Linear Random Early Detection for Congestion Control at the Router Buffer

Ahmad Adel Abu-Shareha*[1], Basil Al-Kasasbeh[2], Qusai Y. Shambour[3], Mosleh M. Abualhaj[4],
Mohmmad Abdalla Alsharaiah[5] and Sumaya N. Al-Khatib[1]
E-mail: a.abushareha@ammanu.edu.jo, bkasasbah@arabou.edu.sa, q.shambour@ammanu.edu.jo,
m.abualhaj@ammanu.edu.jo, m.sharaiah@ammanu.edu.jo, sumayakh@ammanu.edu.jo
*Corresponding author

[1]Department of Computer Science, Faculty of Information Technology
Al-Ahliyya Amman University, Amman, Jordan

[2]Faculty of Computer Studies, Arab Open University (AOU), Riyadh, Kingdom of Saudi Arabia

[3]Department of Software Engineering, Faculty of Information Technology
Al-Ahliyya Amman University, Amman, Jordan

[4]Department of Networks and Information Security, Faculty of Information Technology
Al-Ahliyya Amman University, Amman, Jordan

[5]Department of Data Science and Artificial Intelligence, Faculty of Information Technology
Al-Ahliyya Amman University, Amman, Jordan

Keywords: congestion, random early detection, active queue management

Received: February 2, 2022

Active Queue Management (AQM) methods control the router's buffer to maintain high network performance and control congestion at the router buffer. Random Early Detection (RED) method is the most well-known and the most utilized AQM. RED suffers from a high dropping rate, which motivates the later AQM methods to use more complex processes, which reach the limits of using fuzzy systems as a processing technique. Yet, high computational cost affects the router's performance specifically and the network as a whole, with so-called processing delay. In this paper, a linear version of RED (LRED) is presented to reduce the computational cost of the original RED and maintain the network performance in terms of throughput, delay, dropping, and loss. LRED is built based on two distinctive features, simplifying the congestion indicator calculation and reducing the operations in calculating the dropping probability. The experimental results showed that the proposed method reduces the delay and the processing time while maintaining the throughput and loss of the RED method.

Povzetek: "[Click here and Enter short Abstract in Slovene language]"

1 Introduction

Computer networks consist of multiple devices, one of which is the router. The router is responsible for receiving packets, accommodating, processing, and forwarding them to their destination. The packets are accommodated in the router within the router buffer. The problem of router buffer overflowing with the queued packets or increasing their numbers to extend affects the transmission process, degrades the network performance, and creates so-called congestion, as illustrated in Figure 1 [1]. Managing the queue in the router buffer to maintain high network performance is the general goal of congestion controller methods and active queue management (AQM). AQM methods addressed the goal mentioned above by controlling the queue size and determining the proper action for each arrival packet, queuing in the packet, or dropping it accordingly [2-4].

Among various AQM methods, Random Early Detection (RED) method is the most well-known and the most utilized [5]. RED monitors the queue size increase and detects congestion early, and it responds by dropping packets to maintain high performance. The process of packet dropping in RED is implemented randomly in a stochastic manner to avoid global synchronization. This random, stochastic-based dropping is also performed reasonably to avoid unnecessarily dropping packets. As illustrated in Figure 2, RED calculates and uses the average queue length (AQL) as a monitoring indicator. Besides, the value of the AQL is compared to two threshold values to determine the best decision for the arrival packet (i.e., dropping vs. accommodating). This process is controlled with other parameters, such as the maximum dropping value (D_{max}) and the counter value, which prevent global synchronization by reducing the chance of dropping subsequent packets [6].

Existing AQM methods maintain a random mechanism for packet dropping as implemented in RED.

RED has been enhanced following various directions: predicting early congestion, predicting false congestion, predicting and reducing delay, and easing the

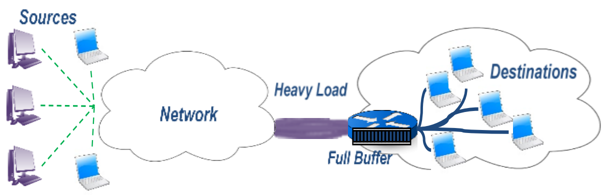

Figure 1: Congestion at the router buffer.

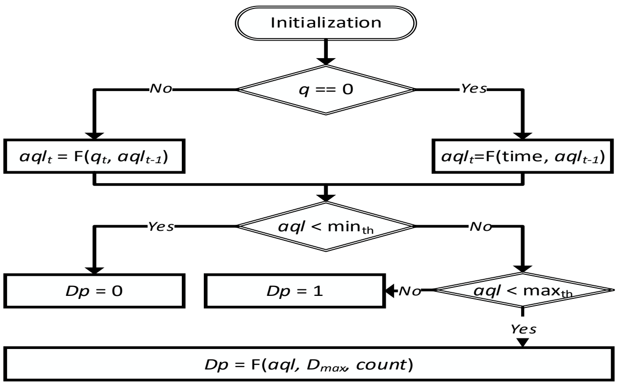

Figure 2: RED's mechanism for *Dp* calculation [7].

parameterization problem. Random dropping controlled by the value of dropping probability (Dp) is required to avoid global synchronization phenomena. The synchronized loop of dropping, stopping, and re-sending among senders degrades the network performance and wastes resources. Avoiding global synchronization by random packet dropping was firstly addressed by Floyd and Jacobson [8], as implemented in RED, the first AQM method. Up to now, this randomization has been embodied in all of the developed AQM methods. Optimized dropping is one of the key characteristics of the AQM that is continuously being enhanced in the developed AQM methods. An optimized dropping should speculate the current queue and load statuses and the expected future status, which can be estimated based on the current and previous load status. These considerations are addressed in the congestion indicator(s) [9, 10].

RED and the rest of the AQM methods have two main distinguished elements that differ from the other: the utilized congestion indicator and the underlying decision-making technique [11]. The decision-making technique involves various calculations to determine the dropping probability (*Dp*) and make the dropping decision. Accordingly, some methods aim to reduce the dropping probability with false congestion indications. Others aim to improve the delay. Some methods aim to ease the parameter settings, especially fuzzy systems [7, 12-15]. Overall, the advanced AQM methods addressed the problems of reducing dropping [5, 16, 17], delay [18, 19], loss [15, 20], and easing the parameter settings [21, 22], which are the main keys to achieve the desired performance of the AQM methods. Yet, one of the main significant issues is to keep the calculation at the router buffer as simple as possible to avoid high computational load and unnecessary delay at the router [23], which is not the case with complex AQM methods, especially the fuzzy-based methods [2, 4, 24-28]. Subsequently, RED is still the best method, as it is based on simple mathematical equations; however, simplifying RED is still demanded in the high network speed era.

This paper proposed a linear version of RED (LRED) based on a simple technique using AQL as the congestion indicator with a limited number of mathematical operations and faster calculation than the original RED. LRED uses a simple average queue length calculation and a simple mathematical equation to calculate the dropping probability and make the dropping decision. Besides, the proposed method considers the main purpose of queue management methods: to avoid congestion and maintain network performance. The rest of the paper is organized as follows: Section 2 discusses the previous queue management work. Section 3 presents the proposed work on the modified RED. The proposed method results in terms of network performance and the proposed method's computational costs are presented in Section 4. A discussion is presented in Section 5. Finally, the conclusion is given in Section 6.

2 Previous work

Early approaches for AQM focused on extending RED while maintaining the AQL utilization. Accordingly, Gentle Random Early Detection (GRED) [5] makes dropping decisions using the same parameters and technique while setting three thresholds compared to two in RED. Adaptive Random Early Detection (ARED) also implemented the same technique as in RED while adding target AQL as a parameter and maintaining the AQL around the target value. Later on, different methods were proposed by replacing the AQL with other indicators, such as arrival or load rates. A set of methods were proposed using the arrival rate and have extended from one version to another with the same technique; these are: Adaptive Virtual Queue (AVQ) [29], Stabilized AVQ (SAVQ) [30], and Enhanced AVQ (EAVQ) [31] used the arrival rate. Meanwhile, other methods replaced AQL with two indicators instead of one, one of which was the Efficient Random Early Detection (ERED) [32], which used AQL and q. As such, a modification of the original technique used by RED is required to accommodate different indicator(s).

Various methods were proposed with an adaptive mechanism to ease the parametrization problem, founded in RED and enlarged with the other methods. The adaptive mechanism calculates the values of the parameters as similar to the calculation of the indicator instead of using a fixed value. Accordingly, New Adaptive Random Early Detection (NARED) [33], Queue-Length Threshold Random Early Detection (LTRED) [9], and Priority Random Early Detection (PRED) [10] are adaptive congestion control methods. These methods, in general, did not focus on the results or improving the network performance. However, since setting up the optimal value for the non-adaptive parameters is difficult and mostly different values are required for different network statues, the adaptive methods' results were slightly better than the static methods.

The Fuzzy-based methods were proposed with a similar mechanism of using random dropping with probability based on *Dp*, but the *Dp* is calculated using a fuzzy inference process (FIP). These methods' goal was to ease the parameterization and create more flexibility in the dropping decision. Accordingly, these methods used the same indicators as earlier methods. Accordingly, different methods used different inputs with different fuzzy components (i.e., membership functions and rules). Among these components, the input variables are the most critical components for the fuzzy-based AQM. Examples of these methods are Fuzzy Explicit Marking (FEM) [34], Fuzzy BLUE (FB) [35], Fuzzy GREEN (FGREEN) [36], Fuzzy-logic Controller-based RED (FConRED) [37], and FLRED [4]. Although these methods were created to ease parameter initialization, other objectives can be achieved based on the input variables to be utilized, such as AQL [34], delay [4], and packet loss [35].

Overall, various AQM methods have been developed over the years. These methods were developed with one or more objectives: 1) Avoid global synchronization, which required dropping a non-ordered set of packets to avoid the synchronization loop. 2) Predicting congestion in early-stage and response by dropping packets to avoid or minimize packet loss in such a case. Predicting congestion at an early stage requires speculation of the network status. 3) Recognize the presence of short heavy traffic (false congestion) and distinguish it from the true congestion to avoid, or at least minimize, unnecessarily packet dropping, which requires network status speculation as predicting congestion in early stage. 4) Minimize delay to optimize the performance of data transmission by speculating the arrival and departure rate and the status of the queue. 5) Optimize parameter initialization problem, which affects the overall performance. These criteria are summarized in Table 1 [1, 7].

According to the AQM optimization criteria, which were mentioned previously, all known AQM methods, such as FRED [48], ARED [44], and GRED [5], achieved the first objective, which is avoiding global synchronization, by randomly dropping packets based on the calculated dropping probability. As for the second objective, which is early congestion prediction, different methods have different abilities. Methods that use the instant queue as a congestion indicator, such as SRED [23] and ERED [32], can achieve this criterion with noticeable changes in queue size. However, RED [8] cannot predict congestion as it depends on the average queue length over the system time. According to this criterion, BLUE [11] has the medium ability because it utilizes a single-capability indicator. As for recognizing false congestion, ERED [32] achieves this objective by using the average queue length with the instance queue, which together works as a time window over the queue size. On the other hand, AVQ [29], SAVQ [30], and EAVQ [31] did not partially or completely achieve this objective because they utilize single-capability indicators. As for delay consideration, LUBA [49] achieved this objective by considering the arrival and departure rates (traffic-load amount). On the other hand, RED [8] did not achieve this objective because the delay was not considered in the implemented technique. Finally, all FIP-based methods were proposed to address this criterion for parameterization problem easiness [50-51]

The results of the existing methods were reported using various parameters and various environments. Accordingly, the improvement can only be clarified by comparison to other methods. A summary of the results of the reviewed literature is presented in Table 2.

Overall, the existing AQM methods were developed over the years to address RED's problems, improve performance results, or achieve other objectives. Yet, one of the main disadvantages of extending RED is the processing overhead required to implement different congestion predicting and queue management. Accordingly, there is a need to reduce the computational cost while maintaining and improving the RED's network performance.

Table 1: AQM optimization criteria.

Criteria	Discussion	Consideration	Methods
Avoid Global Synchronization	When systematic dropping occurs, global synchronization is a loop of dropping, stopping, and re-sending among senders.	Consider a random dropping approach to avoid global synchronization.	All AQM methods
Early Congestion Predicting	Early congestion prediction leads to avoiding the consequence of congestion.	Consider the status of the network and the queue to predict congestion and respond early by dropping packets to avoid packet loss.	REM [38], MRED [39], BLUE [11], ERED [32], AQMRD [21], S-RED [40] and RED [8].
Predict False Congestion	False congestion identification discrimination leads to avoiding, or at least minimizing, unnecessarily packet dropping.	Consider the status of the network to be able to distinguish false congestion from the true one.	CRED [41], Adaptive Threshold RED [42] and Adaptive AQMRD [43].
Predict And Reduce Delay	Delay consideration is an important issue that causes by accumulating packets in the buffer.	Consider the arrival and departure rate and the queue status and drop packets accordingly if the delay increases.	GRED [5], ARED [44], DRED [45], PI [46], AVQ [29] and DQRED [3]
Ease Parameterization	Increasing the number of the parameters in the AQM methods creates the problem of parameter initialization and the performance-dependency problem.	Consider reducing the number of parameters or using low sensitive parameterization such as the fuzzy sets.	FuzzyRED [37], DeepBlue [47], Fuzzy logic for GRED (GREDFL) [2] and FLRED [4].

Table 2: Existing AQM methods.

Methods	Results
RED [8]	Improve link utilizations
NARED [33]	Improve packet loss rate compared to RED
LTRED [9]	A slight improvement over RED, AVQ, and ARED in terms of packet loss and throughput
BLUE [11]	Improve packet loss rate compared to RED
ERED [32]	Improved throughput compared to RED and BLUE
PRED [10]	Improve packet loss rate compared to RED
FEM [34]	Ease parameterization
FB [35]	Ease parameterization
FGREEN [36]	Ease parameterization
FConRED [37]	Ease parameterization
GRED [5]	Improve packet loss rate compared to RED
ARED [44]	Improve packet loss rate compared to RED
FRED [48]	Ease parameterization
AVQ [29]	Improve packet loss rate compared to RED
SAVQ [30]	Improve packet loss rate compared to RED
EAVQ [31]	Improve packet loss rate compared to RED
SRED [23]	Improve packet loss rate compared to RED
FuzzyRED [37]	Ease parameterization
FLRED [4]	Ease parameterization

3 Proposed work

Linear Random Early Detection (LRED) is proposed to reduce the original RED's computational cost while maintaining the network performance in terms of throughput, delay, dropping, and loss. The proposed method is built based on two distinctive features: 1) simplify the calculation of the congestion indicator, and 2) reduce the operations required for calculating the *Dp*. Accordingly, the adaptive estimated average queue length is calculated and used as the congestion indicator to guide the proposed method in estimating the network status. The dropping probability calculation is implemented based on a linear function.

The adaptive estimated average queue length is calculated as a function of the differences in the estimated length and the actual length. The advantages of using this proposed indicator are as follows: 1) calculating the estimated length value reduces the computational cost compared to the costly power operation that is utilized in the average queue length, 2) as similar to the average queue length, the calculated value is not sensitive to false congestion, given that the calculated value depends on the previously estimated value, and 3) the current queue length and sensitive to sudden congestion based on the actual queue length. The boundary between false and true congestion is identified concerning the actual queue length. Thus, the estimated queue length (eql) increases if the actual value increases rapidly when the difference in q and eql increases. In contrast, the value is reduced if the queue length is decreased or maintained. As such, the *eql* value is calculated as given in Equation 1.

$$eql = (q + eql)/2 \tag{1}$$

where q is the actual length, the *eql* is the estimated queue length. Given that the capacity is utilized, the *eql* will always be less than the actual length.

As for the dropping, the probability is calculated as a function of the congestion indicator in the range of [0-1]. However, an extended range has been adapted to linearize the output value in the proposed method. Yet, the extended range does not influence the dropping decision because any value outside the range [0-1] will have the same effect on the dropping decision as the effect of the edge values {0, 1}. Accordingly, the calculated value is output zero or negative as long as the estimated length is within a specific range. As mentioned before, the negative value will be considered zero in the implemented method. Thus, the original randomization dropping will preserve the same effect while improving the dropping decision-making. Besides, the calculated value that exceeded the value will be considered one and will preserve the effect of the original dropping. Accordingly, the dropping probability is calculated as a linear function of the estimated length, the safe range, the risky range, and the router buffer's capacity, as given in Equation 2.

$$Dp = \left(\frac{eql-s}{c-3s}\right) * counter \tag{2}$$

where *s* is the border of the safe and risky boundary, yet, the risky range is doubled the safe range as given in the Equation. Accordingly, the linear function divided the buffer into three regions; as the eql is below the value of s, the Dp will always be equal to or below zero. When the eql is greater than or equal to c-2s, the Dp will always be equal or greater than one. The counter is used to prevent global synchronization. The counter value is set to 1 when no dropping s occurred and reduced by 1/c when ta packet is dropped.

The proposed method calculates the value of the *eql*, which is then used to calculate the dropping probability with each packet arrival. The value of the dropping probability implicitly identifies the proper action to be made, either to accommodate the arrival packet with a dropping probability equal to zero, drop the arrival packet with a dropping probability equal to one, or stochastically drop the arrival packet with a dropping probability in the range (0-1). The proposed method implemented these steps as given in Algorithm 1.

Algorithm 1: LRED

```
INITIALIZATION- PARAMETER: s
INITIALIZATION- VARIABLE: eql:= 0, count:= 1
FOR-EACH Packet Arrival
      IF (q equal 0 AND eql equal 0  )
          eql=0
      ELSE IF (eql equal 0)
          eql=(q * q)/ Capacity
      ELSE IF (q equal 0)
          eql=eql/ Capacity
      ELSE
          eql=(q * eql)/ Capacity
      END-IF
      Dp = (eql-s)/(c-3s) *  counter
      IF (Drop(Dp) is TRUE)
          Drop-A, count -= 1/c
      ELSE
          counter:= 1
      END-IF
END-FOR
```

4 Results

A simulation using a discrete-time queue model is presented to evaluate the proposed method's performance compared to the existing methods. A set of features and parameters characterizes the simulation. The queue is simulated as First-In-First-Out (FIFO), the probabilities of the packet arrival are varied in the range of [0.3-0.95], the probability of packet departure is equal to 0.5, the number of slots for the discrete-time queue model is 2,000,000, and 800,000 of which are for the warm-up period, in which all the results are discarded. The capacity of the router buffer is 20. The proposed method is compared to the RED, ERED, and BLUE methods, with three sets of experiments. The first experiment experienced heavy traffic load, with arrival rate, α, set to the value of 0.95. The second experiment is implemented for moderate traffic with an arrival rate, α, which is set to 0.5. The third experiment examines low traffic load, with arrival rate, α, set to the value of 0.3. The simulation is conducted using Java in NetBeans Integrated Development Environment (IDE) 8.1, in a machine operated with 64bits Windows 10, in Intel Core i3 2.10 GHz processor and 6 GB RAM.

The results are illustrated in Figure 3, Figure 4, Figure 5, and Figure 6. All methods dropped no packet in light traffic nor expressed any packet loss or delay. As for moderate traffic simulation, the proposed method, LRED, drops fewer packets and does not lead to packet loss compared to the other methods. The proposed method outperformed the compared methods in terms of loss in high traffic load. The proposed method is more responsive compared to the other methods. Accordingly, it can be concluded that the AQM method can better monitor and manage the buffer queue even with linear functions and simple calculations. In detail, the proposed LRED outperformed RED in terms of packet loss and packet delay while maintaining the RED's same performance in terms of dropping and throughput.

Compared to the rest of the methods, LRED, RED, and BLUE outperformed ERED in packet loss, while BLUE has lower performance results than the LRED, RED, and ERED in packet dropping. Because of the high dropping rate, BLUE has achieved the best delay. Accordingly, the proposed LRED outperformed all the compared as it provides the best loss and throughput without extra delay or packet dropping. The second set of experiments is illustrated in Figure 7, Figure 8, Figure 9, and Figure 10, which confirm the findings of the previous results.

In the time comparison of the proposed, the time required to complete the whole experiment with each method, under the various traffic load, as an average of ten runs are summarized in Table 3. The proposed LRED reduces RED's high complexity; the time required for the process reached the best method in time requirement, which is the BLUE. The time analysis based on the operations conducted by each method in all cases is summarized in Table 4, in the worst case. The proposed method used less operation to complete the whole process than RED. As confirmed with the time comparison, the proposed method required less time than the RED and

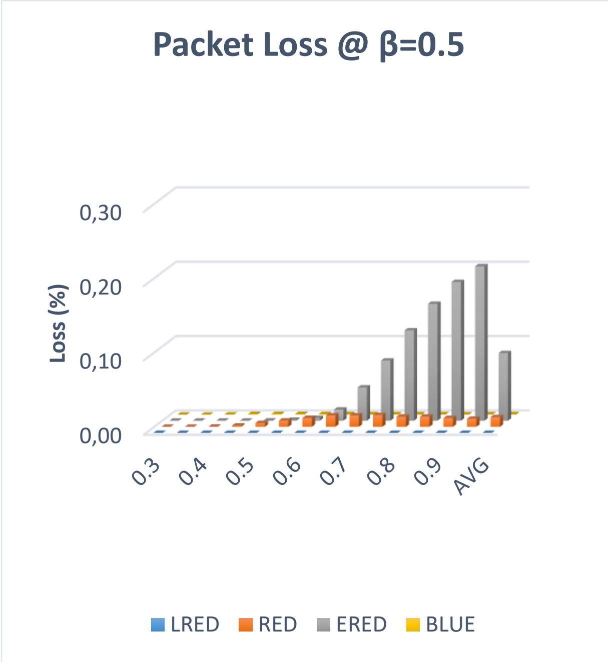

Figure 3: Packet loss comparison at moderate departure rate.

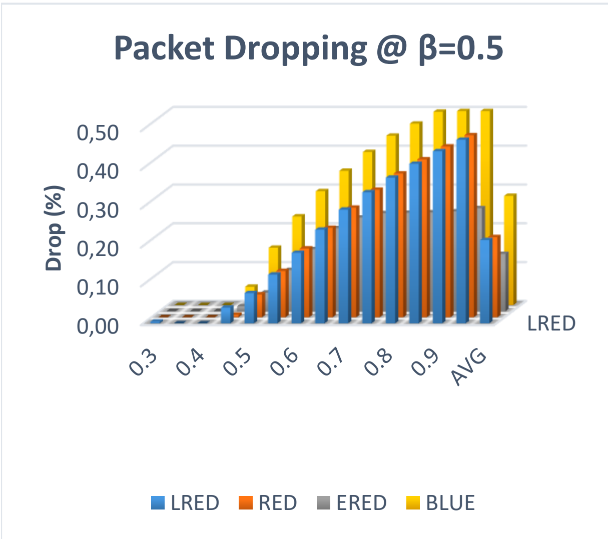

Figure 4: Packet dropping comparison at moderate departure rate.

ERED methods as it depends on two simple linear functions.

The fixed parameters and calculated variables used in each method are summarized in Table 5. The proposed method used the least number of parameters to complete the whole process and thus required less memory space.

As such, the proposed LRED achieved the intended goal by reducing the RED's complexity, improving the loss and delay performance, and maintaining the same performance in terms of dropping and throughput. Besides, the proposed method outperformed the compared methods, ERED and BLUE, based on the conducted experiments and the analytical results.

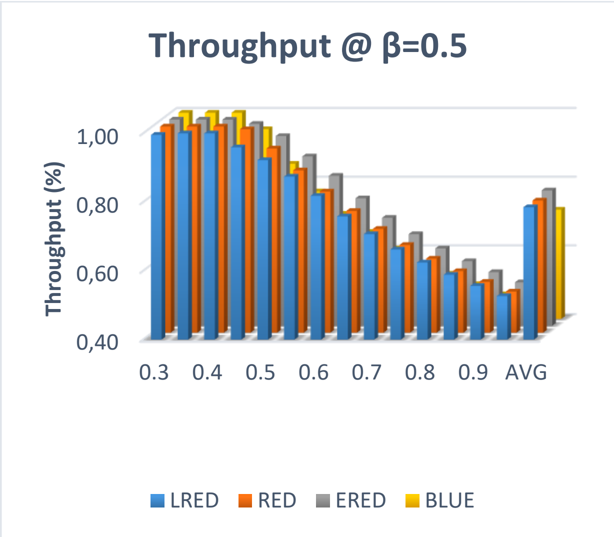

Figure 5: Throughput comparison at moderate departure rate.

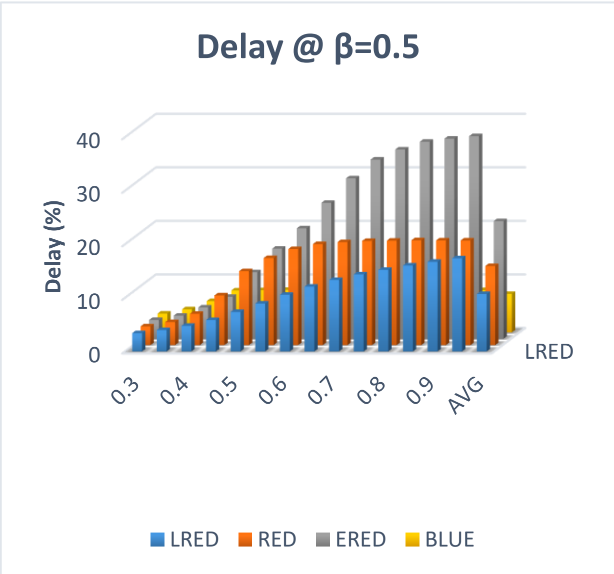

Figure 6: Delay comparison at moderate departure rate.

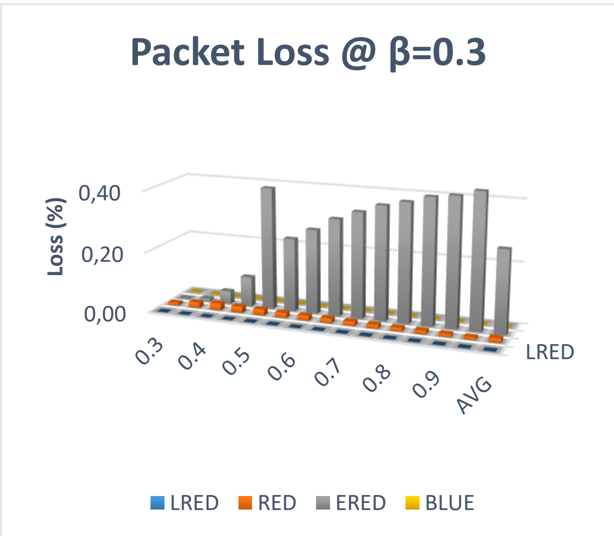

Figure 7: Packet loss comparison at low departuree rate.

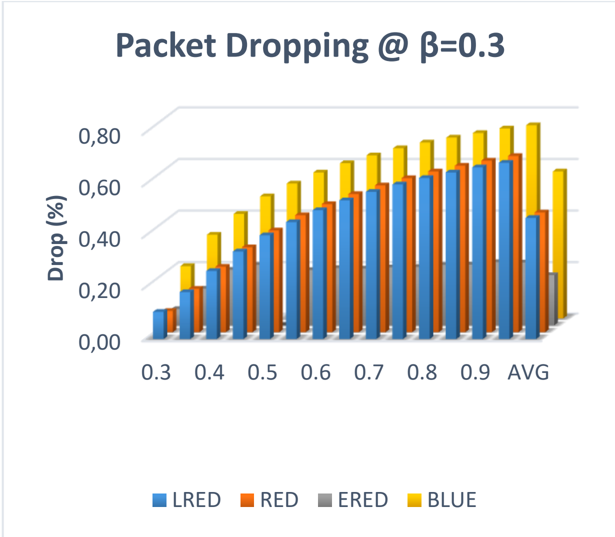

Figure 8: Packet dropping comparison at low departure rate.

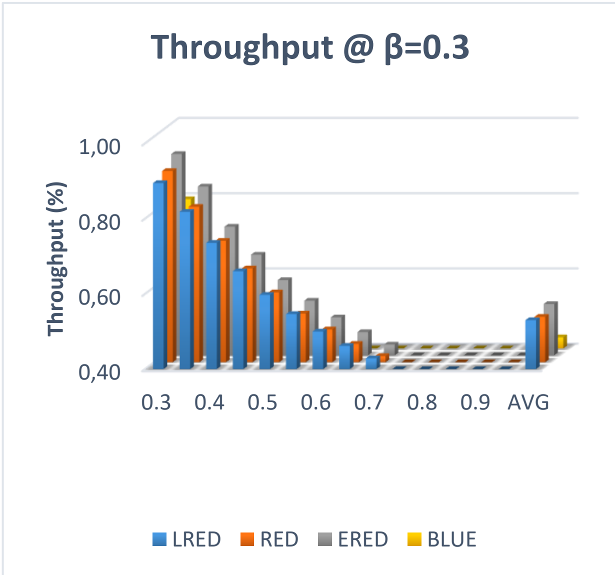

Figure 9: Throughput comparison at low departure rate.

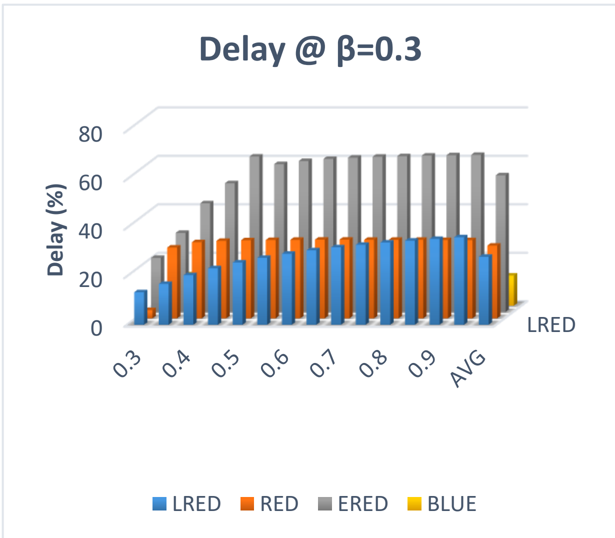

Figure 10: Delay comparison at low departure rate.

Table 3: Time comparison (LRED vs. AQM methods).

Moderate Departure				
	LRED	**RED**	**ERED**	**BLUE**
0.3	38	53	53	38
0.35	38	53	53	46
0.4	47	63	63	46
0.45	47	69	69	54
0.5	47	69	69	53
0.55	47	69	69	53
0.6	47	78	78	53
0.65	54	69	69	53
0.7	62	85	85	53
0.75	53	85	85	54
0.8	69	84	84	54
0.85	62	85	85	62
0.9	53	85	85	69
0.95	69	85	85	53
AVG	52.3571	73.7142	73.7142	52.9285
Low Departure				
	LRED	**RED**	**ERED**	**BLUE**
0.3	46	63	63	46
0.35	46	63	63	46
0.4	46	54	54	47
0.45	54	54	54	53
0.5	62	63	63	54
0.55	62	84	84	54
0.6	62	69	69	69
0.65	62	69	69	63
0.7	62	85	85	63
0.75	62	78	78	63
0.8	62	69	69	63
0.85	62	69	69	63
0.9	62	69	69	63
0.95	62	84	84	63
AVG	58	69.5	69.5	57.8571

5 Discussion

As presented in the results, the proposed method, LRED, improved packet loss similar to NARED [33], PRED [10], GRED [5], ARED [44], AVQ [29], SAVQ [30], EAVQ [31] and SRED [23]. The dropping rate was also improved, which improves the throughput, similar to ERED [32]. At the same time, LRED improves the calculation time by reducing the calculation complexity. Compared to the existing methods (See Table 2), such as GRED [5], and ARED [44], which add extra complexity to improve the throughput and loss.

Table 4: Operation Analysis between the Proposed and Existing AQM Methods.

		LRED	**RED**	**ERED**	**BLUE**
Indicator Calculation	Addition	1	2	0	2
	Multiplication	1	2	0	2
	Exponential	0	1	0	1
Dropping Calculation	Addition	2	3	3	3
	Multiplication	2	4	0	4
	Exponential	0	0	0	0
Total Calculation	Addition	3	5	3	5
	Multiplication	3	6	0	6
	Exponential	0	1	0	1

Table 5: Parameters analysis of the proposed and existing AQM methods.

	Fixed Parameters	**Calculated Variables**	**Total**
LRED	1	4	4
RED	4	10	14
ERED	4	10	14
BLUE	3	4	7

6 Conclusion

Existing methods for AQM suffer from the high computational cost required for its process, which in turn affects the processing delay and the network performance. A modified version of the Random Early Detection (RED) is proposed to maintain high network performance and control congestion at the router buffer while reducing the computational cost. The proposed method is built based on two distinctive features, simplifying the congestion indicator calculation and reducing the operations in calculating the dropping probability. The proposed method used a simple congestion indicator, the adaptive estimated average queue length, and low computation cost calculation for the dropping probability depending on a linear function.

References

[1] Abu-Shareha, A.A., "Enhanced Random Early Detection using Responsive Congestion Indicators," *International* Journal *of Advanced Computer Science and Applications (IJACSA)*, 2019, vol. 10 no. 3, pp. 358-367. https://doi.org/10.14569/ijacsa.2019.0100347.

[2] Baklizi, M., H. Abdel-Jaber, A. A. Abu-Shareha, M. 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*, 2014, vol. 16, no. 1, pp. 9-19.

[3] Mohammed, H., 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*, 2017, vol. 5 no. 02, pp. 37-49. https://doi.org/10.12785/ijcnt/050201.

[4] Abualhaj, M.M., A.A. Abu-Shareha, and M.M. Al-Tahrawi, "FLRED: an efficient fuzzy logic based network congestion control method," *Neural Computing and Applications*, 2018, vol. 30, no. 3, pp. 925-935. https://doi.org/10.1007/s00521-016-2730-9.

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

[6] Seifaddini, O., A. Abdullah, and H. Vosough, "RED, GRED, AGRED Congestion Control Algorithms in Heterogeneous Traffic Types," in *International Conference on Computing and Informatics*. 2013.

[7] Abu-Shareha, A.A., "Controlling Delay at the Router Buffer Using Modified Random Early Detection," *International Journal of Computer Networks & Communications (IJCNC)*, 2019, vol. 11, no. 6, pp. 63-75. https://doi.org/10.5121/ijcnc.2019.11604.

[8] Floyd, S. and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, 1993, vol. 1, no. 4, pp. 397-413. https://doi.org/10.1109/90.251892.

[9] Chhabra, K., M. Kshirsagar, and A. Zadgaonkar, "An Improved RED Algorithm with Input Sensitivity," in *Cyber Security*. 2018, pp. 35-45. https://doi.org/10.1007/978-981-10-8536-9_5.

[10] Sharma, N., S. S. Rajput, A. K. Dwivedi and M. Shrimali, "P-RED: Probability Based Random Early Detection Algorithm for Queue Management in MANET", in *Advances in Computer and Computational Sciences*. 2018, pp. 637-643. https://doi.org/10.1007/978-981-10-3773-3_62.

[11] Feng, W. C., D. Kandlur, D., Saha, and K. Shin, "BLUE: A New Class of Active Queue Management Algorithms," 1999, *University of Michigan*, Ann Arbor, MI, Technical Report.

[12] Koo, J., S. Ahn, and J. Chung, "A comparative study of queue, delay, and loss characteristics of AQM schemes in QoS-enabled networks," *Computing and Informatics*, 2004, vol. 23, no. 4, pp. 317-335.

[13] Liu, S., T. Başar, and R. Srikant, "TCP-Illinois: A loss-and delay-based congestion control algorithm for high-speed networks," *Performance Evaluation*, 2008, vol. 65, no. 6, pp. 417-440.

[14] Lim, L.B., L. Guan, A. Grigg, I. W. Phillips, X. G. Wang, and I. U. Awan, "Controlling mean queuing delay under multi-class bursty and correlated traffic," *Journal of Computer and System Sciences*, 2011, vol. 77, no. 5, pp. 898-916. https://doi.org/10.1016/j.jcss.2010.08.007.

[15] Wang, P., H. Chen, X. Yang, and X. Lu, "Active queue management of delay network based on constrained model predictive control," in *2011 Chinese Control and Decision Conference (CCDC)*, 2011. https://doi.org/10.1109/ccdc.2011.5968294.

[16] Feng, W. C., K. G. Shin, D. D. Kandlur and D. Saha, "The BLUE active queue management algorithms," *IEEE/ACM Transactions on Networking*, 2002, vol. 10, no. 4, pp. 513-528. https://doi.org/10.1109/tnet.2002.801399.

[17] Li, J.-S. and Y.-S. Su, "Random early detection with flow number estimation and queue length feedback control," *Journal of Systems Architecture*, 2006, vol. 52, no. 6, pp. 359-372. https://doi.org/10.1016/j.sysarc.2005.09.004.

[18] Stanojevic, R., R.N. "Shorten, and C.M. Kellett, Adaptive tuning of drop-tail buffers for reducing queueing delays," *IEEE Communications Letters*, 2006, vol. 10, no. 7, pp. 570-572. https://doi.org/10.1109/lcom.2006.224422

[19] Hadjadj-Aoul, Y., "Towards AQM Cooperation for Congestion Avoidance in DiffServ/MPLS Networks," *Recent Patents on Computer Science*, 2009, vol. 2, no. 1, pp. 1-13. https://doi.org/10.2174/1874479600902010001.

[20] Al-Diabat, M., H. Abdel-Jaber, F. Thabtah, O. Abou-Rabia and M. Kishta, "Analytical models based discrete-time queueing for the congested network," *International Journal of Modeling, Simulation, and Scientific Computing*, 2012, vol. 3, no. 01, pp. 1150004. https://doi.org/10.1142/s1793962311500048.

[21] Patel, S. and S. Bhatnagar, "Adaptive Mean Queue Size and Its Rate of Change: Queue Management with Random Dropping," *Telecommunication Systems*, vol.65, no. 2, 2016. https://doi.org/10.1007/s11235-016-0229-4.

[22] Tsavlidis, L., P. Efraimidis, and R.-A. Koutsiamanis, "Prince: an effective router mechanism for networks with selfish flows," *Journal of Internet Engineering*, 2016, vol. 6, no. 1.

[23] Ott, T.J., T.V. Lakshman, and L. Wong, "SRED: stabilized RED," in *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, 1999. https://doi.org/10.1109/infcom.1999.752153.

[24] Moghaddam, M.H.Y., "A fuzzy Active Queue Management mechanism for Internet congestion control," in *Third International Workshop on Advanced Computational Intelligence (IWACI)*, 2010, pp. 203-208. https://doi.org/10.1109/iwaci.2010.5585186.

[25] Mohammadi, S., H. M. Pour, M. Jafari and A. Javadi, "Fuzzy-based PID active queue manager for TCP/IP networks," in *International Conference on Information Sciences Signal Processing and their Applications (ISSPA)*, 2010, pp. 434-439. https://doi.org/10.1109/isspa.2010.5605462.

[26] 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.

[27] Nassar, K.A. and A.A. Abdullah, "Fuzzy RED to Reduce Packet Loss in Computer network," *Journal of AL-Qadisiyah for computer science and mathematics*, 2016, vol. 8, no. 1,pp. 107-114.

[28] Khatari, M. and G. Samara, "Congestion control approach based on effective random early detection and fuzzy logic," *CoRR*, 2017.

[29] Kunniyur, S. and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," *IEEE/ACM Transactions on Networking*, 2003, vol. 11, no. 5, pp. 689-702. https://doi.org/10.1109/tnet.2003.818183.

[30] Long, C. N., B. Zhao, and X.-P. Guan, "SAVQ: Stabilized adaptive virtual queue management algorithm," *Communications Letters*, IEEE, 2005, vol. 9, no. 1, pp. 78-80. https://doi.org/10.1109/lcomm.2005.1375248.

[31] Yanping, Q., L. Xiangze, L. Qi and J. Wei, "A stable enhanced adaptive virtual queue management algorithm for TCP networks," in *IEEE International Conference on Control and Automation (ICCA)* , 2007. https://doi.org/10.1109/icca.2007.4376380.

[32] Abbasov, B. and S. Korukoglu, "Effective RED: An algorithm to improve RED's performance by reducing packet loss rate," J*ournal of Network and Computer Applications*, 2009, vol. 32, no. 3, pp. 703-709. https://doi.org/10.1016/j.jnca.2008.07.001

[33] Zhang, J., W. Xu, and L. Wang, "An Improved Adaptive Active Queue Management Algorithm Based on Nonlinear Smoothing," *Procedia Engineering*, 2011, vol. 15, pp. 2369-2373. https://doi.org/10.1016/j.proeng.2011.08.444

[34] Chrysostomou, C., A. Pitsillides, G. Hadjipollas, A. Sekercioglu and M. Polycarpou, "Fuzzy Explicit Marking for Congestion Control in Differentiated Services Networks," in *Proceedings of the Eighth IEEE International Symposium on Computers and Communications*, 2003. https://doi.org/10.1109/iscc.2003.1214139

[35] Yaghwaei, M., M.B. Menhaj, and H. Amintoosi, "A fuzzy extension to the blue active queue management algorithm," in *Chinese Control and Decision Conference*, 2004. https://doi.org/10.1109/ccdc.2010.5498421

[36] Zargar, S.T., M.H. Yaghmaee and A.M. Fard, "Fuzzy proactive queue management technique,"in *Annual IEEE India Conference*, 2006, pp. 1-6. https://doi.org/10.1109/indcon.2006.302841

[37] Abdel-Jaber, H., M. Mahafzah, F. Thabtah and M. Woodward, "Fuzzy logic controller of Random Early Detection based on average queue length and packet loss rate," in *International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, 2008.

[38] Lapsley, D. and S. Low, "Random early marking: an optimisation approach to Internet congestion control," *IEEE International Conference on Networks (ICON '99)*, 1999. https://doi.org/10.1109/icon.1999.796161

[39] Koo, J., B. Song, K. Chung, H. Lee and H. Kahng, "MRED: a new approach to random early detection," in *International Conference on Information Networking*, 2001. https://doi.org/10.1109/icoin.2001.905450

[40] Yu-hong, Z., 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*, 2016, vol. 9, no. 2, pp. 375-384. https://doi.org/10.14257/ijunesst.2016.9.2.36

[41] Zhao, Y., Z. Ma, X. Zheng and X. Tu, "An Improved Algorithm of Nonlinear RED Based on Membership Cloud Theory," *Chinese Journal of Electronics*, 2017, vol. 26, no. 3, pp. 537-543. https://doi.org/10.1049/cje.2017.03.013

[42] Patel, Z.M., "Queue occupancy estimation technique for adaptive threshold based RED," in *IEEE International Conference on Circuits and Systems (ICCS)*, 2017. https://doi.org/10.1109/iccs1.2017.8326038

[43] Bhatnagar, S. and S. Patel, "A stochastic approximation approach to active queue management," *Telecommunication Systems*, 2018, vol. 68, no. 1, pp. 89-104. https://doi.org/10.1007/s11235-017-0377-1

[44] Floyd, S., R. Gummadi, and S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management," *AT&T Center for Internet Research at ICSI*, 2001.

[45] Aweya, J., M. Ouellette, and D.Y. Montuno, "A control theoretic approach to active queue management," *Computer Networks*, 2001, vol. 36, no 2-3, pp. 203-235. https://doi.org/10.1016/s1389-1286(00)00206-1

[46] Hollot, C.V., V. Misra, D. Towsley and Wei-Bo Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2001. https://doi.org/10.1109/infcom.2001.916670

[47] Masoumzadeh, S.S., G. Taghizadeh, K. Meshgi and S. Shiry, "Deep Blue: A Fuzzy Q-Learning Enhanced Active Queue Management Scheme," in *International Conference on Adaptive and Intelligent Systems (ICAIS'09)*. 2009, pp. 43-48. https://doi.org/10.1109/icais.2009.17

[48] Lin, D. and R. Morris, "Dynamics of random early detection," in *ACM SIGCOMM Computer Communication Review*, 1997, vol. 27, no. 4, pp. 127-137. https://doi.org/10.1145/263109.263154

[49] Agarwal, M., R. Gupta, and V. Kargaonkar, "Link utilization based AQM and its performance," in *Global Telecommunications Conference (GLOBECOM'04)*, 2004. https://doi.org/10.1109/glocom.2004.1378054

[50] Hilgers, U., P. Holleczek, P. and R. Hofmann, "Providing Quality of Service in Wide Area Networks," *Informatica (Slovenia)*. 2003, vol. 27, no. 3,pp. 361-371.

[51] Taenaka, Y., M. Tagawa, M. and K. Tsukamoto, "An experimental approach to examine a multi-channel multi-hop wireless backbone network," *Informatica (Slovenia)*, 2015, vol. 39, no. 4, pp. 365–374.