ENHANCING NETWORK PERFORMANCE WITH IMPROVED RANDOM EARLY DETECTION

Vu Van Dien

University of Information and Communication Technology, Thai Nguyen University, Vietnam

ARTICLE INFORMATION

Journal: *Vinh University Journal of Science Natural Science, Engineering and Technology*
p-ISSN*: 3030-4563*
e-ISSN: *3030-4180*

Volume: 54
Issue: 1A
Correspondence: *vvdien@ictu.edu.vn*

Received: *13 September 2024*
Accepted: *18 December 2024*
Published: *20 March 2025*

Citation:
Vu Van Dien (2025). Enhancing network performance with improved random early detection. ***Vinh Uni. J. Sci.*** *Vol. 54 (1A), pp. 5-13 doi: 10.56824/vujs.2024a083a*

ABSTRACT

Congestion has become an important issue affecting the performance of network systems. Active Queue Management (AQM) algorithms are crucial in ensuring network stability. The first active queue management technique deployed to control congestion was Random Early Detection (RED). RED compares the average queue length with predefined thresholds to mark or discard packets. DyRED [1] has improved RED based on changing the upper threshold. The author proposes an enhanced DyRED algorithm called ImRED (Improved DyRED) to enhance network performance further. Through simulation evaluation on the NS2, the author observed that ImRED outperforms DyRED regarding packet loss rate and average queue delay.

Keywords: Active queue management; congestion; average queue size; RED; threshold.

1. Introduction

The Internet is a global information-sharing system. It is built upon the IP foundation to provide data transmission services to users. This network has grown rapidly in recent years, leading to increased congestion within the network. A packet sent to a network node will be put in the node's buffer or queue before being processed. Retrieving packets from the queue depends on the principle of serving requests in the queue. Network congestion is detected when a node's queue becomes full, causing incoming packets to be lost (dropped). Network congestion is an important issue affecting the quality of network services [1], [2]. Minimizing packet loss rate and reducing queue delay, thereby decreasing average latency, are important objectives in enhancing the quality of service on the IP network.

TCP is a reliable, connection-oriented data transmission protocol widely used on the Internet. It offers a control mechanism to prevent network congestion. TCP uses several techniques to avoid congestion and improve network performance [3], [4].

Various algorithms have been proposed to address the issue of network congestion, including DropTail and Active Queue Management (AQM) strategies. DropTail processes packets in the queue using the First In, First Out (FIFO) principle. When the queue of a node becomes full, the packets that come later will be dropped. AQM is extensively used on routers. It detects congestion early, does not wait until the queue is complete, and drops packets with a certain probability.

Along with dropping packets, the router notifies some sending nodes to adjust their transmission rates. Instead of waiting for the queue to become whole, all nodes must reduce their transmission rates. This results in a decrease in the number of packets lost and an increase in the average throughput. RED [5], proposed by Floyd and Jacobson, is the first dynamic queue management strategy. Then, many more strategies were proposed, including BLUE, ERED, ENRED, and DyRED. RED avoids early congestion by using the average queue size (*avg*). This parameter is computed based on the current queue size (*q*), queue weight (*wq*), and previous *avg*. It is recalculated every time a packet arrives. Then, *avg* is compared to the values of two thresholds. If *avg* is lower than the lower threshold, no packets are dropped. If *avg* is higher than the upper threshold, packets are dropped. If *avg* is between the two thresholds, discard the packet according to the probability calculated based on *avg* and the value of the thresholds.

RED has shown better network performance than DropTail in terms of parameters such as packet loss rate, average delay, and average throughput. However, when network traffic suddenly increased, RED proved ineffective in reducing the number of lost packets and the average queue delay [6], [7], [8], [9], [10]. DyRED [1] solved the above problem. This research will focus on improving DyRED to achieve better network performance.

2. RED, DyRED, and some other active queue management strategies

2.1. RED

The RED algorithm [5] introduces a different network congestion control strategy than the previous method, DropTail. If DropTail drops packets only when the node's queue is complete, then RED will drop packets before the queue is full. This reduces the number of dropped packets and avoids the phenomenon of source nodes simultaneously reducing the window size. For each packet that arrives at the router, the average queue size is calculated in cases where the queue is empty, and the queue has packets in it. Then, this *avg* is compared with two thresholds (upper threshold max_{thr}, lower threshold min_{thr}) in the router's buffer to decide whether to drop the packet. If so, what is the probability?

Thus, two computations are performed within RED: calculating the average queue size and determining the packet drop probability. The average queue size is calculated based on the queue weight, the current queue size q, and the previous *avg* using the following formula:

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

The probability of dropping a packet is calculated according to the following formula:

$$p_b = \begin{cases} 1, & if\ avg \geq max_{thr} \\ \frac{max_p(avg - min_{thr})}{max_{thr} - min_{thr}}, & if\ min_{thr} \leq avg \leq max_{thr} \\ 0, & if\ avg < min_{thr} \end{cases} \quad (2)$$

Where max_p is the maximum value for p_b.

The general algorithm of RED [5] is described as follows:

For each packet arrival
    Calculate the average queue size *avg*
    if $avg < min_{thr}$
        Accept incoming packet
    else if $min_{thr} \leq avg < max_{th}$
        Calculate probability p_a
        With probability: mark the arriving packet
    else mark the arriving packet

RED has detected congestion early, reduced packet loss, and avoided global synchronization. Subsequently, many RED improvement algorithms have been proposed to increase network performance when traffic suddenly increases.

2.2. DyRED

DyRED is an enhanced RED strategy proposed by Danladi *et al* [1]. It extends RED by recomputing *avg*. When comparing *avg* with two thresholds, if *avg* is smaller than the lower threshold or greater or equal to the upper threshold, it keeps the *avg* unchanged. Moreover, if *avg* is between the lower and upper thresholds, it adjusts the upper threshold. The upper threshold is calculated based on the average queue size. Through simulation and evaluation, DyRED performs better than RED regarding network performance parameters such as a smaller packet loss rate and higher throughput.

2.3. ERED

ERED (Enhanced RED) is another improved RED algorithm [6]. While DyRED focuses on adjusting the upper threshold, ERED modifies the average queue size to improve network performance. The calculation of *avg* uses two parameters, α and β. If q is between the two thresholds, calculate *avg* as RED by comparing the current queue length with the two thresholds. However, if q is outside the range of two thresholds, the formula for calculating *avg* is adjusted. Through simulation, evaluated with the values of two parameters, α and β, equal to 1.1, ERED gave better results than RED regarding packet loss rate and network stability.

2.4. ENRED

ENRED (Enhanced Random Early Detection) was proposed by Ismail *et al* [7]. It also focuses on adjusting the average queue size. ENRED uses the queue target parameter (qt) and a queue weight to calculate the *avg*. This qt parameter is calculated as the difference between the current queue size and the average of the upper and lower thresholds. The average queue size (*avg*) is adjusted according to the following formula:

$$avg = qt.(1 - w_q) + q.(qt - w_q) \quad (3)$$

3. Improvement of the DyRED strategy

Although the RED strategy is more straightforward and practical than DropTail, it performs poorly when network traffic increases dramatically. DyRED has been proposed to improve network performance even in cases of increased network traffic [1]. The DyRED strategy improves RED by adjusting the upper threshold when the queue size is between two thresholds. The author proposed improving the DyRED strategy to improve network system performance in this study. The improvement proposed here is an extension of DyRED. ImRED combines adjustments to the lower threshold, upper threshold, and average queue size to control congestion in a router's cache early before its queue becomes full. This proposed improvement aims to reduce the packet loss rate and the average queuing delay in the following congestion cases: Heavy congestion and light congestion. ImRED extends DyRED by combining upper and lower threshold tuning based on the average queue size and recalculating the *avg* parameter when comparing its current value with the two thresholds. *Avg* is calculated each time a packet arrives based on the current queue size and the previously calculated *avg*. The two thresholds and *avg* are adjusted according to the following formulas:

$$\boldsymbol{avg} = \frac{\boldsymbol{avg}.(\mathbf{1}-\boldsymbol{w_q})}{\boldsymbol{m}} + \boldsymbol{q}.\boldsymbol{w_q}, \boldsymbol{with}\ \boldsymbol{m} > \mathbf{1} \quad (4)$$

$$\boldsymbol{min_{thr}} = \boldsymbol{n}.\boldsymbol{avg} \quad , \ \boldsymbol{with}\ \boldsymbol{m} > \mathbf{1} \quad (5)$$

$$\boldsymbol{max_{thr}} = \frac{\boldsymbol{avg}}{\boldsymbol{p}} \quad , \ \boldsymbol{with}\ \boldsymbol{t} > \mathbf{0} \quad (6)$$

Where m, n, and p are chosen logically to achieve a low average queue delay.

The algorithm checks whether the queue is empty for each incoming packet and gives the formula to calculate *avg*. Then, compare the value of *avg* with the two thresholds. If *avg* is smaller than the lower threshold, adjust *avg* and the lower threshold according to *avg*. If *avg* is between the two thresholds, it adjusts the upper threshold and removes packets with a certain probability. However, if *avg* is greater than or equal to the upper threshold, it discards the packet.

The proposed method is described in detail as follows:

Initialization :
avg = 0 ; count = -1
for each packet arrival
calculate the new average queue size avg:
if the queue is not empty
$avg = (1 - w_q).avg + q.w_q$
if $(avg < min_{th})$ $\boldsymbol{avg} = \frac{\boldsymbol{avg}.(\mathbf{1}-\boldsymbol{w_q})}{\boldsymbol{m}} + \boldsymbol{q}.\boldsymbol{w_q}$
else
$u = f(time - q\_time)$
$avg = (1 - w_q)^u.avg$
if $min_{thr} \leq avg < max_{th}$
$count = count + 1$
$\boldsymbol{max_{thr}} = \frac{\boldsymbol{avg}}{\boldsymbol{p}}$
calculate probability p_a *:*

$$p_b = \frac{(avg - min_{thr})}{max_{thr} - min_{thr}} max_p$$

$$p_a = \frac{p_b}{1 - count.p_b}$$

with probability p_a *:*

mark incoming packet

count = 0

else if $avg \geq max_{thr}$

mark incoming packet; count = 0

else count = -1; $\boldsymbol{min_{thr} = n.avg}$

When the queue becomes empty: q_time = q

Variables:

q_time: start of the queue idle time.

count: the number of packets arriving immediately after the last marked packet.

p_a: Probability of marking the current packet.

f(t): Linear function of time t.

time: current time.

4. Simulation results and performance evaluation

4.1. Simulation environment

In this study, the author conducted simulations of DyRED and the proposed enhancement method, ImRED, using the NS2 network simulation software. The network's topology diagram was designed as shown in Figure 1.

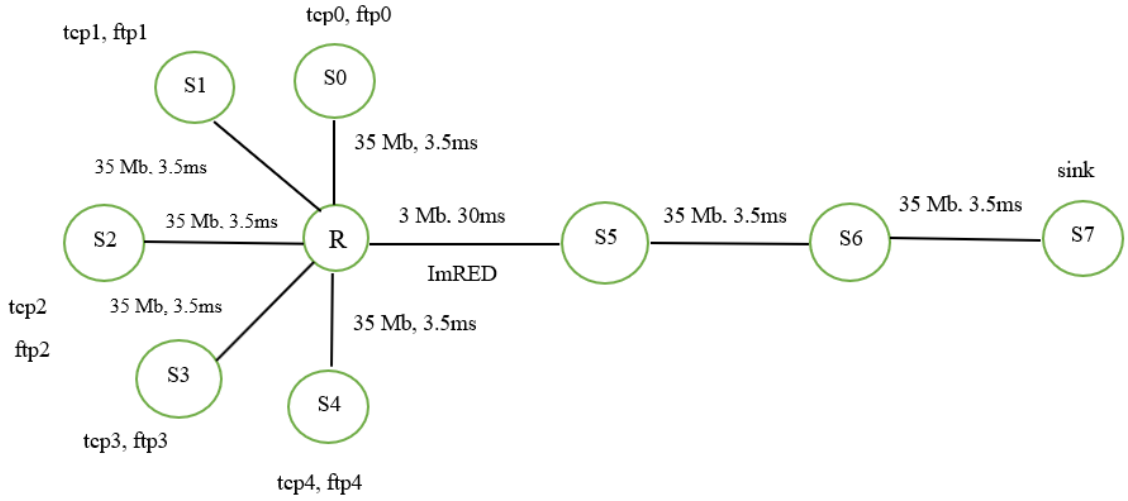

Figure 1: *Topology diagram of the simulated network*

The bandwidth and delay of the links are 35 Mbps and 3.5 ms, respectively, except for the R-S5 link, which has a bandwidth of 3 Mbps and a delay of 30 ms. The queue size for the R-S5 link is 50 packets. The full-duplex link between two network nodes, R and S5, uses a queue type called DyRED or ImRED. Parameters such as bandwidth, latency, queue size, start time of traffic generation, stop time of traffic generation, simulation duration, etc., must be kept consistent across all simulations for comparison purposes.

4.2. Average queuing delay for packets

The average queue delay of the packets is determined by dividing the total queue delay of the packets by the total number of packets. This delay is determined in two scenarios: heavy congestion and light congestion. The collected comparison results are presented as shown in Figures 2 and 3.

4.2.1. Scenarios 1: heavy congestion

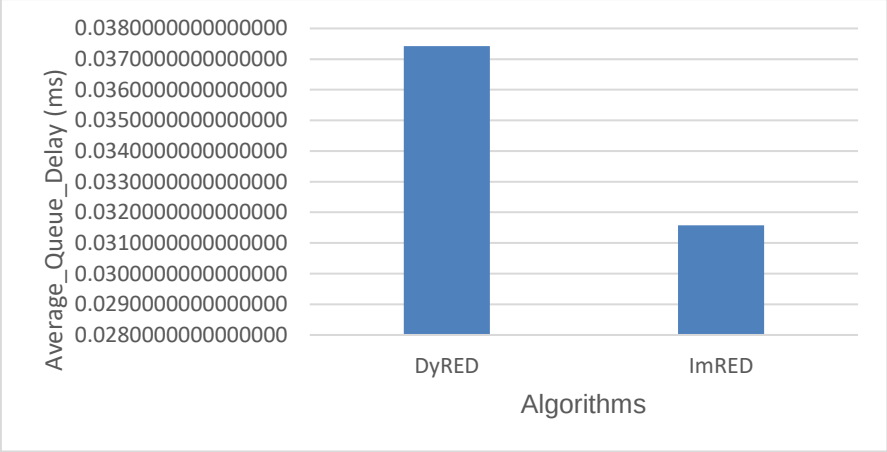

Figure 2: *Average queue delay of the packets under heavy congestion*

4.2.2. Scenarios 2: light congestion

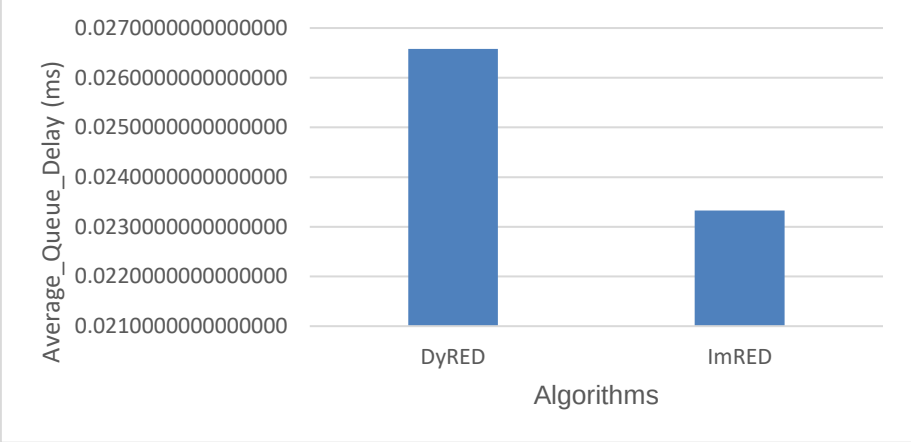

Figure 3: *Average queue delay of the packets under light congestion*

Through simulation evaluation, we see that in both scenarios, the average queue delay of packets in ImRED is lower than in DyRED: for heavy congestion, it is 15.6%, and for light congestion, it is 12.2%. The more severe the congestion, the greater the average packet queue delay.

4.3. Packet drop rate

The packet drop rate is determined by the total number of dropped packets divided by the total number of packets arriving in the queue.

This packet drop rate is determined in two scenarios: heavy congestion and light congestion.

4.3.1. Scenarios 1: heavy congestion

The values of the total number of packets arriving in the queue, the total number of dropped packets, and the packet drop rate of both strategies under heavy congestion are shown in Table 1.

Table 1: *Compare the dropped packets under heavy congestion*

DyRED algorithm			**ImRED algorithm**		
Total incoming packets	**Total dropped packets**	**Percentage of dropped packets**	**Total incoming packets**	**Total dropped packets**	**Percentage of dropped packets**
321538	5145	1.600	331698	2143	0.646
321402	5095	1.585	332082	2458	0.740
321524	5114	1.590	331516	2239	0.675
321032	5136	1.599	331616	2308	0.695
320794	5105	1.591	331568	2127	0.641

4.3.2. Scenarios 2: light congestion

The values of the total number of packets arriving in the queue, the total number of dropped packets, and the packet drop rate of both strategies under light congestion are shown in Table 2.

Table 2: *Compare the dropped packets under light congestion*

DyRED algorithm			**ImRED algorithm**		
Total incoming packets	**Total dropped packets**	**Percentage of dropped packets**	**Total incoming packets**	**Total dropped packets**	**Percentage of dropped packets**
283158	2447	0.864	290930	1297	0.445
276668	2316	0.837	283432	1421	0.501
282854	2298	0.812	290216	1304	0.449
273412	2250	0.822	279650	1296	0.463
258942	2206	0.851	265864	1284	0.482

Under heavy congestion, the packet drop rate of ImRED decreases significantly compared to DyRED, with a reduction of over 50%. Meanwhile, under light congestion, this reduction is more than 40%. This result is achieved because ImRED reduces the value of *avg* whenever it falls below the lower threshold, which corresponds to the scenario where the packet drop probability is set to 0.

5. Conclusions

This study has presented an enhanced congestion control method called ImRED aimed at improving the performance of DyRED. ImRED combines the adjustment of two

thresholds and the average queue size. Simulation results and evaluations show that ImRED outperforms DyRED in reducing the packet drop rate and decreasing the average queue delay for packets. This leads to a reduction in packet transmission delays between nodes within the network. As a result, ImRED significantly enhances network system performance.

REFERENCES

[1] S. B. Danladi and F. U. Ambursa, "DyRED: An Enhanced Random Early Detection Based on a new Adaptive Congestion Control," in *Proc. 15th Int. Conf. Electron. Comput. Comput.*, Abuja, Nigeria, 2019. DOI: 10.1109/ICECCO48375.2019.9043276

[2] A. M. Alakharasani, M. Othman, A. Abdullah, and K. Y. Lun, "An Improved Quality-of-Service Performance Using RED's Active Queue Management Flow Control in Classifying Networks," *IEEE Access*, vol. 5, pp. 24467-24478, 2017. DOI: 10.1109/ACCESS.2017.2767071

[3] R. J. La, P. Ranjan, and E. H. Abed, "Analysis of Adaptive Random Early Detection (ARED)," *IEEE/ACM Trans. Netw.*, vol. 12, pp. 1079-1092, 2004. DOI: 10.1109/TNET.2004.838600

[4] D. Que, Z. Chen, and B. Chen, "An Improvement Algorithm Based on RED and Its Performance Analysis," in *Proc. 9th Int. Conf. Signal Process.*, Beijing, China, 2008.

[5] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397-413, 1993. DOI: 10.1109/90.251892

[6] D. Que, Z. Chen, and B. Chen, "An Improvement Algorithm Based on RED and Its Performance Analysis," in *Proc. ICSP*, 2008.

[7] A. H. Ismail, A. El-Sayed, I. Z. Morsi, and Z. Elsaghir, "Enhanced Random Early Detection (ENRED)," *Int. J. Comput. Appl.*, vol. 92, no. 9, pp. 20-24, 2014. DOI: 10.5120/16039-5015

[8] R. Sharma and G. Dixit, "Experimental Study of RED Performance by Regulating Upper Threshold Parameter," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 5, pp. 6202-6204, 2014.

[9] A. M. Alkharasani, M. Othman, A. Abdul, and K. Y. Lun, "An Improved Quality of Service Performance Using RED's Active Queue Management Flow Control in Classifying Networks," *IEEE Access*, vol. 4, pp. 1-12, 2016.

[10] H. P. Uguta and L. N. Onyejegbu, "An Intelligent Fuzzy Logic System for Network Congestion Control," *Circ. Comput. Sci.*, vol. 2, no. 11, pp. 23-30, Dec. 2017. DOI: 10.22632/ccs-2017-252-69

TÓM TẮT

NÂNG CAO HIỆU NĂNG MẠNG VỚI PHÁT HIỆN SỚM NGẪU NHIÊN CẢI TIẾN

Vũ Văn Diện

Trường Đại học Công nghệ thông tin và truyền thông, Đại học Thái Nguyên, Việt Nam

Ngày nhận bài 13/9/2024, ngày nhận đăng 18/12/2024

Tắc nghẽn đã trở thành vấn đề quan trọng làm ảnh hưởng đến hiệu năng hệ thống mạng. Các thuật toán quản lý hàng đợi động (AQM - Active queue management) đóng vai trò quan trọng để đảm bảo sự ổn định của mạng. RED (Random Early Detection) là kỹ thuật quản lý hàng đợi động đầu tiên được triển khai để điều khiển tránh tắc nghẽn. RED dựa trên việc so sánh chiều dài trung bình hàng đợi với các ngưỡng để đánh dấu hoặc loại bỏ gói tin. DyRED đã cải tiến RED dựa trên việc thay đổi ngưỡng trên. Trong bài báo này, tác giả đề xuất một thuật toán DyRED cải tiến có tên gọi là ImRED (Improved DyRED) để nâng cao hiệu năng mạng. Qua mô phỏng đánh giá trên bộ mô phỏng NS2, tác giả đã thấy được ImRED cho kết quả tốt hơn DyRED xét về tỉ lệ mất gói và độ trễ hàng đợi trung bình.

Từ khóa: Quản lý hàng đợi động; tắc nghẽn; kích thước hàng đợi trung bình; RED; ngưỡng.