RED-LE: A Revised Algorithm for Active Queue Management

CSE 322 Project

February 27

Bangladesh University of Engineering and Technology
Authored by: Mohammad Tamimul Ehsan, 1805022

Summary

In 1993, the random early detection (RED) algorithm was created, and in the nearly 30 years since, many scientists have proposed enhanced versions. However, the use of a pure linear function to compute packets drop probability has led to the problem of significant delays. To address this issue, researchers suggest using both linear and non-linear (such as, exponential) functions for packet drop probability. We have proposed a revised RED algorithm called a RED-linear exponential (RED-LE) that uses a combination of linear and exponential drop functions. At low and moderate network traffic loads, RED-LE employs the linear drop function, while at high traffic loads, it uses the exponential function to calculate the packet drop probability rate. Experimental results demonstrate that RED-LE is effective in managing congestion and improving network performance for different traffic loads.

Introduction

When the amount of incoming data packets is greater than the network's available resources, it leads to network congestion. This can result in poor quality of service with high delays, loss rates, and low throughput. Routers play a crucial role in controlling network congestion, leading to improved network performance. To manage network congestion, a router can drop packets at an early stage using active queue management (AQM) algorithms. These algorithms, such as random early detection (RED), help prevent the buffer from becoming full and send feedback signals to sources to reduce transmission rates. RED is the most common AQM algorithm and is used as the basis for many new AQM algorithms. When a packet arrives at the router, RED updates the average queue size (avg) to detect congestion.

To perform this work, the current status of the queue is examined. If the router's queue is not empty, average value (avg) is determined with the help of a predefined queue weight ($W_q \in [0, 1]$), the previously calculated average queue size (avg') and the current queue size (q_{cur}) using the following equation:

$$avg = (1 - W_q)avg' + (W_q \times q_{cur}) \tag{1}$$

The probability drop function of original RED depends on the avg in the following manner:

$$P_b = \begin{cases} 0, & if\ avg \in [0, minTH) \\ maxP\left(\dfrac{avg - minTH}{maxTH - minTH}\right), & if\ avg \in [minTH, maxTH) \\ 1, & if\ avg \geq maxTH \end{cases} \tag{2}$$

Here, minTH is the router's minimum queue threshold, maxTH is the router's maximum threshold, maxP is the maximum packet drop probability and P_b is the initial packet dropping probability.

The final packet drop probability P_a is as follows:

$$P_a = \frac{P_b}{1 - count \times P_b} \tag{3}$$

Here, count is the number of packets that arrived since the last dropped packet.

RED-LE

The proposed algorithm involves an interplay of linear and exponential drop functions in an aim to improve the performance of original RED algorithm. The probability drop function used in this report is given below:

$$P_b = \begin{cases} 0, & if\ avg \in [0, minTH) \\ 2maxP\left(\frac{avg - minTH}{maxTH - minTH}\right), & if\ avg \in [minTH, target) \\ maxP^{2\left(\frac{maxTH-avg}{maxTH-minTH}\right)}, & if\ avg \in [Target, maxTH) \\ 1, & if\ avg \geq maxTH \end{cases} \tag{4}$$

In which

$$Target = \frac{maxTH + minTH}{2} \tag{5}$$

It can be mentioned that Target distinguishes between two traffic scenarios: lower and moderate buffer occupancies, and higher buffer occupancies.

Simulations

In this section, the proposed RED-LE AQM algorithm is implemented using the ns-2 simulator. The effectiveness of this algorithm is evaluated and compared against original RED and between two tcp variants- Tahoe and Newreno. Two types of topologies are used, namely, a wired dumbbell topology and a wireless grid topology.

Wired Topology

The RED with P_b defined by Eq. (4) was applied to the router R1 while Tail-Drop, that is the most basic control algorithm for the buffers, was implemented in the router R2. The parameter values in our simulations are summarized in Table 1 and Table 2. The network traffic was simulated for 40 s. The topology is shown in Figure 1.

Network Parameter	Value
Access Link Capacity [Mbps]	100
Bottleneck Link Capacity [Mbps]	10
Propagation Delay (Access Link) [sec]	5

Propagation Delay (Bottleneck Link) [sec]	100
Maximum window size rwnd [packet]	100-500
Number of TCP connections N	10-50
R1 queue length [packet]	100

Table 1 : Network Parameters

RED Parameter	Value
Minimum Threshold, minTH [packet]	30
Maximum Threshold, maxTH [packet]	90
Queue Weight, W_q	1
Maximum Packet Drop Probability, maxP	0.1
Mean Packet size [bits]	1000

Table 2: RED parameters

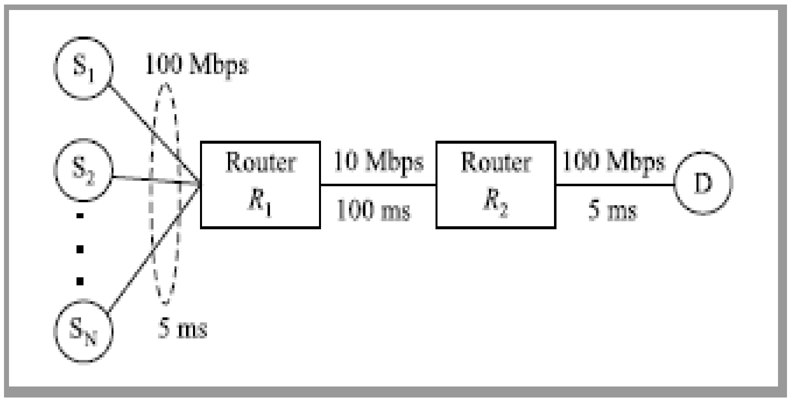


Figure 1: Network Topology

Wireless Topology

For this part we have used a grid topology with 802.11 wireless standard. The nodes are static and there is a flow from one node to another node counted two away, such as, there is a flow from node 0 to node 2. As such, we have got deterministic result in each run and there is no randomness in the topology. The parameter values in our simulations are summarized in Table 3, Table 4 and Table 5. The network traffic was simulated for 40 s. The topology is shown in Figure 2.

Network Parameter	Value
Channel Type	WirelessChannel
Radio-propagation model	TwoRayGround
Antenna type	OmniAntenna
Link Layer Type	LL
Interface Queue Type	RED

Interface Queue Length [packet]	50
MAC type	802.11
Adhoc Routing Protocol	DSDV

Table 3: Network Parameter (Wireless)

Energy Parameter	Value
Energy Model Type	EnergyModel
Initial Energy [J]	1000
Idle Power [W]	1.0
rxPower [W]	1.0
txPower [W]	1.0
Sleep Power [W]	0.001
Transition Power [W]	0.2
Transition Time [sec]	0.005

Table 4: Energy Parameter (wireless)

RED Parameter	Value
Minimum Threshold, minTH [packet]	20
Maximum Threshold, maxTH [packet]	40
Queue Weight, W_q	1
Maximum Packet Drop Probability, maxP	0.1
Mean Packet size [bits]	1000

Table 5: RED parameters (wireless)

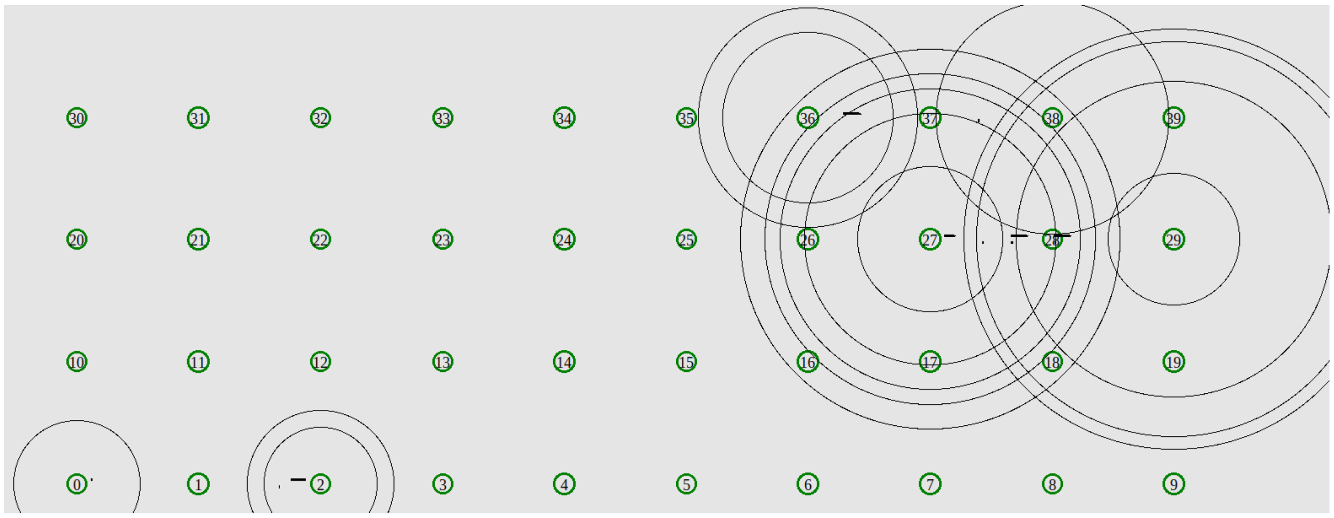
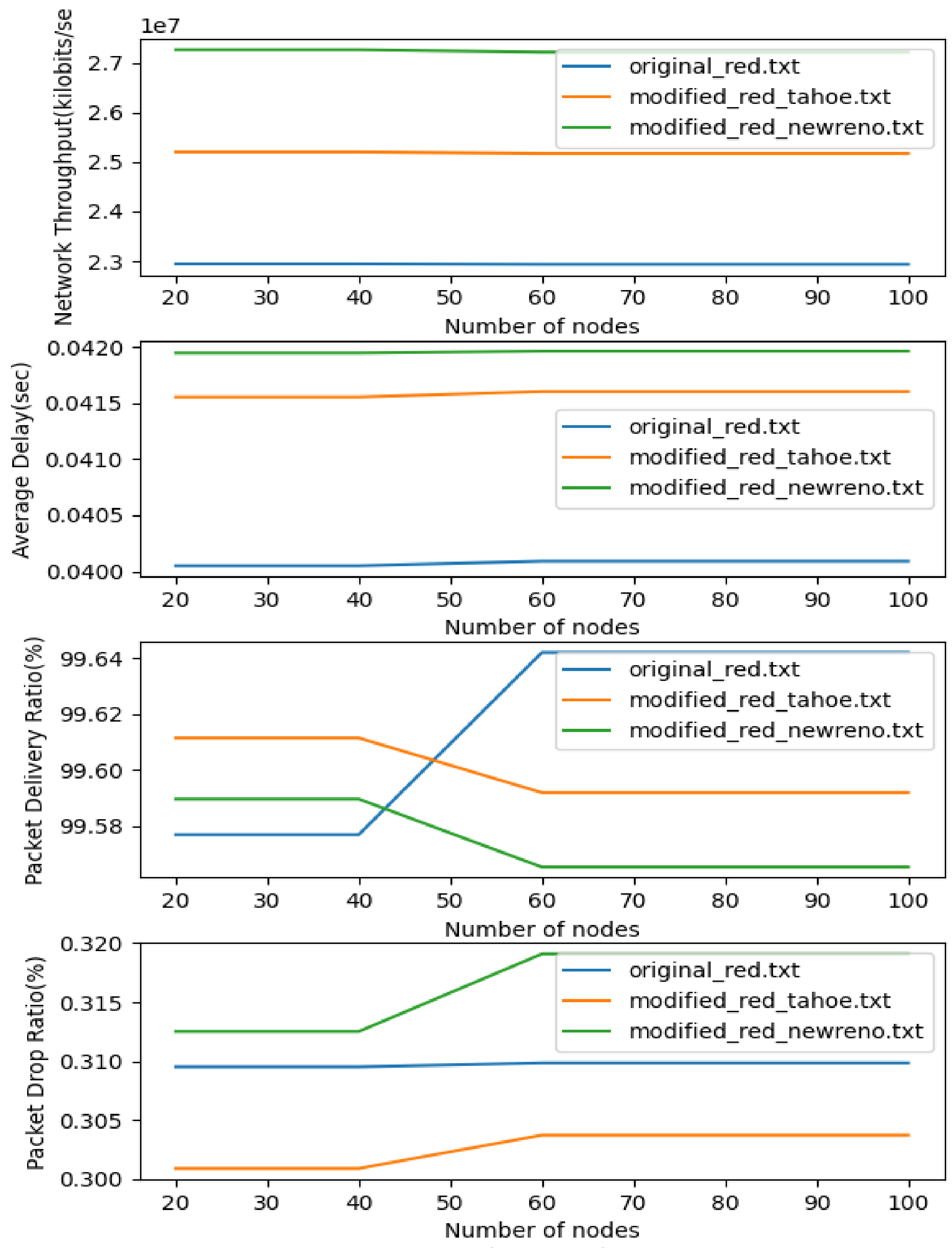


Figure 2: Network Topology (Wireless)

Simulation Results

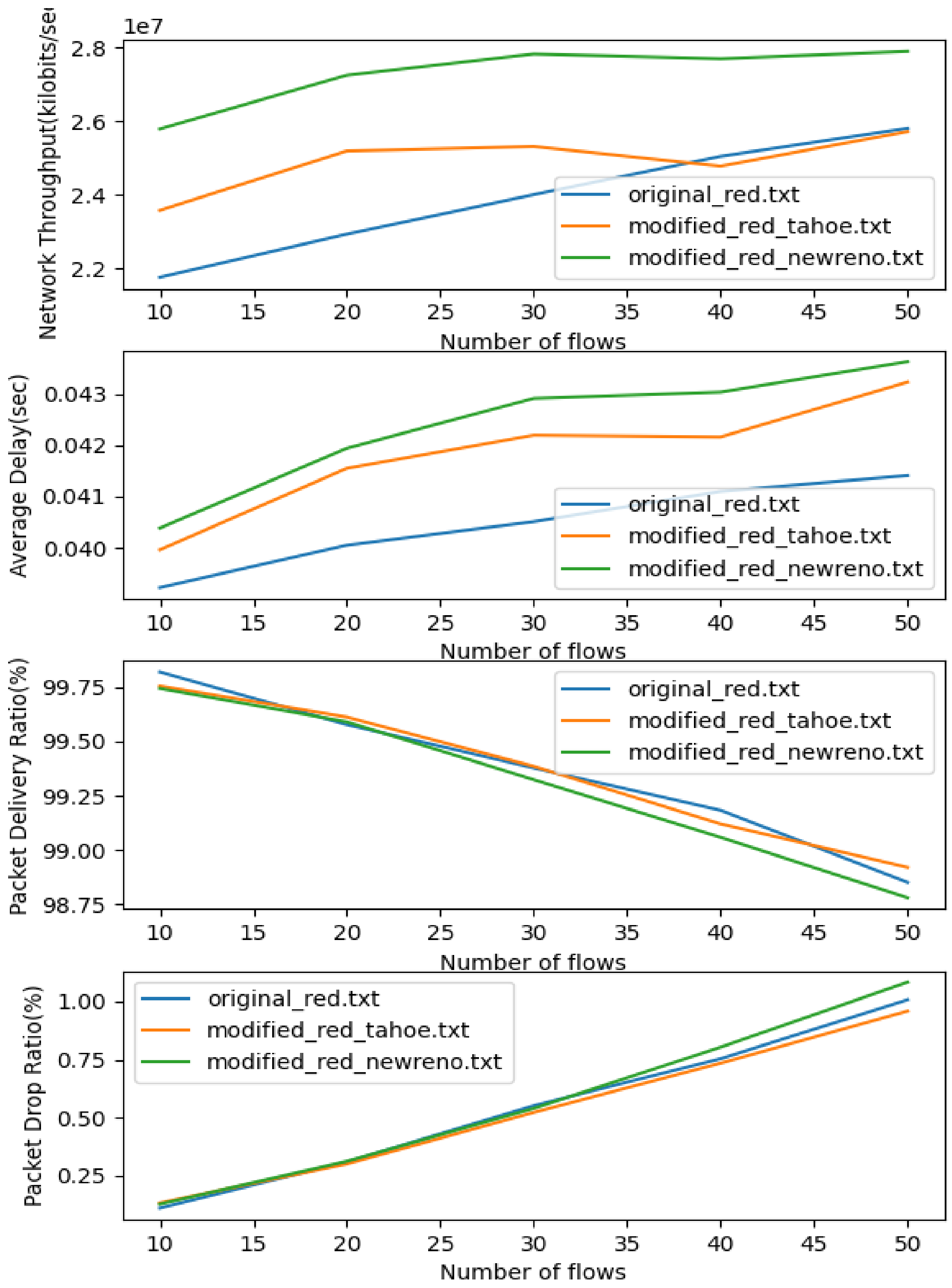
Wired Topology

The number of total nodes is varied keeping the number of flows and packet rate fixed at 20 and 100 respectively. The following comparative graph is obtained. The data are added to with the project.



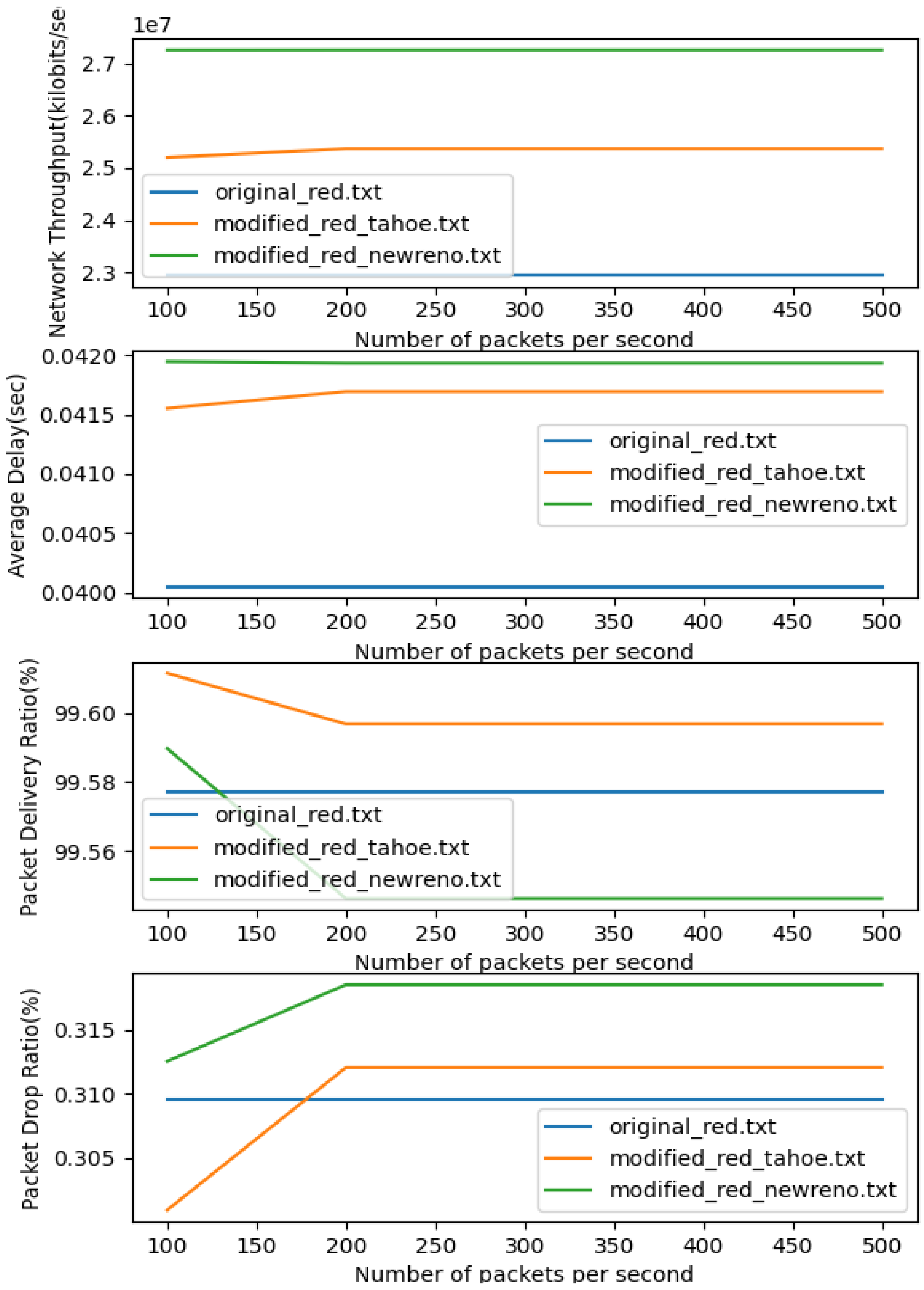
Graph 1:Varying Number of Nodes for wired topology

After that, the number of flows is varied keeping the number of nodes and packet rate fixed at 40 and 100 respectively. The following comparative graph is obtained. The data are added to with the project.



Graph 2::Varying Number of Flows for wired topology

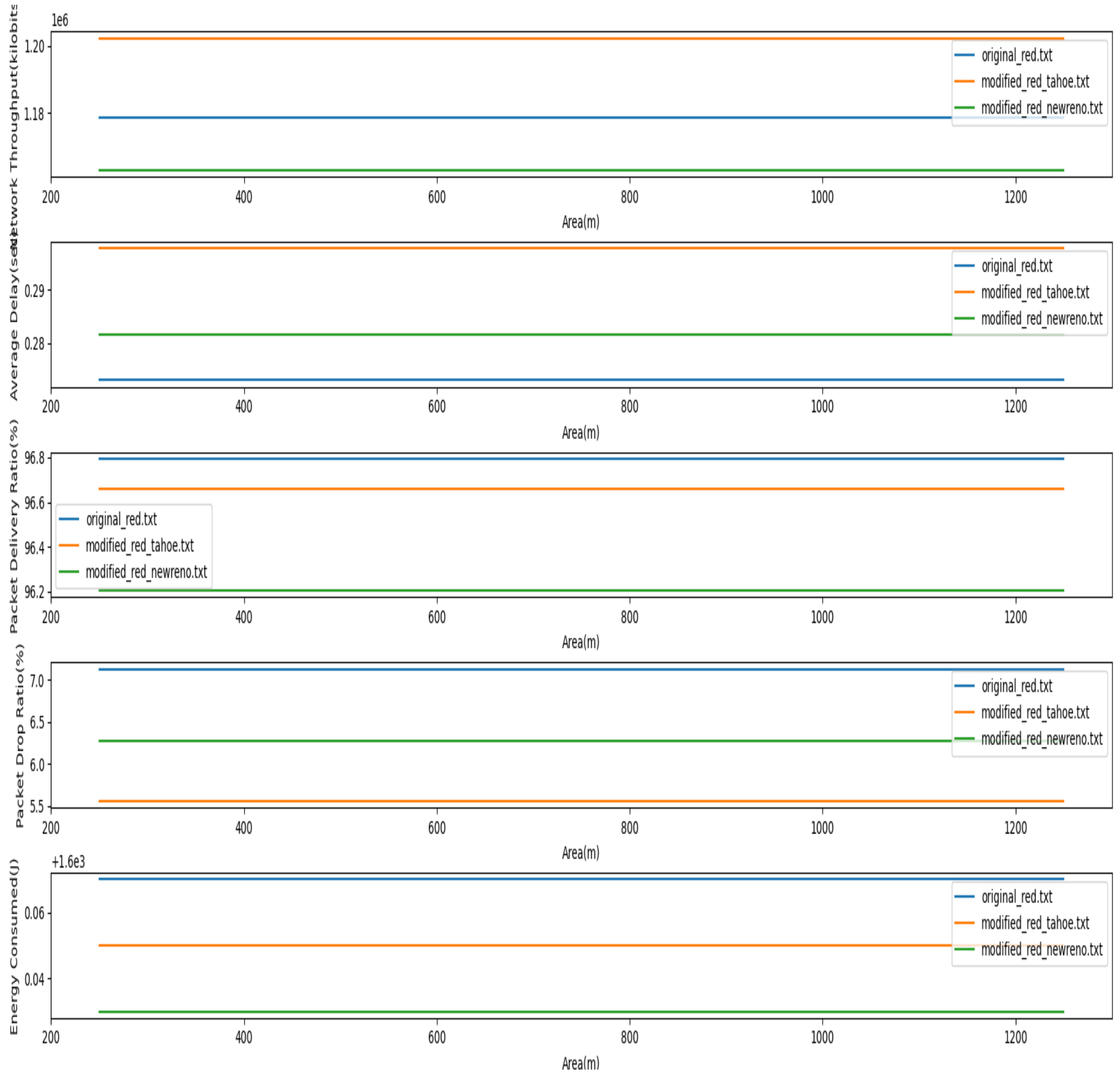
Lastly, the number of packets per second is varied keeping the number of nodes and number of flows fixed at 40 and 20 respectively. The following comparative graph is obtained. The data are added to with the project.



Graph 3:Varying Number of Packets per second for wired topology

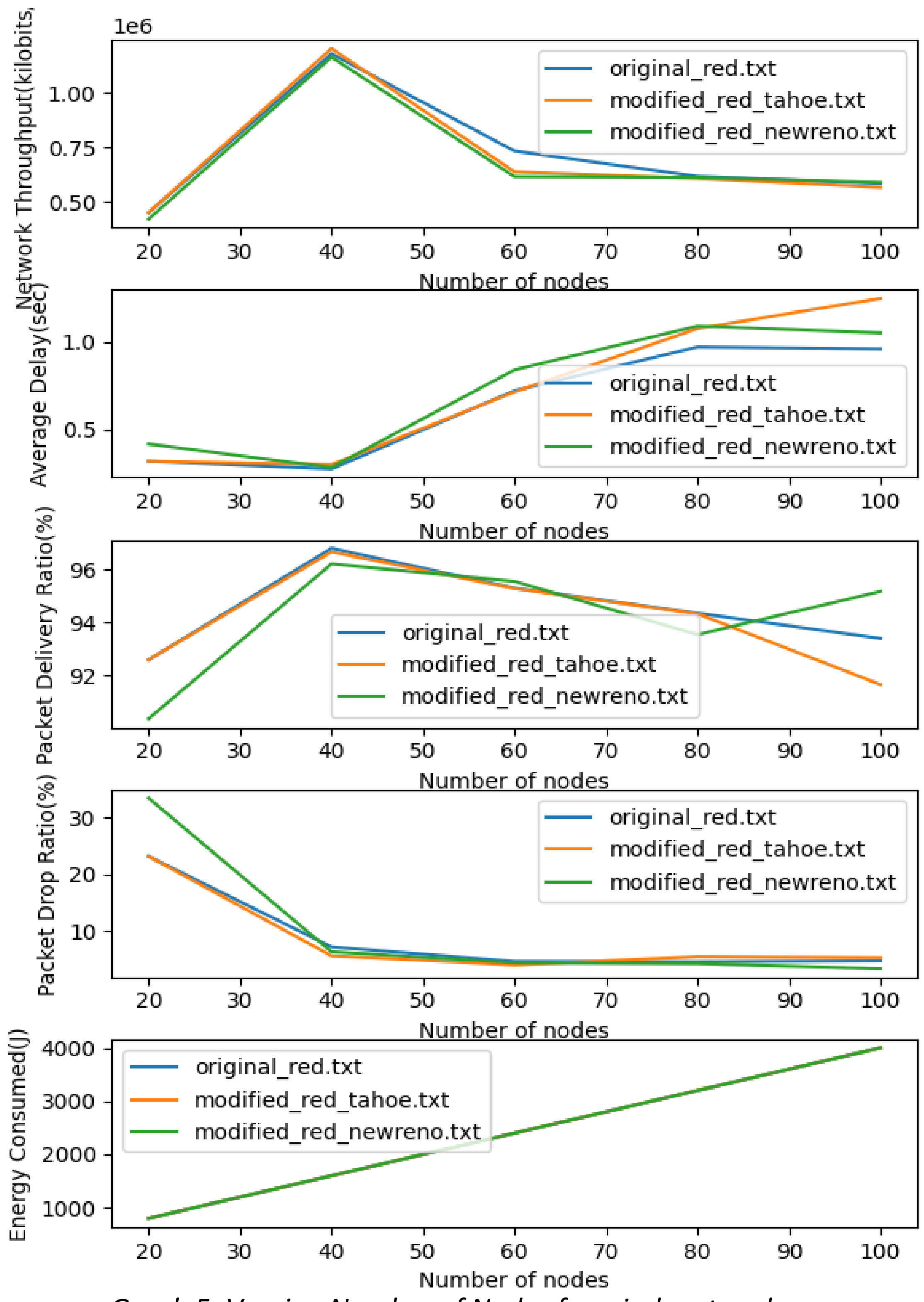
Wireless Topology

The area is varied keeping the number of nodes, number of flows and packet rate fixed at 40, 20 and 100 respectively. The following comparative graph is obtained. The data are added with the project.



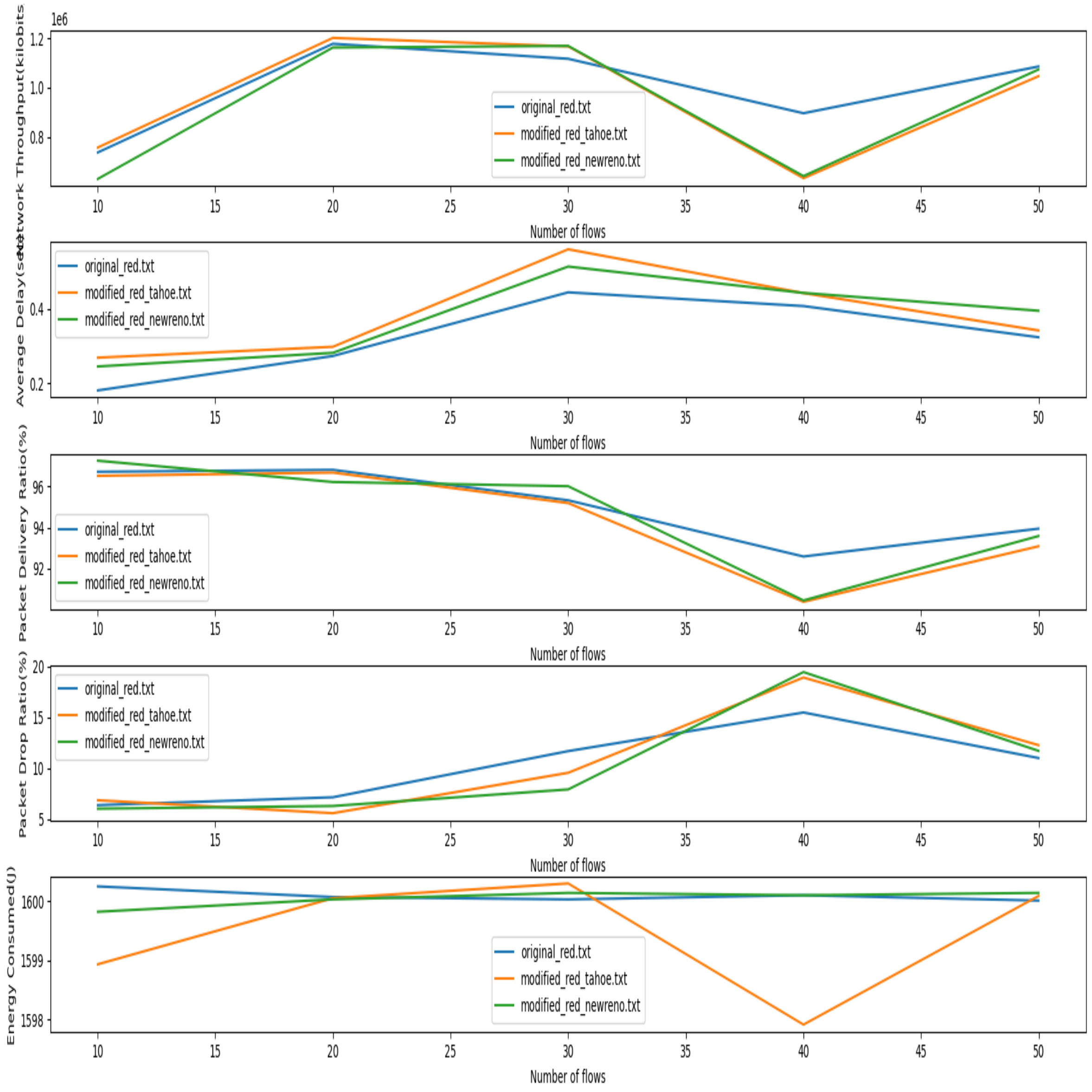
Graph 4: Varying Area Size for wireless topology

The number of total nodes is varied keeping the area, number of flows and packet rate fixed at 500m*500m, 20 and 100 respectively. The following comparative graph is obtained. The data are added with the project.



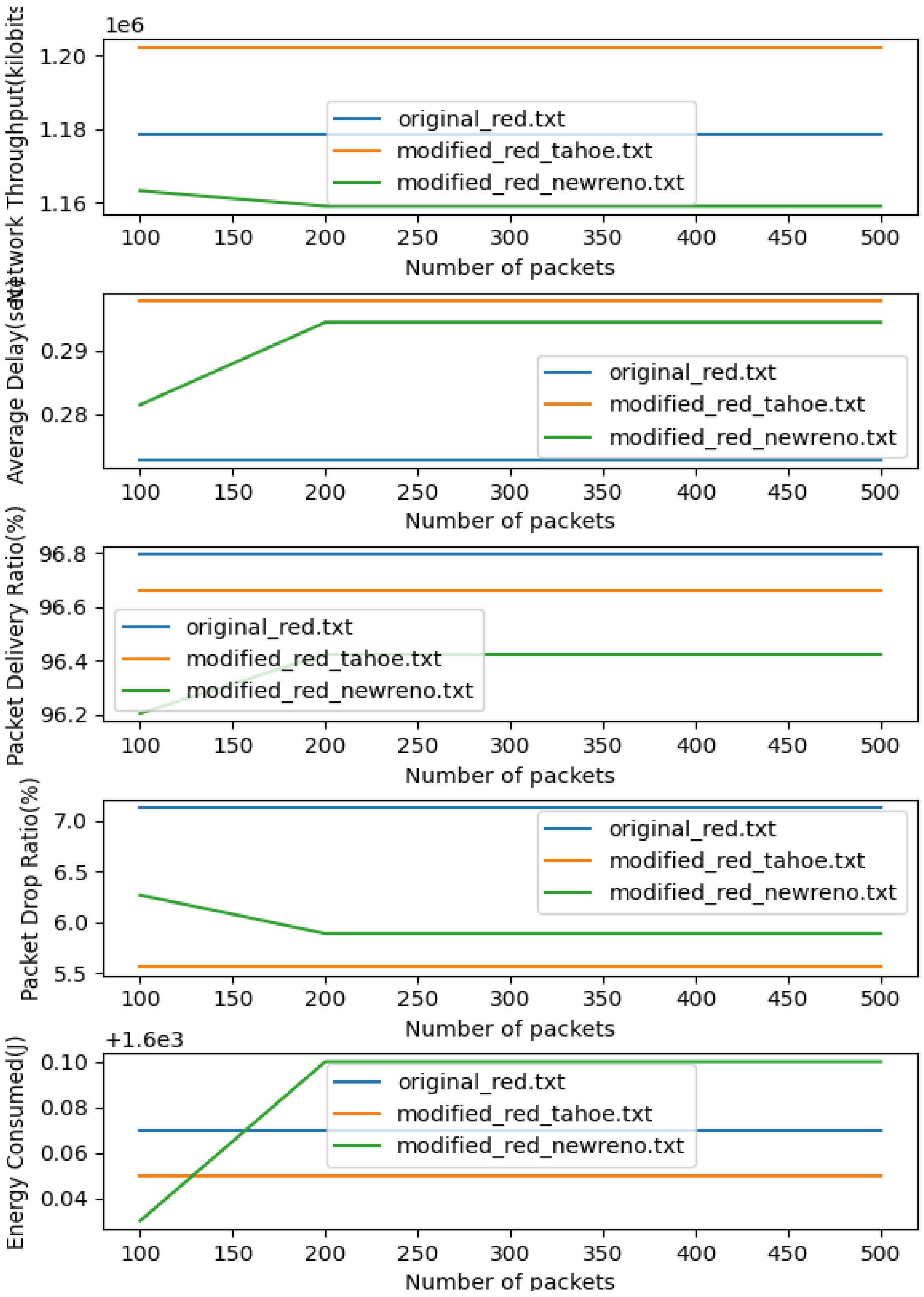
Graph 5: Varying Number of Nodes for wireless topology

After that, the number of flows is varied keeping the area, number of nodes and packet rate fixed at 500m*500m, 40 and 100 respectively. The following comparative graph is obtained. The data are added to with the project.



Graph 6::Varying Number of Flows for wireless topology

Lastly, the number of packets per second is varied keeping the area, number of nodes and number of flows fixed at 500m*500m, 40 and 20 respectively. The following comparative graph is obtained. The data are added to with the project.



Graph 7:Varying Number of Packets per second for wireless topology

Discussion

Wired Topology

Throughput increases for all varieties of parameters for modified RED and especially for TCP Newreno. On the contrary, delay performance goes on the opposite trend that is it does not improve for modified RED, rather the lowest delay is obtained for original RED. In the proposed modification of RED, we increase the probability more than the original RED, as a result, the amount of packet drop increases and thus the packet drop ratio is the highest for modified RED TCP Newreno version. On another note, while varying number of nodes, we see that the modified RED algorithm works best in packet drop ratio for TCP Tahoe (Graph 1). For the case mentioned previously, it is seen that packet delivery ratio is the best for original RED, that means for less congestion, our proposed algorithm does not perform any better than the original version (Graph 1). After that, we see that, when the number of flows is varied, for more flow, more congestion, and in this case, all the algorithms show almost the same performance in packet delivery ratio (Graph 2). Lastly, when we increase the window size, more packets pass and in this case, our proposed algorithm outperforms original RED (Graph 3).

Wireless Topology

We did not get any reference for a wireless topology for a RED queue, so we designed the topology on our own accord. As a result, we have not got very promising result as of yet.

In a wireless topology, TCP Tahoe may have an advantage over TCP Newreno due to its conservative approach, which may lead to fewer packets drops and retransmissions, resulting in higher throughput and lower packet drop ratio.

A larger area may lead to more interference or weaker signal strength, which may cause more packet drops and retransmissions. TCP Tahoe’s conservative approach may help it better to adapt to these changing network conditions, resulting in higher throughput compared to TCP Newreno. Also, with a large congestion window size, TCP Newreno’s aggressive congestion control strategy can lead to more frequent and longer congestion events which can result in more packet drops and retransmissions. In contrast, TCP Tahoe’s more conservative approach can help to avoid congestion and prevent packet loss, leading to higher throughput.

From the graphs we see that, with modified RED, the performance of TCP Tahoe is best, even better than the original RED, in terms of packet drop and throughput while we vary area size and congestion window size(Graph 4)(Graph 7).

On the contrary, varying the number of nodes and number of flows for the topology, we did not get any significant changes for any of the versions.

Conclusion

The study proposed a slight adjustment to the packet drop probability function of the RED algorithm. The proposed solution, call RED-Linear Exponential (RED-LE), is suggested for implementation in routers. The study compares the performance of RED-LE with original RED algorithm and also between TCP Tahoe and TCP Newreno congestion control algorithms. For wired topology and for some variations of the wireless topology, the experimental results show that RED-LE provides more effective congestion control by achieving higher throughputs and lower packet drop ratio, indicating that it might be a promising solution for managing network congestion.

Reference

RED-LE: A Revised Algorithm for Active Queue Management
Journal of Telecommunications and Information Technology, February, 2022
https://bibliotekanauki.pl/articles/2142301.pdf