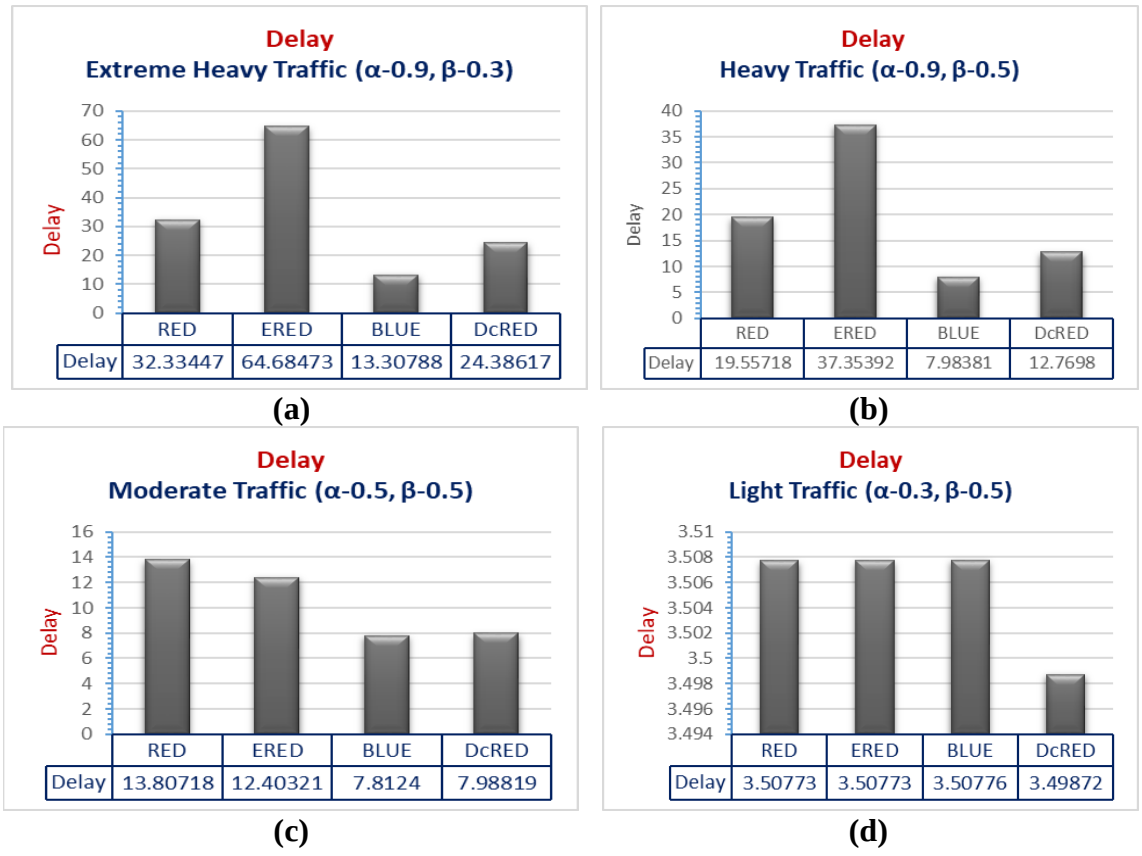


Figure 4. Delay Comparison under Four Traffic Statuses

As noted, compared to RED and ERED, the proposed method, DcRED reduces the delay significantly in all traffic classes. However, it is noted that BLUE over-performs DcRED in all classes except with light traffic. In Figure 4(a) with extremely heavy traffic, ERED doubles the amount of delay produced by RED, while BLUE cut delay by more than half and DcRED reduce the delay by one third. The variations in delay between these methods are decreased as the traffic load decrease, as illustrated in Figure 4(b) and Figure 4(c). In light traffic, all the compared methods result in the same delay and DcRED slightly reduce delay compared to the rest of the methods.

Figure 5 illustrates the loss comparison between the proposed and compared methods under various traffic statuses. Again, ERED, compared to RED, performs badly in terms of packet loss, while BLUE reducesloss significantly compared to RED. The proposed DcRED method over-performs all other methods and does not result in any loss with various traffic classes. In Figure 5(a) with extremely heavy traffic, ERED increases loss significantly compared to RED, while BLUE cuts loss significantly and DcRED lose no packets during this experiment. While loss should be decreased as the traffic load decreased, RED increases loss in Figure 5(b), while the loss in ERED decreased significantly, BLUE increases loss slightly, and DcREDmaintains zero loss. In moderate traffic, as illustrated in Figure 5(c) ERED performance-enhanced significantly, and the loss resulted from ERED over-performed both RED and BLUE methods, while the loss of DcRED remains zero. Finally, using all the compared methods, loss in light traffic is equal to zero.

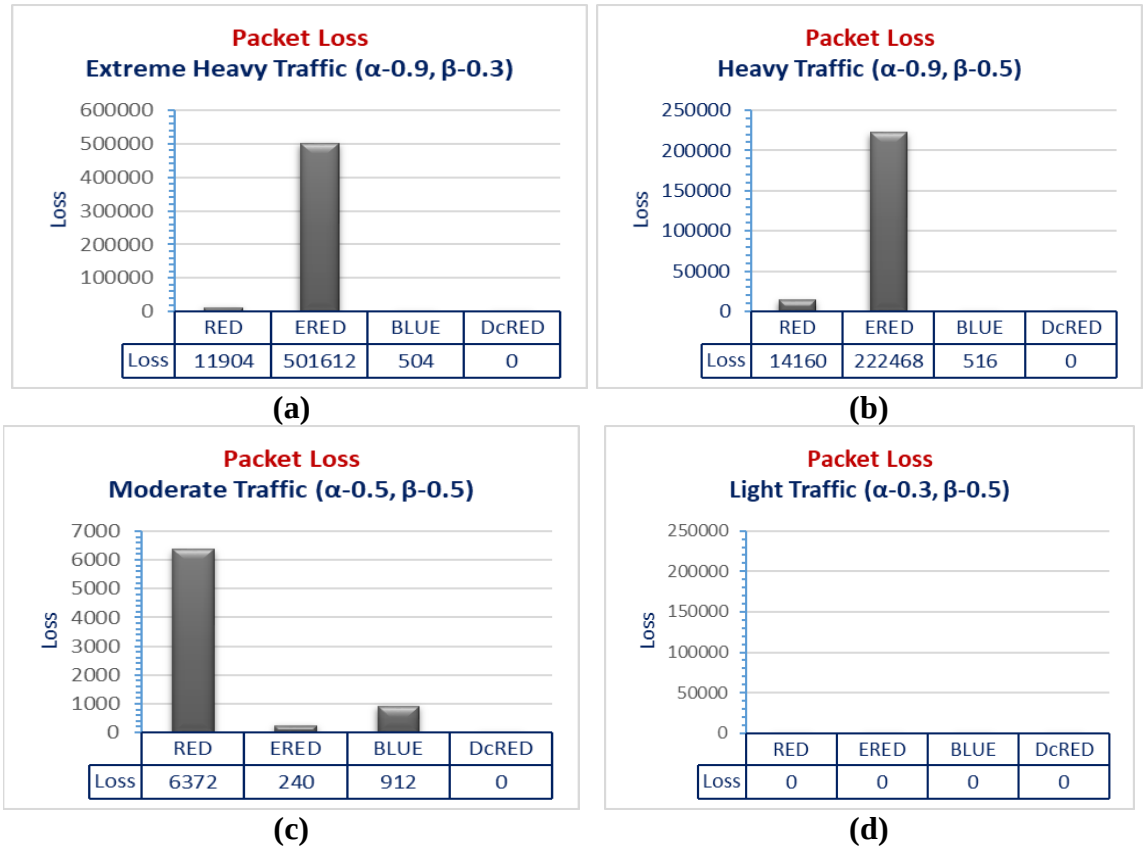


Figure 5. Packet Loss Comparison under Four Traffic Statuses

Figure 6 illustrates a comparison between the proposed and compared methods based on packet dropping under various traffic classes. The bad performance of ERED in terms of delay and loss

can be justified with the low dropping rate of ERED compared to the rest of the methods, as illustrated in the Figure. BLUE gained its superior results in terms of delay and loss by scarifying high dropping rate compared to RED, as illustrated in Figure 6. Unlike BLUE, the proposed DcRED method gained the advantages of reducing delay and loss of RED without scarifying much of the dropping rate.

In Figure 6(a) with extremely heavy traffic, ERED decreases dropping, BLUE increases dropping DcRED maintain the same dropping as RED. Accordingly, DcRED over-performs all other methods by maintaining the dropping rate while reducing delay and loss significantly in extremely heavy traffic. BLUE has the worst dropping rate and the best delay and loss in extremely heavy traffic. ERED have the worst delay and loss and the best dropping rate in extremely heavy traffic. In Figure 6(b) and Figure 6(c) with heavy and moderate traffic classes, similar results obtained from the compared methods. As such, ERED drops the least number of packets, RED and DcRED drop more packets compared to RED, while BLUE drops significantly more packets compared to all other methods. Accordingly, DcRED over-performs all other methods by maintaining the dropping rate while reducing delay and loss significantly in heavy and moderate traffic classes. BLUE has the worst dropping rate and the best delay and loss in the moderate traffic class. ERED have the worst delay and loss and the best dropping rate in the moderate traffic class. In Figure 6(c), all methods almost drop zero packets, except for DcRED, which dropped an insignificant number of packets.

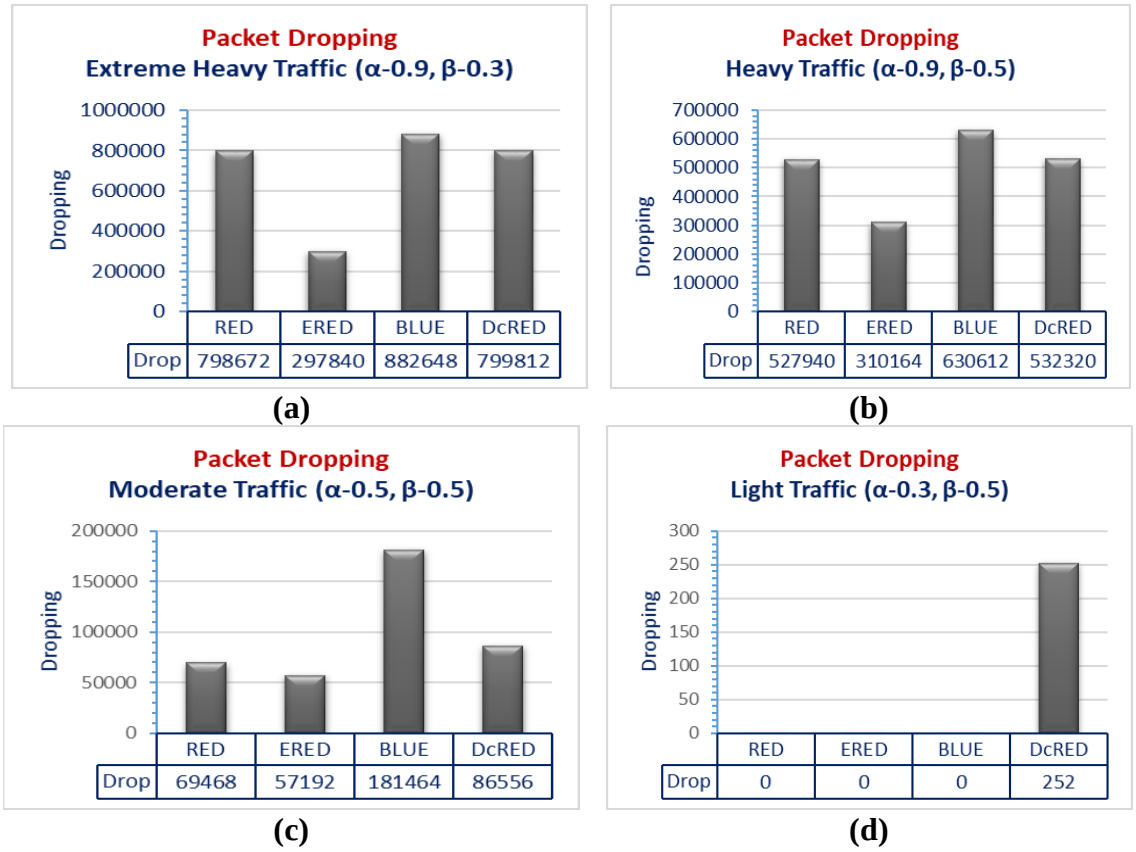
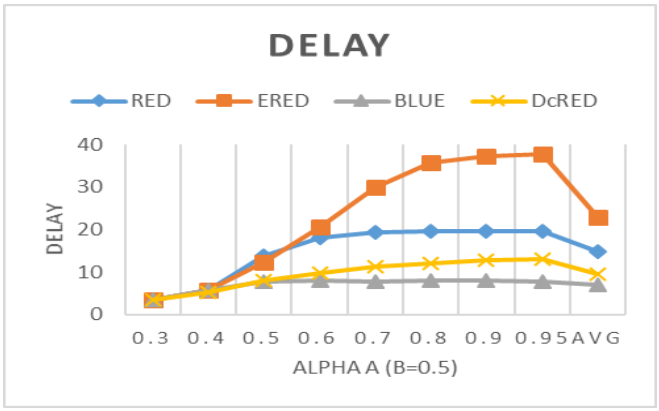


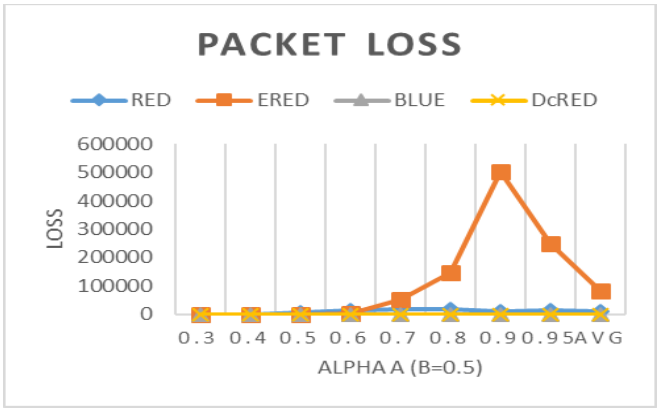
Figure 6. Dropping Comparison under Four Traffic Statuses

Figure 7 illustrates delay, loss, and drop comparison between the proposed and compared methods under various traffic statuses. The results confirmed the findings of the previous experiments. Overall, DcRED over-performs all other methods by maintaining the dropping rate

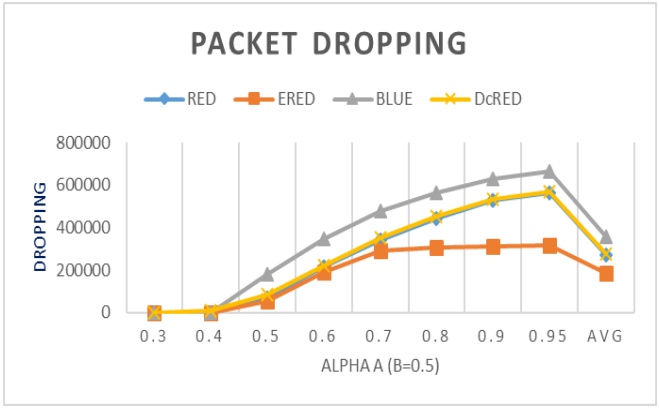
while reducing delay and loss significantly in extremely heavy, heavy, and moderate traffic classes. BLUE has the worst dropping rate and the best delay and loss in these traffic classes. ERED have the worst delay and loss and the best dropping rate in these traffic classes. In light traffic, all the compared methods perform equally.



(a)



(b)



(c)

Figure 7. Performance Measures Comparison under Four Traffic Statuses

CONCLUSION

In this paper, an extended RED method that concerns about the delay at the router buffer are proposed. The proposed method, DcRED,enhances the delay while preserving the original RED's characteristics. Delay is represented as a parameter that is used to calculate the dropping probability of the arrival packets and accordingly, reduce the delay at the router buffer. Delay at the router buffer is calculated regarding three parameters; these are the arrival rate, the departure rate, and the queue size. The performance of the proposed method in terms of delay, loss and drop under various traffic classes showed that DcRED over-performs all other methods by maintaining the dropping rate of RED while reducing delay and loss significantly in extremely heavy, heavy and moderate traffic classes. In light traffic, all the compared methods perform equally. Future work will focus on using other delay parameters that enhance network performance and reduce the overall delay, loss,and dropping rate.

REFERENCES

[1] S. Floyd, And V. Jacobson, Random Early Detection Gateways For Congestion Avoidance, Ieee/Acm Transaction On Networking, Vol. 1, No. 4, Pp. 397-413, Augest, 1993.

[2] M. Baklizi, H. Abdel-Jaber, A. A. Abu-Shareha, M. Abualhaj, And 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, No. 1, Pp. 9-19, March, 2014.

[3] M. M. Abualhaj, A.A. Abu-Shareha, And M.M. Al-Tahrawi, Flred: An Efficient Fuzzy Logic Based Network Congestion Control Method. Neural Computing And Applications, Vol. 30, No. 3, Pp. 925-935, Augest, 2018.

[4] Z. Yu-Hong, Z. Xue-Feng, And 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, No. 2, Pp. 375-384, 2016.

[5] Y. Zhao, Z. Ma, X. Zheng, And X. Tu, An Improved Algorithm Of Nonlinear Red Based On Membership Cloud Theory. Chinese Journal Of Electronics, Vol. 26, No. 3, Pp. 537-543, May, 2017.

[6] B. Abbasov, And S. Korukoglu, Effective Red: An Algorithm To Improve Red's Performance By Reducing Packet Loss Rate. Journal Of Network And Computer Applications, Vol. 32, No. 3, Pp. 703-709, May, 2009.

[7] H. Wang, Z. Tian, And Q. Zhang,Self-Tuning Price-Based Congestion Control Supporting Tcp Networks. 19thinternational Conference On Computer Communications And Networks (Icccn), Zurich, Switzerland, 2010, Pp. 1-6.

[8] A.A. Abu-Shareha, Enhanced Random Early Detection Using Responsive Congestion Indicators, International Journal Of Advanced Computer Science And Applications (Ijacsa), Vol. 10, No. 3, Pp. 358-367, March, 2019.

[9] A. Fakharian, And A. Abbasi, Design Of Congestion Controller For Tcp Networks Based On Lmi Formulation, Journal Of Optimization In Industrial Engineering, Vol. 8, No. 17, Pp. 51-56, 2015.

[10] L. Tsavlidis, P. Efraimidis, And R.-A. Koutsiamanis, Prince: An Effective Router Mechanism For Networks With Selfish Flows. Journal Of Internet Engineering, Vol. 6, No. 1, Pp. 355-362, 2016.

[11] Z. M. Patel, Queue Occupancy Estimation Technique For Adaptive Threshold Based Red. Ieee International Conference On Circuits And Systems (Iccs), Thiruvananthapuram, Kerala, India. 2017, Pp. 437-440.

[12] K. Chhabra, M. Kshirsagar, And A. Zadgaonkar, An Improved Red Algorithm With Input Sensitivity, Cyber Security (Csi), Singapore, 2015, Pp. 35-45.

[13] H. Abdel-Jaber, M. Mahafzah, F. Thabtah, And M. Woodward, Fuzzy Logic Controller Of Random Early Detection Based On Average Queue Length And Packet Loss Rate,International Symposium On Performance Evaluation Of Computer And Telecommunication Systems (Spects), Edinburgh, Uk, 2008, Pp. 428-432.

[14] H.Abdel-Jaber, J. Ababneh, F. Thabtah, A. M. Daoud, And M. Baklizi, Performance Analysis Of The Proposed Adaptive Gentle Random Early Detection Method Under Noncongestion And Congestion Situations, International Conference On Digital Enterprise And Information Systems,London, Uk,2011, P. 592-603.

[15] Qiao, Y. And L. Qiongyu, A New Active Queue Management Algorithm Based On Self-Adaptive Fuzzy Neural-Network Pid Controller, International Conference On Internet Technology And Applications (Itap), London, Uk, 2011, P. 1-4.
[16] Y. Hadjadj-Aoul, Towards Aqm Cooperation For Congestion Avoidance In Diffserv/Mpls Networks. Recent Patents On Computer Science, Vol. 2, No. 1, Pp. 1-13, January, 2009.
[17] S. Patel, And S. Bhatnagar, Adaptive Mean Queue Size And Its Rate Of Change: Queue Management With Random Dropping, Telecommunication Systems, Vol. 65, No. 2, Pp. 281-295, September, 2017.
[18] S. Bhatnagar, And S. Patel, A Stochastic Approximation Approach To Active Queue Management, Telecommunication Systems, Vol., 68, No. 1, Pp. 89-104, May, 2018.
[19] T. J. Ott, T.V. Lakshman, And L. Wong, Sred: Stabilized Red. Annual Joint Conference Of The Ieee Computer And Communications Societies (Infocom '99), New York, Usa, 1999, Pp. 1346-1355.
[20] N. Sharma, S. S. Rajput, A. K. Dwivedi, And M. Shrimali, P-Red: Probability Based Random Early Detection Algorithm For Queue Management In Manet, Advances In Computer And Computational Sciences, Pp. 637-643, September, 2017.
[21] R. Stanojevic, R.N. Shorten, And C.M. Kellett, Adaptive Tuning Of Drop-Tail Buffers For Reducing Queueing Delays. Ieee Communications Letters, Vol. 10, No. 7, Pp. 570-572, July, 2006.
[22] G. Da-Gang, A New Adaptive Blue Algorithm, International Conference Onelectrical And Control Engineering (Icece), Wuhan, China, 2010, Pp. 1-9.
[23] J. Zhang, W. Xu, And L. Wang, An Improved Adaptive Active Queue Management Algorithm Based On Nonlinear Smoothing. Procedia Engineering, Vol. 15, No. 1, Pp.2369-2373, 2011.
[24] S. S. Masoumzadeh, G. Taghizadeh, K. Meshgi, And S. Shiry, Deep Blue: A Fuzzy Q-Learning Enhanced Active Queue Management Scheme, International Conference On Adaptive And Intelligent Systems (Icais'09), Klagenfurt, Austria, 2009, Pp. 43-48.
[25] S. Floyd, Recommendations On Using The Gentle Variant Of Red, Icsi Networking Group Technical Report,March, 2000.
[26] S. Floyd, R. Gummadi, And S. Shenker, Adaptive Red: An Algorithm For Increasing The Robustness Of Red's Active Queue Management. At&TCenter For Internet Research At Icsi Technical Report, 2001.
[27] J. Aweya, M. Ouellette, And D.Y. Montuno, A Control Theoretic Approach To Active Queue Management. Computer Network, Vol. 36, No. 2-3, Pp. 203-235, July, 2001.
[28] C. V. Hollot, V. Misra, D. Towsley, And W. Gong, On Designing Improved Controllers For Aqm Routers Supporting Tcp Flows, Twentieth Annual Joint Conference Of The Ieee Computer And Communications Societies, Anchorage, Usa, 2001, Pp. 1726-1734.
[29] S. Kunniyur, R. And Srikant, End-To-End Congestion Control Schemes: Utility Functions, Random Losses And Ecn Marks, Ieee/Acm Transactions On Networking, Vol. 11, No. 5, Pp. 689-702, October, 2003.
[30] H. Mohammed, G. Attiya, And S. El-Dolil, Active Queue Management For Congestion Control: Performance Evaluation, New Approach, And Comparative Study, International Journal Of Computing And Network Technology, Vol. 5, No. 2, Pp.37-49, May, 2017.
[31] M. Khatari, And G. Samara, Congestion Control Approach Based On Effective Random Early Detection And Fuzzy Logic, Magnt Research Report, Vol. 3, No. 8, Pp. 180-193, 2017.

AUTHORS

Ahmad Adel Abu-Shareha is an assistant professor at the ITC Department in Arab Open University (Riyadh/Saudi-Arabia). He received his first degree in Computer Science from Al Al-Bayt University (AABU), Jordan, 2004, Master degree from UniversitiSains Malaysia (USM) – Malaysia, 2006, and Ph.D degree from UniversitiSains Malaysia (USM) – Malaysia, 2012. His research focuses on Data mining, artificial intelligent and Multimedia Security. He investigated many Supervised and Unsupervised machine learning algorithms and employed artificial intelligent in variety of fields, such as network, medical information process, knowledge construction and extraction. His research is guided by the today's computer-related problems, which deemed for untraditional solutions and outcomes to what so called convergence. The convergence is recently gain researcher interest to bridge the gap between the different computer disciplines and to improve the future technologies.