FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 7 No. 3, June (Special Issue), 2023, pp 33 - 39
DOI: https://doi.org/10.33003/fjs-2023-0703-1763

EXPONENTIAL-SELF-ADAPTIVE RANDOM EARLY DETECTION SCHEME FOR QUEUE MANAGEMENT IN NEXT GENERATION ROUTERS

***Yusuf Surajo, Aminu Bashir Suleiman, Usman Yahaya**

Department of Computer Science, Federal University Dutsin-Ma, Katsina State

*Corresponding authors' email: surajoyusuf65@gmail.com

ABSTRACT

As network traffic continues to grow exponentially, efficient queue management algorithms are essential for ensuring optimal performance in next-generation routers. Active Queue Management (AQM) scheme, has been advocated by the Internet research community for the next generation routers. Random Early Detection (RED) is the most well-known AQM scheme. However, RED lacks self-adaptation mechanism and it is susceptible to parametrization problem. Several variants of RED were developed, however all of them possess a static drop pattern; as such they are severely affected when a traffic load changes. To address the self-adaptation shortcoming of the RED and its variant schemes, Self-Adaptive Random Early Detection (SARED) scheme was developed. However, to avoid congestion, SARED aggressively drops packets once the queue length reached a certain maximum threshold limit, subsequently, this will increase the average queuing delay for networks with high traffic load conditions, therefore, to eliminate the aggressiveness of SARED in such situations, an Exponential version of SARED was proposed in this paper. Results of the simulation experiments carried out have indicated that in high traffic load situations, Exponential-SARED (ESARED) has significantly reduced average queuing delay by 4% and maximized average throughput by 3% compared to SARED and QERED.

Keywords: Congestion control, AQM, RED, SARED, QERED, ESARED

INTRODUCTION

With the rapid development of network traffic and the increasing need for high-speed data transmission, next-generation routers face significant challenges in efficiently managing their queues to ensure smooth and reliable network performance (Karmanje, A. R. et al., 2023, Atzori, L. *et al*, 2010 and Varghese, B. *et al*, 2017). Queue management algorithms play a significant function in maintaining optimal router performance by controlling the packet transmission and preventing network congestion (Floyd, S. and Jacobson, V., 1993 and Jain, R., 1990). Traditionally, routers used the tail-drop queue management strategy to avoid congestion. When a queue reaches capacity using a tail-drop technique, freshly arriving packets are dropped until there is adequate space in the queue for incoming traffic (Adamu, A., et al., 2021). However, due to issues with overflow, global synchronization, lock out, and bias towards bursty traffic, tail-drop does not completely control congestion (Karmeshu et al., 2017). For Internet routers to properly control queue lengths, offer lower queuing delay, and prevent global synchronization, the active queue management (AQM) strategy was presented to solve the difficulties of the tail-drop approach that were noticed (Karmeshu et al., 2017). By giving feedback to the sources to slow down their sending rate and the pace at which packets enter the queue when the queue is full, AQM attempts to utilise the queue more intelligently (Feng, C., et al., 2014).

Random Early Detection (RED) is a well-known active queue management algorithm widely used in current routers (Floyd, S. and Jacobson, V., 1993). RED manages congestion by dropping all arriving packets probabilistically before the router's queue fills up . The basic parameters used by RED algorithm are minimum and maximum thresholds (min_{th} and max_{th}), maximum packet-dropping probability (max_p), average queue length (avg) and weighted parameter (w). RED algorithm uses Exponentially Weighted Moving Average (EWMA) method applied to the current queue length to calculate the average queue length (avg) and drops packets probabilistically based on the calculated avg. Packets will be dropped with probability 0 when the calculated avg is below the min_{th} threshold value, and all the arriving packets will be dropped with probability 1 when avg is above max_{th} threshold value. However, packets will be dropped linearly from 0 to max_p when the calculated avg is between min_{th} and max_{th} (Floyd, S. and Jacobson, V., 1993; and Bonald *et al*, 2015).

However, RED algorithm suffers from some performance related issues such as large delay, low throughput, and insensitivity to traffic load because it drops at a very constant rate. The poor performance of the RED algorithm can be related with its linear drop function, that tends to be extremely aggressive when traffic load is low and not aggressive enough at high loads (Bonald, T., *et al*, 2015 and Korolkova, A., *et al*, 2019).

To address the weakness of RED, several improved variants of RED were developed, such as Gentle RED (Floyd S., 2000), Adaptive RED (Floyd, S., *et al*, 2001), Double Slope RED (Zheng, B., 2006), Nonlinear RED (Zhou, K., *et al*, 2006) Autonomous RED (Ho, H. J., & Lin, W. M., 2008), Cautious Adaptive (Tahiliani, M. P., *et al*, 2011), Improved Nonlinear RED (Zhang *et al.*, 2012), Three Section RED (Feng, C., *et al*, 2014), Adaptive queue management with random dropping (Karmeshu *et al*, 2017), Change Trend Queue Management (Tang, L. & Tan, Y., 2019), Quadratic RED (Kumhar, D., *et al*, 2021), Quadratic Exponential RED (Hassan, S., *et al*, 2023), etc. Despite all of these improvements, whenever there is changes in the traffic load, the impact of congestion control will be significantly impacted.

Self-Adaptive Random Early Detection (SARED), a novel RED variant, was proposed in (Adamu, A., et al., 2021) to solve the self-adaptive problem of RED and its variant schemes. Different drop patterns are included in the proposed SARED in (Adamu, A., et al, 2021) for various load conditions. As opposed to previous RED-enhanced variations, SARED takes into account the load condition at hand and intelligently modifies an appropriate drop pattern

and the maximum dropping probability to deliver the best performance possible. Results from (Adamu, A., et al, 2021) have demonstrated that SARED performs well no matter the load scenario. Even while SARED performs well under low, moderate, and high load situations, networks with persistent high load conditions will have an increase in average queuing delay with SARED.

This study proposes an Exponential-Self-Adaptive Random Early Detection (ESARED) algorithm, a unique method that uses exponential-based processes to improve the flexibility and effectiveness of queue management in next-generation routers, in order to solve these limitations. The ESARED algorithm offers a reliable solution for effective queue management in contemporary network infrastructures by combining the advantages of RED and exponential adaptation.

The rest of the paper has been organized as follows. In Section 2, the related works are described, while Section 3 introduces the suggested Exponential Self-Adaptive RED (ESARED). In Section 4, the findings of the experiments carried out for the study of the suggested method are reported. Finally, the conclusion of the paper is described in Section 5.

Related works

The Internet Engineering Task Force (IETF) recommends the Random Early Detection (RED) algorithm for next-generation routers since it is currently the most well-known AQM method for reducing network congestion (Floyd, S. and Jacobson, V., 1993). By calculating the average queue length (*avg*), the RED algorithm drops packets with a predetermined probability in order to alert sources to the earliest signs of network congestion. Using a basic exponentially weighted moving average (EWMA), the average queue length (avg), or average number of packets in the router's queue, is determined as shown in Equ. (1).

$$avg = ((1 - w_q) \times avg') + (w_q \times q) \quad (1)$$

Where q represents the instantaneous queue length; *avg* represents the calculated average queue length, and w_q is the pre-defined weight factor to calculate *avg* (*where* $0 < wq < 1$). The computed *avg* is compared with the minimum and maximum threshold values (min_{th} and max_{th}). All incoming packets are dropped with probability 0 (permitted into the queue) if $avg < min_{th}$. In the event that ($min_{th} \leq avg < max_{th}$), packets are dropped with a probability proportional to the average queue length, which rises linearly from 0 to max_p. However, all arriving packets are dropped with probability 1 when $max_{th} \leq avg$. Equ. (2) can be used to represent the drop function of RED, $p_d(avg)$ (Floyd, S. and Jacobson, V., 1993):

$$p_d(avg) = \begin{cases} 0, & avg < \min_{th} \\ \dfrac{avg - \min_{th}}{\max_{th} - \min_{th}} \bullet \max_p, & \min_{th} \leq avg < \max_{th} \\ 1, & \max_{th} \leq avg \end{cases} \quad (2)$$

Despite the fact that it has been demonstrated that the RED algorithm significantly outperforms the Tail Drop algorithm, research has shown that RED has some drawbacks, including parameterization issues (i.e., constant tuning of RED parameters is necessary to achieve an improved performance), low throughput, significant delays, and lack of a self-adaptation approach (Patel, S., 2013; Misra, V., *et al*, 2000; Plasser, E., *et al*, 2010 and Floyd, S., 2000).

Floyd proposed Gentle RED (GRED) in (Floyd, S., 2000) to increase the throughput of RED. After the max_{th} queue threshold in GRED, $2max_{th}$ was added. As in traditional RED, all incoming packets are dropped linearly with a probability that changes from 0 to max_p when the calculated *avg* is between min_{th} and max_{th}. Similarly, If the calculated average is between a maximum threshold (max_{th}) and twice maximum threshold ($2max_{th}$) then the probability changes from maximum dropping probability (max_p) to 1, making it more gentle than RED.

In order to improve RED's stability and robustness, Floyd et al. (Floyd, S., et al., 2001) presented a dynamic variant of RED named Adaptive RED (ARED). Adaptive RED dynamically adjusts max_p to maintain *avg* within the range of minimum threshold (min_{th}) and maximum threshold (max_{th}). In Adaptive RED, maximum dropping probability (max_p) is likewise restricted to stay between 0.01 and 0.5 range of threshold values. Adaptive RED implementation in routers will result in significant computing overhead, particularly in complex network environments with erratic traffic loads.

Double-Slope RED (DS-RED) was proposed by Zheng and Atiquzzaman so as to improve the queuing latency and throughput of Random Early Detection (RED) algorithm (Zheng, B., 2006). Two distinct drop functions are used by DS-RED to enhance RED's performance. However, the two distinct slopes produced by the linear dropping functions of DS-RED were found to behave remarkably similar to GRED (Floyd, S., 2000). Given that it relied on two linear dropping functions and sensitivity to parameters is still an issue in Double-Slope RED. Hence, RED's aggressiveness is carried over into DS-RED.

In (Ho, H. J., & Lin, W. M., 2008), Ho and Lin suggested Autonomous RED (AURED), an improved form of Adaptive RED (ARED). In contrast to ARED, AURED addresses the connection underutilization and overflows seen in ARED by using a transient distribution to instantaneous queue length as a congestion indicator. ARED can be thought of as an improved variant of AURED.

Additionally, to enhance the network's overall performance, Mohit et al. in (Tahiliani, M. P. et al, 2011) proposed the Cautious Adaptive Random Early Detection (CARED) algorithm, which constantly changes the maximum dropping probability (max_p) in accordance with the volume of traffic. The CARED algorithm's sensitivity to parameters is, however, identical to that of the ARED algorithm.

In (Zhou, K., et al., 2006), Zhou et al. presented Nonlinear RED (NRED), a modified version of RED. NRED used a non-linear quadratic drop function in place of RED's linear drop function to increase throughput. All other characteristics of RED are kept in NRED, with the exception of this change. They have held that NLRED is better than RED under low traffic loads, however, its aggressiveness increases when the traffic load is higher because of the nonlinear packet drop function used in NLRED. However, NLRED may cause forced packet drops and congestion in situations with extremely high loads.

An enhanced nonlinear RED (MRED) was suggested by Zhang et al. in (Zhang et al., 2012). While Nonlinear RED (NRED) employs a linear dropping function if the average queue length (*avg)* falls within the range of minimum threshold (min_{th}) and maximum threshold (max_{th}), however, if the average value of the queue length (*avg*) falls within the range of max_{th} and $2max_{th}$, then then the nonlinear dropping function is used in MRED (Floyd, 2000). In essence, NRED and GRED are believed to be upgraded by INRED.

Furthermore, Feng et al. Developed a Three-Section RED (TRED) in order to solve the issues of connection underutilization and forced packet drops of Random Early Detection algorithm as well as its improved variants (Feng, C., et al., 2014). TRED divides the queue length (i.e. min_{th} to max_{th}) of RED into three equal portions, in order to distinguish between various network loads. The traffic load is

considered to be low, if the average value of the queue length (*avg*) is in the initial portion; it is assumed to be moderate, if it is in the middle portion; and when it falls in the last area, it is assumed to be high. However, parameterization is still a challenge for TRED.

Another AQM algorithm Adaptive-Queue-Management with Random-Dropping (AQMRD), was suggested by Karmeshu et al. in (Karmeshu et al, 2017). The average value of the queue length (*avg*) as well as its rate of change are two factors that the AQMRD algorithm seeks to take into account. The maximum threshold max_{th}, the middle threshold mid_{th}, and the minimum threshold min_{th} are the next three thresholds established in AQMRD. However, the aggressive dropping method used in AQMRD results in low link utilization, lengthy delays, and a high loss rate.

Change Trend Queue-Management (CT-AQM) is a novel AQM algorithm that Tang and Tan designed so as to increase the reaction time of RED scheme (Tang, L. & Tan, Y., 2019). In order to determine packet drop probabilities, CT-AQM forecasts the change trend of queue size based on the change rate of the traffic load and average queue length (*avg*). According to the analysis's findings (Tang, L. & Tan, Y., 2019), CT-AQM was effective at reducing packet loss as well as improving network throughput at various scenarios with varying levels of traffic load. However, CT-AQM adds a lot of delay and permits more packets to enter the queue, which could cause packets overflow.

Kumhar et al. developed Quadratic RED (QRED) in order to enhance the RED algorithm's network performance in (Kumhar, D., et al, 2021). A nonlinear drop function similar to NRED in (Zhou et al., 2006) replaces the RED's linear dropping function in QRED. QRED employs the extra parameter Qmax to enhance RED's drop functionality. All other characteristics of NRED in (Zhou et al, 2006) are kept in QRED, with the exception of this modification. They have held that NLRED is kinder than RED under low traffic loads but more aggressive under big loads because of the nonlinear packet dropping function used in NLRED. However, QRED's aggression under heavy demand may cause congestion and overflow.

The Quadratic Exponential RED (QERED) algorithm was designed by (Hassan, S., et al., 2023) as an improved version of RED. QERED algorithm replaces RED's drop function with the combination of a nonlinear and exponential packet dropping function in the QERED algorithm. The issues with QRED and NRED algorithms, however, were carried over into QERED.

An enhanced RED scheme that addresses the self-adaptation issue with RED-based AQM schemes was proposed in (Adamu, A., et al., 2021). In order to attain the desired performance level, the self-adaptive method in (Adamu, A., et al, 2021) takes into account the present traffic load as well as the average queue length and adapts a suitable drop pattern and maximum dropping probability. This paper suggests an exponential variant of the SARED technique to significantly improve its performance under heavy loads.

Exponential Self Adaptive Random Early Detection Algorithm (ESARED)

An enhanced AQM algorithm is needed to deliver the intended network quality irrespective of the variations of network load while also preventing force drop and link underutilization. Numerous studies have shown that RED and its several enhancements failed to deliver optimized performance whenever load changes from one state to another in networks, so as to behave appropriately. Some networks perform well if the traffic load is light and perform poorly when traffic load is heavy, while some networks perform well at high load and perform poorly at low load. Nevertheless, load does fluctuate naturally in networks. The load of the network that has a direct impact on the observed average queue length, is typically ignored by several AQM algorithms in their definition of packet dropping probability.

An Exponential Self Adaptive RED (ESARED) algorithm has been proposed in this paper, that defines information packet dropping probability based on computed average queue length and current traffic load. ESARED takes into account all packets that arrive from the source nodes at a given time t ($\lambda_i(t)$), $i = 1, \ldots, M$, and M is the total number of active sources as depicted in Fig. 1. Equation (3) represents the overall number of arriving packets at the router per unit time t.

$$\lambda(t) = \sum_{i=1}^{M} \lambda_i(t) \quad (3)$$

The traffic load ρ (t) at a given time t is described in Equ. (4) and the bandwidth of the bottleneck link is denoted by μ.

$$\rho(t) = \frac{\lambda(t)}{\mu} \quad (4)$$

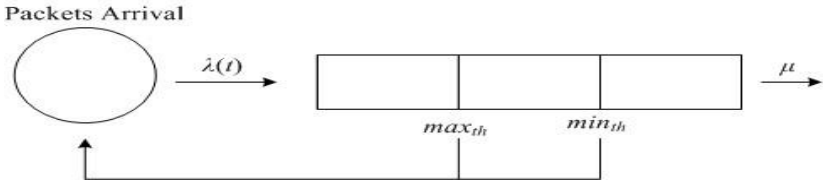

Figure 1: ESARED Queue Model

The system can be in any of the three (3) traffic load states (i.e., low, moderate and high). The underutilization of the link will happen when the system stays in the first-state ($\rho(t) < 1$) for an extended period of time because the traffic in the network is low and packets do not build up in the queue. When the system is in the second-state ($\rho(t) \approx 1$), the traffic in the network is moderate and performance is at its best. Although packets will build up in the queue and wait to be dispatched when the system is in the third-state ($\rho(t) > 1$,), if this state is sustained for an extended period of time, overflow and congestion are likely to occur.

The computed *avg* is compared with the minimum and maximum threshold values (min_{th} and max_{th}). Equ. (2) can be used to represent the drop function of RED, p_d (*avg*) (Floyd, S. and Jacobson, V., 1993):

In contrast to SARED, ESARED substitutes the values of min_{th} , *avg* , and max_{th} with $e^{min_{th}}$, e^{avg} and $e^{max_{th}}$ respectively, in order to enhance SARED's performance under heavy load situation. The Exponential Weighted Moving Average (EWMA) Equation (1) is used in ESARED to calculate the average queue length (*avg*). All incoming packets are dropped with probability 0 (permitted into the queue) if $avg < min_{th}$. In the event that ($min_{th} \leq avg < max_{th}$), packets are dropped with a probability proportional to the average queue length, which rises either linearly or nonlinearly from 0 to max_p. However, all arriving packets are

dropped with probability 1 when $max_{th} \leq avg$. Equation (5) presents the ESARED drop function.

$$p_{ESARED} = \begin{cases} 0, avg < min_{th} \\ \left(\frac{e^{avg} - e^{min_{th}}}{e^{max_{th}} - e^{min_{th}}}\right)^n max_p, min_{th} \leq avg < max_{th} \\ 1, avg \geq max_{th} \end{cases} \quad (5)$$

The current maximum drop probability (max_p) and exponent of the nonlinear drop function (n) are computed using equation (6) and equation (7) respectively.

$$max_p = 1 - e^{-\rho(t)} \quad (6)$$

$$n = c^{\frac{1}{\rho(t)}}, c \geq 2 \quad (7)$$

ESARED's drop function is presented in Figure 2.

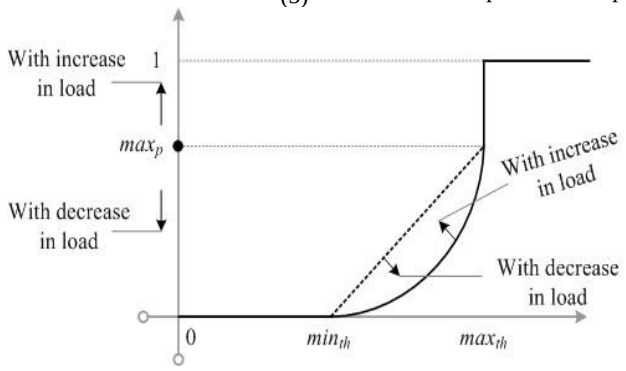

Figure 2: ESARED Drop Function

As shown in Figure 2, ESARED basically modifies its max_p dependent on the current load. In ESARED, a drop function whose exponent is likewise a function of load is created and grows linearly or nonlinearly from 0 to the present max_p when the average value falls between the min_{th} and max_{th}. These characteristics of ESARED allow it to function well under various load situations. Figure 3 presents the proposed ESARED algorithm's pseudocode.

Pseudocode of ESARED

Input: min_{th}, max_{th}, w, $\rho(t)$, n, c, μ
Output: *count*
Initialization:
$avg \leftarrow 0$
count ← -1
for all source nodes **do**
  Return $\lambda(t)$ [Mbps] (equation 3)
  $\rho \leftarrow \lambda(t)/\mu$ (equation 4)
  $n \leftarrow c^{1/\rho(t)}$ (equation 6)
  $max_p \leftarrow 1 - e^{-\rho(t)}$ (equation 7)
end for
for each packet arrival **do**
  Calculate the average queue size: *avg*
    if the queue is nonempty **then**
      $avg \leftarrow (1 - w) \times avg' + w \times q(t)$
    else
      $m \leftarrow f(t - t_{queue\_idle\_time})$
      $avg \leftarrow ((1 - w)^m \times avg')$
    end if
  Determine packet discard
    if $avg < min_{th}$ **then**
      No packet drop
      Set *count* ← -1
    else if $min_{th} \leq avg < max_{th}$ **then**
      Set *count* ← *count* + 1
      Calculate the packet drop probability P_b
        $P_b \leftarrow ((e^{avg} - e^{minth}) / (e^{maxth} - e^{minth}))^n \times max_p$
        $P_a \leftarrow P_b / (1 - count \cdot P_b)$
      Mark the arriving packet with probability P_a
      Set *count* ← 0
      Drop the packet
    else if $max_{th} \leq avg$ **then**
      Drop the arriving packet
      Set *count* ← 0
    else *count* ← -1
    When the router's queue becomes empty
      Set $t_{queue\_idle\_time} \leftarrow t$
    end if
end for

Figure 3: Pseudocode of ESARED Algorithm

Definitions of the saved, fixed and other parameters used in the proposed ESARED algorithm are presented in Table 1.

Table 1: ESARED Algorithm Parameters

Saved Parameters	Fixed Variables	Other
avg: present average queue length	max_{th}: maximum threshold for the queue	$q(t)$: instantaneous queue length
avg': previous average queue length	min_{th}: minimum threshold for the queue	$\lambda(t)$: total number of arriving packets (Mbps)
$t_{queue\_idle\_time}$: queue idle time	w: weighted queue	$\rho(t)$: traffic load
count: packets since last marked packets	k: nonlinear index	p_a: probability used for marking packets
i: exponent of the nonlinear drop function	μ: bottleneck link capacity (Mbps)	max_p: maximum drop probability

Simulation Experiments

The experiments were run on the NS2 simulator to measure how well the proposed ESARED algorithm performed. *N* TCP flows were created by *N* FTP sources sending packets to *N* destinations through a network of two routers (A and B) and a bottleneck link as shown in Figure 4. The bottleneck link has a 10 Mbps capacity and a 20 ms propagation delay. The hosts' propagation delays range between 5 and 10 ms and the routers are connected using a combined bandwidth of 10 Mbps.

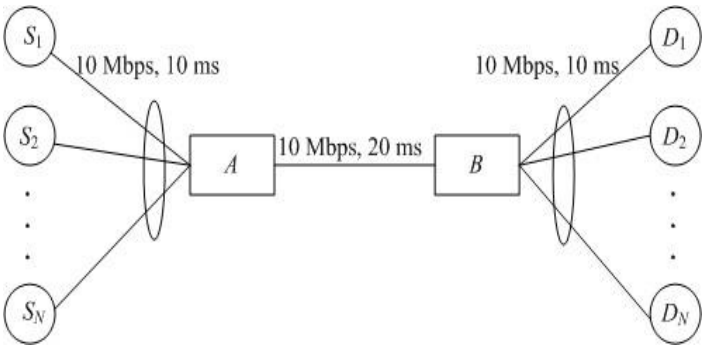

Figure 4: Simulation topology

Low, moderate, and high load states were used in the simulation experiments. N = 10 flows were employed when the traffic load is low, N = 50 flows were used if the traffic load is moderate, and N = 120 flows were deployed when the traffic load is high. The AQM scheme was installed at Router A with queue capacity of 140 packets. There was a new Reno TCP implementation. After 200 seconds, the simulation result was received. The traffic load $\rho(t)$ was calculated using the present value of λ (t) acquired from the queue monitoring object. The simulation parameters used in the experiments were k = 2, w = 0.002, min_{th} = 25 and max_{th} = 120 (Table 1). The performance of the proposed ESARED was compared with SARED and one of the most currently enhanced RED variants, i.e. QERED in (Hassan, S., et al., 2023). Since it has been established that QERED offers superior performance to RED, RED is not taken into account in this paper. max_p = 0.1 was employed for the analysis of QERED.

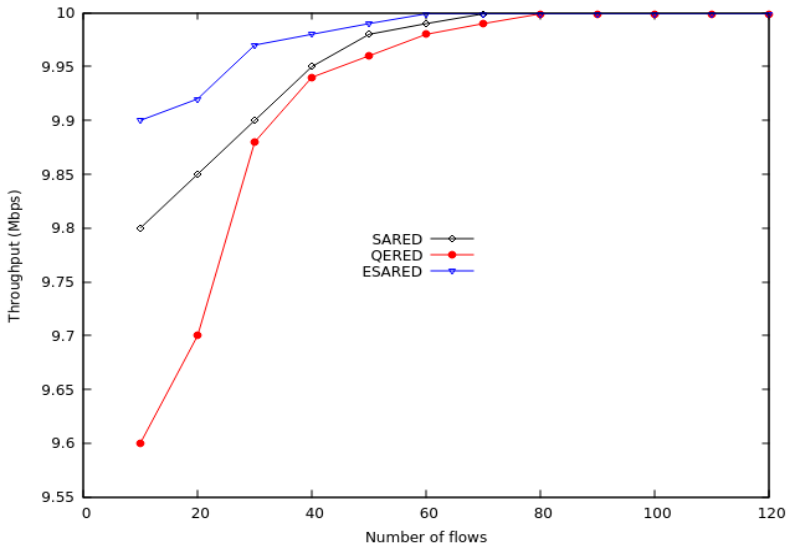

Figure 5: Throughput vs. Number of flows

The results shown in Figure 5 showed that ESARED maximizes throughput even at low load; however, in their respective high-load stages, QERED, SARED, and ESARED's average throughputs were getting close to their maximum limits.

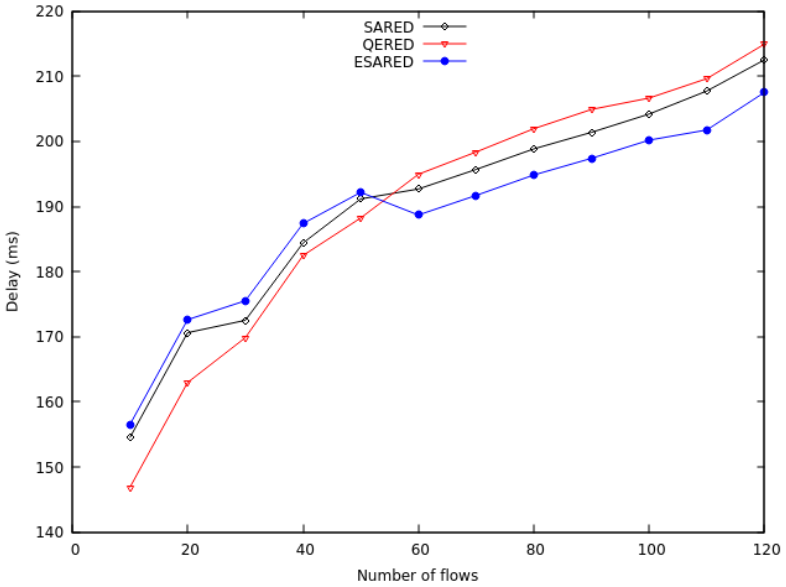

Figure 6: Delay vs. Number of flows

Furthermore, ESARED experienced longer delays than QERED and SARED at moderate and low load states as depicted in Fig. 6. This is due to the fact that, under these load conditions, ESARED had a lesser packet drop rate than SARED and QERED, allowing accumulation of many arriving packets in the queue. Because of this, packets using ESARED will experience more queuing delay than those using SARED and QERED, which will ultimately affect the total delay the packets experience. The delay seen with ESARED, however, was less than that of SARED and QERED at a high load. This is due to the fact that in that scenario, ESARED had a greater drop rate, which led to lower accumulation of packets in the queue. Moreover, the arriving packets then encountered a decreased delay with ESARED at high loads compared to SARED and QERED.

CONCLUSION

An Exponential Self-Adaptive RED (ESARED) scheme was developed in this paper as a way to enhance the performance of networks that are constantly under heavy load. According to the results of the simulation experiments, the proposed ESARED has reduced the average queuing delay by 4% and increased average throughput by 3% under high-load scenarios. Additionally, the results revealed that ESARED and SARED performed similarly in low- to moderate-load circumstances, with differences only being apparent when the traffic load reached an extremely high level.

REFERENCES

Adamu, A., Surajo, Y., Jafar, M. T. (2021). SARED: Self-Adaptive Active Queue Management Scheme for Improving Quality of Service in Network Systems. Computer Science 22(2), 253–267

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Bonald, T., May, M., Bolot, J., Bonald, T., May, M., Analytic, J. B., & Bonald, T. (2015). *Analytic evaluation of RED performance To cite this version : Analytic Evaluation of RED Performance*.

Cisco (2014). *Cisco delivers vision of fog computing to accelerate value from billions of connected devices*. Available at https://newsroom.cisco.com/press-release content?type=webcontent&articleId=1334100, accessed April 2023.

Kumhar, D. kumar, A. and Kewat, A. (2021). QRED: an enhancement approach for congestion control in network communications. *Int. J. Inf. Technol.*, vol. 13, no. 1, pp. 221–227, 2021, doi: 10.1007/s41870-020-00538-1.

Feng, C., Huang, L., Xu, C., & Chang, Y. (2014). *Analysis Based on Nonlinear RED*. 1–8.

Floyd S. (2000). Recommendation on using the gentle variant of RED. http://www.icir.org/oyd/red/gentle.html.

Floyd, Sally, & Jacobson, V. (1993). Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, *1*(4), 397–413. https://doi.org/10.1109/90.251892

Floyd, S., Gummadi, R., & Shenker, S. (2001). Adaptive RED: An algorithm for increasing the robustness of RED's active queue management. *Icsi*, 1–12. Retrieved from http://www.icsi.berkeley.edu/pubs/networking/adaptivered01.pdf

H. J. Ho and W. M. Lin, (2008). AURED - Autonomous Random Early Detection for TCP congestion control. Proc. - 3rd Int. Conf. Syst. Networks Commun. *ICSNC 2008 - Incl. I-CENTRIC 2008 Int. Conf. Adv. Human-Oriented Pers. Mech. Technol. Serv.*, pp. 79–84, 2008, doi: 10.1109/ICSNC.2008.22.

Jain, R. (1990). Congestion Control in Computer Networks: Issues and Trends. *IEEE Network*, *4*(3), 24–30. https://doi.org/10.1109/65.56532

Karmanje, A. R., Olanrewaju O. M., & Falalu, S., (2023). Corporate Network Security using Extended Access Control List (ACL) in a Simulation Environment. Fudma Journal of Sciences, 3(3), 264-269. Retrieved from https://fjs.fudutsinma.edu.ng/index.php/fjs/article/view/1568

Karmeshu, Patel, S., & Bhatnagar, S. (2017). Adaptive mean queue size and its rate of change: queue management with random dropping. *Telecommunication Systems*, *65*(2), 281–295. https://doi.org/10.1007/s11235-016-0229-4

Korolkova A., Kulyabov D., Velieva T., Zaryadov I. (2019): *Essay on the study of the self-oscillating regime in the control system*. Communications of the European Council for Modelling and Simulation, Caserta, Italy, pp. 473{480.

Tahiliani,M. P. , Shet, K. C. and T. G. Basavaraju (2012). CARED: Cautious Adaptive RED gateways for TCP/IP networks. *J. Netw. Comput. Appl.*, vol. 35, no. 2, pp. 857–864, 2012, doi: 10.1016/j.jnca.2011.12.003.

Misra, V., Gong, W. B., & Towsley, D. (2000). Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. *Computer Communication Review*, *30*(4), 151–160. https://doi.org/10.1145/347057.347421

Patel, S. (2013). Performance analysis and modeling of congestion control algorithms based on active queue management. *2013 International Conference on Signal Processing and Communication, ICSC 2013*, 449–454. https://doi.org/10.1109/ICSPCom.2013.6719832

Plasser, E., Ziegler, T., & Reichl, P. (2010). *On the Non-Linearity of the RED Drop Function*.

Hassan, S. and Oluwatope, A. (2014). Curvilinear red: An improved red algorithm for internet routers. *Proc. IASTED Int. Conf. Model. Simulation, AfricaMS 2014*, no. September, pp. 154–160, 2014, doi: 10.2316/P.2014.813-025.

S. Hassan, A. Rufai, V. Nwaocha, S. Ogunlere, A. Adegbenjo, M. Agbaje, T. Enem (2023). Quadratic exponential random early detection: a new enhanced random early detection-oriented congestion control algorithm for routers. International Journal of Electrical and Computer Engineering (IJECE) Vol. 13, pp. 669~679 ISSN: 2088-8708, DOI: 10.11591/ijece.v13i1.pp669-679

Tang, L., & Tan, Y. (2019). Adaptive queue management based on the change trend of queue size. *KSII Transactions on Internet and Information Systems*, *13*(3), 1345–1362. https://doi.org/10.3837/tiis.2019.03.013

Varghese, B., Wang, N., Nikolopoulos, D. S., & Buyya, R. (2017). *Feasibility of Fog Computing*. Retrieved from http://arxiv.org/abs/1701.05451

Zheng B. (2006). DSRED: A New Queue Management Algorithm for the Next Generation Internet. *IEICE Transactions on Communications*, *E89-B*(3), 764–774. https://doi.org/10.1093/ietcom/e89-b.3.764

Zhou, K., Yeung, K. L., & Li, V. O. K. (2006). Nonlinear RED: A simple yet efficient active queue management algorithm. *Computer Networks*, *50*(18), 3784–3794. https://doi.org/10.1016/j.comnet.2006.04.007