Hindawi
Journal of Computer Networks and Communications
Volume 2019, Article ID 3645717, 17 pages
https://doi.org/10.1155/2019/3645717

Research Article

Adaptive Receiver-Window Adjustment for Delay Reduction in LTE Networks

N. D. Adesh and A. Renuka

Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India

Correspondence should be addressed to A. Renuka; renuka.prabhu@manipal.edu

Received 27 September 2018; Revised 11 December 2018; Accepted 8 January 2019; Published 26 February 2019

Academic Editor: Sabrina Gaito

The cellular network keeps the vast capacity of queue space at eNodeBs (base stations) to reduce the queue overflow during the burst in data traffic. However, this adversely affects the delay sensitive applications and user quality of experience. Recently, few researchers have focused on reducing the packet delay, but it has a negative impact on the utilization of network resource by the users. Further, it fails to maintain fairness among the users, when competing for a shared resource in coexistence with conventional TCP or UDP users. Therefore, in this paper, the adaptive receiver-window adjustment (ARWA) algorithm is proposed to efficiently utilize the network resources and ensure fairness among the users in resource competitive environment, which requires slight modification of TCP at both the sender and receiver. The proposed mechanism dynamically varies the receiver window size based on the data rate and delay information of the packets, to enhance the performance of the system. Based on extensive experiments, the results illustrate that the ARWA algorithm reduces the delay of TCP packet and increases fairness among the users. In addition to that, it enhances the packet delivery fraction (PDF) and maintains the throughput of the system. Moreover, it competes with other conventional TCP users for the shared network resources in a fair manner.

1. Introduction

In recent years, with the advent of smart phones, there is an increasing demand for good quality of service at low operational cost [1]. This essentiates the system resources to be utilized efficiently. Based on Ericsson 2017 survey statistics [2], the 4G subscribers have grown more than 2.6 billion, and in the next five years, around 5.5 billion additional 4G users are expected. The Cisco report [3] indicates that downloading of data and video streaming have contributed tremendously to the increase in network traffic in 2017, and mobile data traffic will increase by more than 47% by 2021 compared to that of 2017. The rapid increase in wireless data demand and usage has created a significant burden on the existing network resources. Supporting this large increase in data and connectivity is a challenge. This has motivated the industrial personnel and researchers to examine the 4G systems to provide the better solutions to improve the performance of the 4G network.

The LTE radio protocol stack consists of four layers such as Radio Link Control (RLC), Medium Access Control (MAC), Physical layer (PHY), and Packet Data Convergence Protocol (PDCP) [4] to transfer the data securely and reliably between eNodeB and user. Real-time applications have strict requirements on end-to-end packet delay to achieve acceptable quality of experience by the user. Congestion may take place on radio interface due to various reasons, which may result in accumulation of packets in queues of RLC entities and waiting for the instruction from the MAC layer. Thus, there is overflow of RLC queues due to the large volume of traffic in a short period of time leading to high delay resulting in poor performance. To guarantee high throughput and low delay during congestion, the researchers have proposed various methods such as buffer-aware

scheduling [5–8], active queue management (AQM) techniques [9–12], receiver window control [13–15], loss-based congestion control [16–18], delay-based congestion control [16, 19, 20], rate-based congestion control [4, 20, 21], admission and congestion control [22–25], and resource starvation [26–28].

To avoid queue overflow problem, the service providers have installed large capacity buffers (queue) in the eNodeB owing to the reduction in memory price [14]. During congestion in the network, the enormous amount of packets start accumulating in the large queue space at eNodeB and wait for the resources to be allotted [29, 30]. This leads to increase in packet delivery time which adversely affects the delay sensitive applications known as bufferbloat problem [13]. Another problem in the loss-based TCP congestion control method is that it detects the network congestion only after the packet gets lost [17, 31], that is, when congestion window (cwnd) size is large and user queue at eNodeB overflows. To overcome these problems, the delay-based congestion detection approach was introduced in TCP Vegas [20] and TCP Westwood [21]. It uses round-trip time (RTT) to detect the network congestion. However, this approach solves the packet loss and bufferbloat problem, but the network bandwidth is underutilized. Suppose loss-based TCP and delay-based TCP streams are competing with each other for resources, then the delay-based TCP stream suffers from bandwidth starvation due to reduction in its cwnd size based on the delay information [13].

The active queue management (AQM) technique in the router is the promising solution for congestion avoidance, and it is an intelligent way of indicating to the source node (user) to reduce the traffic generation rate either by dropping or marking the packets in router queue before it gets filled up, due to congestion in the network [32]. The queue scheduler uses various algorithms such as random early detection (RED) [33], explicit congestion notification (ECN) [34], and random exponential marking in order to avoid more number of packet losses from the queue. Recently, the router with the timestamp-based AQM method named controlled delay (CoDel) [9] and proportional integral controller enhanced (PIE) [11] has been developed to solve the bufferbloat problem. But, in these methods, packets which exceed the delay budget (target limit) are dropped from the queue. However, it is still unclear that whether these AQM methods are deployed in LTE network.

On the receiver side, the flow control approach is used to monitor and adjust cwnd of the sender to avoid the receiver queue overflow [23]. The receiver sends the information to the sender through advertised receiver window (rwnd). However, recent mobile devices have sufficient queue space and hardly use the flow control method. The receiver side sets the maximum queue size as default rwnd size, and it is commonly called as autotuning [13]. The rwnd can be used for restricting the number of bytes-in-flight in network. Some smartphones have put a limit on byte-flight to avoid the bufferbloat problem. But, this setting is static, and the performance of the device purely depends upon the network condition. The dynamic receiver-window adjustment (DRWA) [14] method solves the bufferbloat problem by changing the rwnd value by keeping the current RTT (packet delay) of packet close to the minimum RTT value. But, when the DRWA method competes with other conventional TCP methods, it suffers from resource starvation and fairness problem due to its conservative behavior.

These issues have inspired us to develop an adaptive receiver-window adjustment (ARWA) to enhance the overall performance of the network. The proposed receiver-side congestion control algorithm aims to achieve three objectives: (1) to reduce the packet latency by considering delay information of the packet; (2) to maintain the fairness among the competing users; and (3) to maximize the utilization of the network resources. To meet the above-mentioned goals, the receiver has to consider the user parameters such as current packet delay, packet size, minimum packet delay, and data rate while calculating the congestion level in the network before informing the sender to reduce its traffic generation rate. The contribution of this paper is as follows:

(i) Experiments have been conducted in LTE network to show the existence of bufferbloat problem in the autotuning method and underutilization of network resources and fairness problem in the DRWA method.

(ii) An innovative technique is developed to detect congestion in the network through the measurement of packet delay and its data rate at the receiver side. The level of network congestion is estimated based on comparison of the current data rate of the user and previously computed user data rate with minimum delay. The receiver intimates the sender to adjust its traffic generation rate via advertising receiver window. This helps the sender to adjust the traffic generation rate moderately which in turn reduces the number of packets waiting in the eNodeB queue and decreases packet delay. It also minimizes the packet loss in the network. At the same time, it prevents the sender from reaching the halt state. It enhances resource utilization and maintains fairness among the users when competing with coexisting conventional users in LTE network.

(iii) The simulation results show that the proposed algorithm improves the performance of the network by decreasing the packet delay and maintains the system throughput and fairness among the users as compared to the existing TCP autotuning method and DRWA method.

The rest of the paper is as follows: Section 2 briefly discusses the related work in this field. In Section 3, the design of the proposed network congestion estimation method at the receiver side and adaptive packet generation rate algorithms are explained. Section 4 presents the experimental setup and performance comparison of the proposed algorithm with existing algorithms. Finally, the research work is concluded in Section 5.

2. Related Work

Lot of research work has been done related to performance issues in congested network and the solution to these problems. However, implementing these methods in the LTE system may not effectively solve the problems, due to the unique characteristics of the cellular networks compared to wired and wireless networks. Recently few research work related to congestion in LTE network has been done. The proportional fair (PF) scheduler is used in [35] to study the performance of TCP variants in LTE wireless network. The purpose of their investigation is to understand the unexpected low performance of TCP protocols over the mobile users in the system. In [27], the impact of TCP congestion window on scheduler performance in LTE network is studied. The paper describes the loophole of schedulers while selecting the users for resource allocation which leads to degradation in system performance. They proposed the TCP-oriented multilayer packet scheduler to enhance the performance of the system.

The active queue management (AQM) technique for queues at eNodeB was proposed in [36], to keep minimal queue level occupancy without degrading the performance of the user applications. However, fine-tuning is required which is not only based on traffic load but also on user application. On the other hand, delay sensitive application problems are solved by delay-based congestion control algorithms. The packet prediction mechanism (PPM) is designed in [9]. The algorithm guesses the next incoming packet time to the queue, based on the existing packets in the user queue at eNodeB. It also predicts the highest delay of the packet in the queue based on the outgoing rate of packets from the queue. The drawback of this mechanism is that it allocates the resource to the packets based on highest delay of the packet. However, it does not consider the fairness and queue overflow while scheduling the eNodeB resources among active users. Also, it is applicable only for guaranteed application traffic and not applicable for nonguaranteed application traffic.

The dynamic receiver-window adjustment (DRWA) method [14] was proposed to solve the bufferbloat problem by dynamically changing the rwnd size at the receiver side based on round-trip time (RTT) of the packet by keeping minimum RTT as the baseline. In [37], the authors proposed receiver-assisted congestion control (RACC), which measures the network bandwidth based on interarrival times of packets and computes the congestion window at the receiver end. After receiving the advertising value from the receiver, the sender adjusts its window size based on feedback information. However, in both the methods, the user with low signal strength suffers from resource starvation and low throughput when competing with other TCP users, due to its conservative behavior.

Im et al. [13] designed a receiver-side TCP adaptive queue control (RTAC), which addresses the bufferbloat problem in WiFi and LTE network. The receiver side estimates the packet delay and amount of in-flight data in the network and controls the transmission rate via advertised window. Here, the queue length of wireless router is assumed which is not practical.

From the extensive literature survey covering various aspects of LTE congestion control, it is presumed that there is a lack in managing and controlling the downlink packet delay problem. To solve these issues, it is necessary to look into dynamic adjustment of the congestion window which controls the packet generation rate at the sender side.

3. Adaptive Receiver-Window Adjustment Algorithm

The aim of the proposed algorithm is to estimate congestion in the network based on packet delay at the receiver side in order to reduce the packet delivery time as well as packet loss in the network. Most of the existing TCP congestion control schemes that estimate the network congestion at the sender side are either loss-based (reactive congestion control) [17] or delay-based (proactive congestion control) [20], and few are hybrid, based on both the methods [30]. The receiver side accomplishes the flow control by sending back receiver window (rwnd) size to the sender. This receiver window size gives an indication to the sender about the maximum amount of data that the receiver can handle. When the receiver is saturated with data packets, it might not be able to send back the signal to the sender to slow down the transmission rate without dropping packets. However, the TCP sliding window protocol at the sender side is designed to determine its sending rate by taking the minimum of congestion window size (cwnd) and receiver window size (rwnd) [38]. By observing this technique, the sender-side transmission rate (tr) in Equation (2) can be controlled by receiver-side window size [14] as shown in Equation (1).

$$\text{ws} = \min(\text{cwnd},\ \text{rwnd}), \tag{1}$$

$$\text{tr} = \frac{\text{ws}}{\text{RTT}}, \tag{2}$$

where ws denotes the window size at the sender side and round-trip time (RTT) indicates the time taken by the packet to travel from the sender to receiver and back to the sender. The RTT calculation at the sender may not be an accurate estimate of the packet transmission time from the sender to receiver. Suppose the packet traveling from the sender to receiver in a network with network congestion, it takes x units of time, and to return an acknowledgment from the receiver to sender without network congestion, it takes y units of time, then the total time taken would be $x + y$ units of time. The time required to travel from the sender to receiver is not half the total time. In order to overcome this problem, the packet delay (D) estimation is done at the receiver side.

The proposed algorithm consists of four phases; measurement of packet delay and packet size, network congestion estimation at receiver side, avoidance of aggressive reduction in congestion window size, and adaptive packet generation process as shown in Figure 1. In the proposed adaptive receiver-window adjustment (ARWA) method, the sender uses the loss-based mechanism (NewReno) and the receiver performs the delay-based congestion control. The receiver side (user equipment) accepts the incoming packets from the server or remote user via eNodeB and extracts the

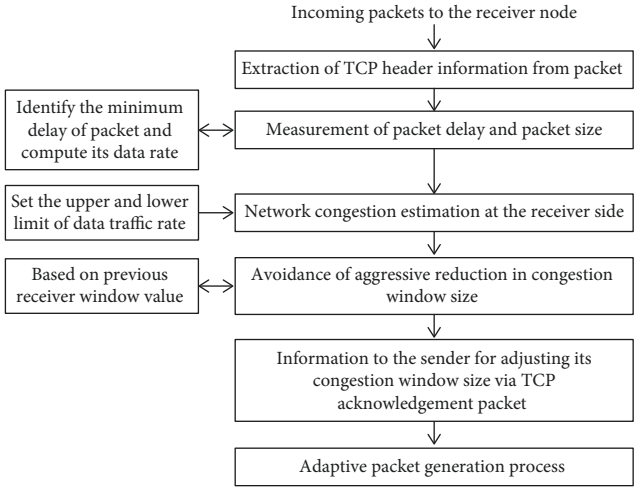

Figure 1: Overall proposed methodology.

TCP header information from packets for further estimation process.

3.1. Measurement of Packet Delay and Packet Size. The proposed method estimates the network congestion from the packet delay (D) information. The receiver measures the delay of packet by using the timestamp option in the TCP header of the incoming packet. The packet delay is measured by taking the difference between the current time and value of the timestamp (refer Algorithm 1 line number 9). Even though the timestamp option is used, the synchronization of the clocks between the senders and the receivers is not required because the delay of the packet is measured relatively at the receiver in a particular session. When the receiver receives the packet with delay which is less compared to the delay of the packets received in the recent past (in that particular session), that delay is called as minimum delay, and is denoted by $D_{\min}$ [13]. When the incoming packets arrive at the receiver in a particular session, the receiver continuously measures the delay of the packets and compares with the minimum delay. If it is less, it considers that packet delay as minimum delay. The minimum packet delay is considered as the delay of the packet traveled from the sender to receiver along the network without experiencing queuing delay. The queuing delay of the packet along the path is calculated as the difference between the current packet delay and the minimum packet delay ($D_{\min}$). After knowing minimum delay of the packet, the algorithm calculates the data rate ($\text{datarate}_{\max}$) of the receiver by computing the ratio of the packet size of minimum delay packet (D_{pktsize}) to minimum packet delay ($D_{\min}$) (refer Algorithm 1 line number 13). However, the estimation of minimum packet delay may suffer from inaccurate measurement as in delay-based congestion estimation due to the change in quality of the signal received by the user from eNodeB. Suppose the user moves from the tower (base station) location with good signal strength to poor signal strength location, then the packet delivery time (packet delay) increases as shown in Section 4.3.3. But, the minimum delay of packet remains the same, which affects the estimation of receiver window size and leads to underutilization of network resource. However, most of the time, the session time is small compared to the moving timescale. In future work, the accelerometer module of the user device can be incorporated with the algorithm to reset the minimum delay of packet for fast movement [14].

3.2. Network Congestion Estimation at Receiver Side. The algorithm estimates the receiver window size either aggressively or conservatively depending upon the traffic load. If the network is congested, the receiver window (rwnd) value ranges from 1 to 100. If the network is underutilized, the rwnd value ranges from 101 to 120. Otherwise, the rwnd value is set to default (maximum) rwnd size. To reduce the computation complexity, the estimation of rwnd is done either periodically every thirty milliseconds (targetperiod) or when the countervariable (total number of received packets) is greater than the maximum limit of counter (targetcounter) as shown in Algorithm 1 line number 21 and the targetcounter value is set to fifteen. Otherwise, it sends the maximum rwnd size. The countervariable is used to estimate the correct traffic level in the network when the receiver side receives the sudden chunk of data packets with different packet delay within the target period (targetperiod). The algorithm uses the amount of data received (totaldata) by the receiver in terms of bytes and the time taken by the packets to reach the receiver (totaldelay) in milliseconds for a period of time as shown in Algorithm 1 line numbers 15 and 16 and computes the data rate (datarate) based on these values. The current percentage (currentpercentage) of the receiver data rate is computed by comparing the current data rate (datarate) to the data rate without congestion ($\text{datarate}_{\max}$) (line 3.1). The difference in percentage of the data rate without congestion and current percentage (currentpercentage) indicates the level of congestion in the network (Diffpercentage). If this percentage value is less, it means that the amount of data traffic in the network is less and is close to that without any congestion. Otherwise, the data traffic is high, and some measures have to be taken to reduce the data traffic generation rate at the sender side. The receiver side estimates the rwnd size based on the difference in percentage of data rate as follows:

Case 1: If the difference in percentage of data rate (Diffpercentage) is greater than or equal to the upper threshold limit ($\text{limit}_{\text{high}}$), it indicates that the traffic level in the network is high and there is a need to inform the sender to reduce the traffic generation rate. Here, the upper threshold limit is set to ninety, and lower threshold limit ($\text{limit}_{\text{low}}$) is set to thirty [13]. The rwnd value is computed based on difference between Diffpercentage and lower threshold limit ($\text{limit}_{\text{low}}$) as shown in Algorithm 1 line number 26. This rwnd value will aggressively decrease the cwnd at the sender side and result in sudden decrease in the packet generation rate at the sender side which helps in reducing the packet loss as well as packet delivery time.

```
(1)  procedure RECEIVER WINDOW (parameters)
(2)    INITIALIZE (variables)
(3)    rwnd = default receiver window size
(4)    D_min = ∞
(5)    D = 0, totaldelay = 0, totaldata = 0, counter = 0, targetcounter = 15
(6)    targetperiod = 30 ms, limit_low = 30, limit_high = 90, lowest = 20
(7)    Measure the packet delay and packet size:
(8)    D = current time − packet sent time
(9)    if D < D_min then
(10)     D_min = D
(11)     D_pktsize = packet size
(12)     datarate_max = D_pktsize/D_min
(13)   end if
(14)   totaldelay = totaldelay + D
(15)   totaldata = totaldata + packet size
(16)   counter = counter + 1
(17)   rwnd = default receiver window size
(18)   Estimation of congestion in network:
(19)   if ((current_time − last_time) > targetperiod) or (counter > targetcounter) then
(20)     datarate = totaldata/totaldelay
(21)     currentpercentage = (datarate/datarate_max) × 100%
(22)     Diffpercentage = 100% − currentpercentage
(23)     if Diffpercentage ≥ limit_high then
(24)       rwnd = (Diffpercentage − limit_low)
(25)     else if (Diffpercentage ≥ limit_low) then
(26)       rwnd = (Diffpercentage − limit_low) × (Diffpercentage − limit_low/limit_high − limit_low)
(27)     else if (Diffpercentage ≤ lowest) then
(28)       rwnd = (lowest − Diffpercentage) + 100
(29)     end if
(30)     Avoiding aggressive reduction in congestion window size
(31)     if ((current_time − last_time) < targetperiod) and (rwnd < 100) then
(32)       if rwnd > oldrwnd then
(33)         rwnd = rwnd − oldrwnd
(34)         oldrwnd = rwnd
(35)       else
(36)         oldrwnd = rwnd
(37)       end if
(38)     else
(39)       oldrwnd = 0
(40)     end if
(41)     last_time = current_time
(42)     totaldata = totaldelay = counter = 0
(43)   end if
(44) end procedure
```

Algorithm 1: Network congestion estimation at the receiver side.

Case 2: If the difference in percentage of data rate is greater than or equal to the lower threshold limit, the traffic level is little above the normal traffic load. The rwnd value is computed based on the difference in percentage of data rate, lower threshold limit, and upper threshold limit value as shown in Algorithm 1 line number 28. This will conservatively decrease the cwnd at the sender side and result in decrease of the packet generation rate at the sender side which helps in reducing the packet delivery time.

Case 3: If the difference in percentage of data rate is less than 30% of normal traffic load, it indicates that the traffic level is normal and there is no need to decrease the cwnd value. The rwnd value is set to the default maximum receiver window size in order to allow the cwnd to increase the traffic rate for better utilization of network bandwidth. If the difference in percentage of data rate is less than the lowest (lowest) value of the normal traffic load, this indicates that the traffic level is very less in the network. The lowest (lowest) value is set to twenty. In order to boost the traffic level in the network, the receiver has to intimate the sender to immediately increase the data traffic generation rate in the network based on the rwnd value. The rwnd value is in the range of 101 to 120, and once the sender receives the rwnd value, it does further computation to increase

the data traffic in the network based on the rwnd value (refer Algorithm 2 line number 13). The value of rwnd above 100 is used to differentiate between congestion intimation message and underutilization of network resource intimation message.

The primary intention of setting the rwnd value is to maintain the network queue length as small as possible without underutilization of the bandwidth. The lowest limit, lower threshold limit, and upper threshold limit values play a major role in setting the correct rwnd value. If the difference in percentage of data rate value is directly taken as the rwnd value, then the sender window is unnecessarily reduced which leads to less traffic load and underutilization of bandwidth capacity. Therefore, to achieve better bandwidth utilization, the transmission rate is maintained between one bandwidth delay product (BDP) and 2 ∗ BDP [13]. The rwnd estimation process helps the sender to vary the cwnd size which resembles a sawtooth pattern.

3.3. Avoidance of Aggressive Reduction in Congestion Window Size. Once the rwnd value is estimated, additional conditions are checked on the rwnd value at the receiver side in order to avoid the aggressive reduction in congestion window at the sender side. It checks the following two conditions: (i) whether the rwnd estimation time interval is less than the target period (targetperiod) and (ii) whether the rwnd value is less than 100 (refer Algorithm 1 line number 34). If any one of the conditions is found to be false, the old rwnd value (oldrwnd) is set to zero and the estimated rwnd value is sent to the sender for further processing. Otherwise, if both the conditions are true, then whether the rwnd value is greater than the previous iteration rwnd value (oldrwnd) is checked and if found true, the rwnd value is deducted from the old rwnd value, and the new rwnd value is assigned to the old rwnd variable (refer Algorithm 1 line number 37) and the value is sent to the sender. If it is false, then the rwnd value is assigned to the old rwnd variable and the estimated rwnd value is sent to the sender. The old rwnd value helps in decreasing the rwnd value, if the next iteration condition checking occurs within the target period which in turn avoids the aggressive reduction in congestion window size.

3.4. Adaptive Packet Generation Process. When the sender side receives an acknowledgment packet from the receiver side, it extracts the TCP header and obtains the advertised receiver window (rwnd) size. The algorithm checks the following two conditions: (i) whether rwnd value is less than the congestion window (cwnd) at the sender and (ii) whether the rwnd value is less than 100 (refer Algorithm 2 line number 7). If both the conditions are true, it means that the traffic level is high in the network and there is a need to reduce the traffic generation rate. The sender uses the rwnd information to reduce cwnd. After reducing the cwnd size, one more condition is checked to maintain the minimum cwnd size ($\text{targetlimit}_{\text{low}}$). If this condition is not checked, then the sender goes on reducing the window size and may reach a value below the minimum window size which brings the sender traffic generation to halt state. In order to avoid that, a minimum cwnd size ($\text{targetlimit}_{\text{low}}$) is set to 3000 bytes based on the experiment conducted and is discussed in Section 4.3.4. The computed cwnd size is checked to find out whether it is less than the minimum cwnd size. If it is true, the minimum cwnd size is assigned to cwnd. The minimum cwnd size helps the user in avoiding resource starvation and fairness problem when competing with other flow schemes in LTE network as shown in Section 4.3.7.

```
(1)  procedure SENDER WINDOW (parameters)
(2)    INITIALIZE (variables)
(3)    cwnd = Measure the current cwnd size
(4)    rwnd = received ack with advertised window size
(5)    targetlimit_high = 10000, targetlimit_low = 3000
(6)    if rwnd ≤ 100 and rwnd < cwnd then
(7)      rwnd = (rwnd/100)
(8)      cwnd = cwnd − (cwnd × rwnd)
(9)      if cwnd < targetlimit_low then
(10)       cwnd = targetlimit_low
(11)     end if
(12)   else if rwnd ≤ 120 and rwnd < cwnd then
(13)     if cwnd > targetlimit_high then
(14)       rwnd = (rwnd − 100)/200
(15)     else
(16)       rwnd = (rwnd − 100)/100
(17)     end if
(18)     cwnd = cwnd + (cwnd × rwnd)
(19)   else
(20)     cwnd = min(rwnd, cwnd)
(21)   end if
(22) end procedure
```

Algorithm 2: Adaptive congestion control.

If the above-mentioned conditions are false, then the following two additional conditions are checked: (i) whether the rwnd value is less than 120 and (ii) whether the rwnd value is less than the congestion window (refer Algorithm 2 line number 13). If both the conditions are true, it indicates that the bandwidth of the network is underutilized, so the cwnd value of sender has to be increased to generate more data traffic to maintain bandwidth delay product (BDP) between one BDP and 2 ∗ BDP. Before increasing the cwnd value based on the rwnd value, the current cwnd value is compared with the upper target limit ($\text{targetlimit}_{\text{high}}$) in order to avoid a massive increase in cwnd size which generates more data traffic in the network [39]. The upper target limit ($\text{targetlimit}_{\text{high}}$) is set to 10000 bytes, based on the experiment conducted and is discussed in Section 4.3.4. If it is found that the cwnd value is greater than the upper target limit ($\text{targetlimit}_{\text{high}}$), then the cwnd value is computed based on half the rwnd value (refer Algorithm 2 line number 15). Otherwise, the cwnd value is computed based on the rwnd value. If none of the above conditions are satisfied, then it assigns a value which is the minimum of rwnd and cwnd. Thus, the sender either decreases or increases the traffic generation rate to maintain moderate traffic in the network.

4. Simulation Setup and Performance Analysis

In this section, the performance of the proposed work is compared with the DRWA and autotuning algorithms. The proposed algorithm is implemented using the LTE module of ns-3.26 simulator [40]. The LENA module is used to create an end-to-end LTE network which mimics a real-time system. The topology setup is described in Section 4.1. The simulation scenario setup that is used for the experiments and the results of the experiments are explained in Section 4.3. The outputs of the proposed algorithm in different LTE experiments are evaluated using key performance indicators such as average throughput, average delay, average packet delivery fraction (PDF), and fairness index.

4.1. Configuration of Topology. The experimental setup comprises one base station with three sectors, and the users are distributed uniformly across the sector in eNodeB coverage area such that the users receive various signal-to-interference ratios as shown in Figure 2. The transmission power of eNodeB is set to 43 dBm and noise to 5 dBm. The users are attached to nearby eNodeB sector through an air interface. The gateway node of LTE network is connected to the remote server via a 100 Gbps bandwidth capacity link. The downlink data traffic that is generated at the remote host leaves the evolved packet core of LTE network to reach the destination through eNodeB. The configuration parameters used in LTE simulation are listed in Table 1.

4.2. Performance Parameters. In the experiments, several performance metrics are used for performance evaluation. These metrics are presented below:

Average throughput (AT): It measures how much amount of data is received at the receiver successfully over a period of time. The average throughput is measured as bit/s (bps):

$$\text{AT} = \frac{(\text{number of packets received} * \text{packet size} * 8)}{\text{total simulation time}}. \tag{3}$$

Fairness index (FI): It is used to determine whether the users are receiving the fair amount of network resources or not, to meet the QoS requirements. The expression for computing Jain's fairness index is given below:

$$\text{FI} = \frac{(\text{total received bytes of all users} * \text{POW}(2))}{(\text{number of users} * \text{sum of}(\text{total received bytes of individual user} * \text{POW}(2)))}. \tag{4}$$

Average delay: The sum of time taken by the packets to travel across the network from one end to another end divided by the total number of packets received at the receiver. The packet delay includes all possible delays such as queuing delay, transmission delays, processing delay, and propagation delay. It is measured in terms of milliseconds (ms):

$$\text{Average delay} = \frac{\text{sum of delay of all packets}}{\text{number of packets received}}. \tag{5}$$

Average packet delivery fraction (Average PDF): It is the ratio of the data packets delivered to the destinations to those generated by the sources. It is measured in terms of percentage (%):

$$\text{Average PDF} = \frac{(\text{number of packets delivered} * 100)}{\text{number of packets generated}}. \tag{6}$$

4.3. Simulation Scenarios and Experimental Results. The performance of the LTE network is examined under different simulation scenario settings which include understanding the influence of flat and fat TCP, impact of queue size variation at eNodeB, variation of CQI (channel quality indicator) value, impact of target limit on congestion window, variation in the number of users, estimation of network congestion at the receiver side, coexistence with different flows, and variation in congestion window with time. The simulation setup parameters used to conduct experiments to derive the tuning parameters are representative of a typical network behavior.

4.3.1. Understanding the Influence of Flat and Fat TCP. This experiment is conducted to examine the effect of change in maximum size of TCP advertising window (tcp_rmem_max) on the performance of LTE network [14]. In this scenario, the eNodeB consists of only one user with TCP application running on it and the user receives good signal strength from eNodeB. The user is in stationary position and the user queue size at eNodeB is fixed at 2 gigabyte. The remaining simulation parameters and settings are the same as given in Section 4.1.

In this experiment, two types of TCP congestion window patterns are observed, i.e., flat TCP and fat TCP [14]. The flat TCP is that where TCP congestion window grows linearly up to a particular receiver window size and stays there for a long time until the session is closed. This behavior can be observed for a receiver window (tcp_rmem_max) of size 32768 bytes as shown in Figure 3(a). The eNodeB has an ample quantity of resources to provide service to the sender (user), and the sender does not experience any packet loss due to the large queue space available to accumulate the excess packets, but the flat TCP cwnd value tcp_rmem_max affects the throughput of the system. The throughput of the system increases with the increase in the tcp_rmem_max value to a certain value and remains constant due to the saturation state of the link. Further

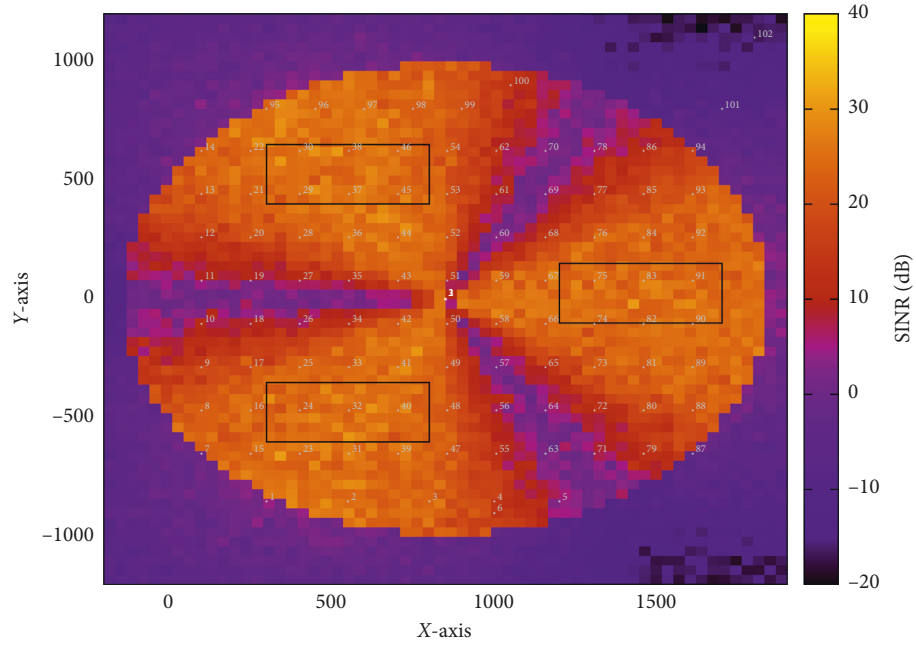

Figure 2: 4G network coverage area.

Table 1: Configuration parameters used in simulation [5].

Parameter	Value
Simulation time	100 seconds
Transmission bandwidth	10 MHz (50 PRBs)
Number of eNodeBs	1 macrocell site
PathlossModel	HybridBuildingsPropagationLossModel
eNodeB antenna	Parabolic antenna model
Total no. of users	20, 40, 60, 80, 100
Queue size	10240, 20480, 30720, 40960, 51200 bytes
Application traffic	Video, FTP
Transport layer protocol	TCP NewReno, UDP
Packet size	FTP = 512 bytes, video = 375 bytes(H.264 of 750 Kbps)
Downlink packet scheduler	Priority set scheduler

increasing the tcp_rmem_max value above the bandwidth capacity of system does not improve the system throughput or user throughput (Figure 3(a)), but it adversely affects the packet queuing delay as shown in Figure 3(b). The main reason for bufferbloat problem is the excessive packet generation rate compared to the capacity of wireless link. The excess packets generated get piled up in eNodeB queue, and the queue overflow probability is less due to its vast capacity. However, the queuing delay is increased due to the waiting time of the packets in the queue to get the eNodeB resource. In fat TCP, the congestion window value goes on increasing until the packet loss occurs in the network. The overgrowth of congestion window allows the sender to generate more packets beyond the network capacity and excess packets get piled up in queue space which leads to queue overflow.

From the above experiments, it is observed that fixing the maximum value of advertising window (tcp_rmem_max) affects the performance of the system. Fixing the appropriate tcp_rmem_max value is also difficult because it depends upon other parameters such as background traffic of other users in eNodeB, eNodeB capacity, and queue size of user at eNodeB. So, the adaptive receiver-advertising window adjustment is required. In the next four sections, the different scenarios are discussed to justify the selection of parameters.

4.3.2. Impact of Queue Size Variation at eNodeB. The impact of queue size variation at eNodeB is investigated by varying the queue size from 10240 bytes to 51200 bytes. The eNodeB maintains individual queues at eNodeB for each active user to store the excess packets. In this experiment, the scenario consists of one eNodeB and users are varied from 40 to 80 users in the eNodeB network. The users are in stationary position, and all the users use TCP application. The remaining simulation parameters and settings are the same as given in Section 4.1.

As noticed from Table 2, with an increase in queue size at eNodeB, the average delay and average throughput of the network increase, but the average PDF and fairness index slightly decrease. The average throughput of the system varies from 40 users to 80 users scenarios, due to the effects of other factors such as the CQI value of the user (signal strength received by the user from eNodeB), the number of users in the eNodeB coverage area, and the data traffic in the network. The throughput of the system increases initially up to a particular queue size, and once the queue size increases beyond the particular size, the throughput becomes almost constant, due to saturation of wireless link. Further increasing the size of the queue above the system capacity does not improve the system throughput and average PDF. But, the average packet delay increases owing to the fact that the packets get piled up in the queue and wait for the eNodeB

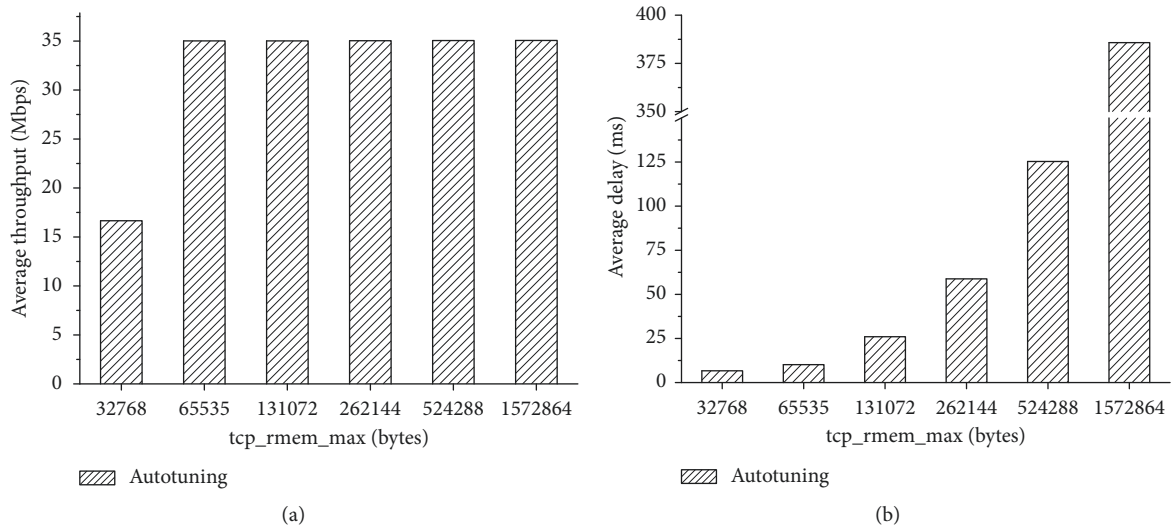

Figure 3: Performance of users with the autotuning method for varying receiver window sizes. (a) Average throughput. (b) Average delay.

Table 2: Performance of the system for varying queue size of the users at eNodeB.

Queue size (bytes)	Average throughput (Kbps)			Average delay (ms)			Average PDF (%)			Fairness index		
	40	60	80	40	60	80	40	60	80	40	60	80
10240	10148.45	11701.62	12748.21	40.70	143.89	153.37	99.48	99.38	99.48	0.9211	0.8227	0.8375
20480	11312.12	11883.18	12828.24	134.25	287.32	300.74	99.42	99.22	99.37	0.8772	0.8162	0.8339
30720	11754.05	11920.38	12857.73	218.17	424.76	442.24	99.31	98.91	99.15	0.8598	0.8157	0.8321
40960	11952.05	11933.80	12857.25	302.24	531.75	552.70	98.90	98.52	98.81	0.8466	0.8151	0.8323
51200	12163.32	11930.25	12857.71	372.80	671.89	638.36	98.62	98.29	98.51	0.8366	0.8152	0.8321

resources. Even if the number of users is increased, the behavior of the network remains the same.

From the above simulation results, it is noticed that maintaining a large queue space above the system execution capacity leads to bufferbloat problem. In the next experiments, the user queue size at eNodeB is fixed to 51200 bytes to create a bufferbloat problem.

4.3.3. Variation of CQI Value. In this experiment, the CQI value of the user is varied to investigate its effects on system performance. The eNodeB consists of 10 users in one sector, and the users are spread throughout the eNodeB sector to receive various signal strengths (CQI) from eNodeB. The modulation and coding scheme (MCS) value is computed in eNodeB based on the CQI value of the user and ranges from 0 to 28 [1]. The users are in stationary position and placed in such a way that the MCS value is increased in step of 4 from 0 to 28. All users use FTP application, and the queue size of users at eNodeB is fixed to 51200 bytes. The remaining simulation parameters and settings are the same as given in Section 4.1.

Figure 4(a) indicates that the throughput of the system increases with increase in the MCS value due to increase in signal strength. The main reason behind this behavior is that when the user is at 0 MCS value, it can send or receive only two bits/frame element (QPSK MCS), whereas a user at an MCS value of 28 can transmit/receive six bits/frame element (QAM MCS). As a result of variation in user signal strength, the throughput of the user also changes.

From Figure 4(b), it is observed that when the user is at 0 MCS value, the eNodeB takes more time to transmit a packet from eNodeB to the user. The reason is that eNodeB transfers less amount of bits/frame element. So, the packets get piled up in user queue at eNodeB and wait for eNodeB resource allocation. Hence, the packet transmission delay is increased. When the user signal strength (MCS) increases, the packet delay decreases because more amount of data bits in a frame element can be transferred, which in turn decreases the amount of packets waiting in the queue.

As observed from Figures 4(a) and 4(b), the CQI value of the user plays an important role in the performance of the LTE system.

4.3.4. Impact of Target Limit on Congestion Window. This experiment is conducted to study the impact of target limit on congestion window size, which affects the performance of the network. In this experiment, the eNodeB consists of users from 40 to 80 with the step of 40 users increased in each scenario and all users use TCP application. The proposed algorithm has been examined by varying target limit of the congestion window for both the increasing case as well as decreasing case of congestion window adjustment at the sender side. At the sender side, the $\text{targetlimit}_{\text{low}}$ on congestion window size is varied from 0 to

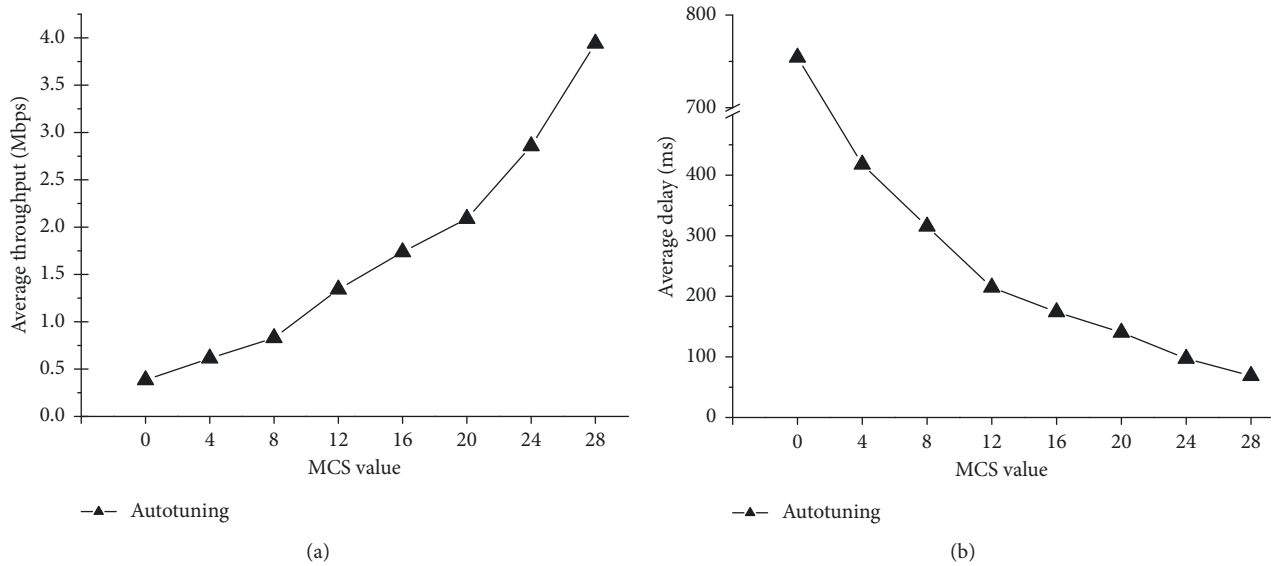

Figure 4: Performance of users with the autotuning method for varying MCS values. (a) Average throughput. (b) Average delay.

5000 bytes and the $\text{targetlimit}_{\text{high}}$ on congestion window size is varied from 5000 to 20000 bytes. The drop-tail queue is used as the user queue at eNodeB with the fixed queue size of 51200 bytes each. The remaining simulation parameters and settings are the same as shown in Section 4.1.

Case 1. Lower target limit ($\text{targetlimit}_{\text{low}}$) for decreasing congestion window.

Table 3 shows the experimental results for varying lower target limits of cwnd. As the lower target limit of congestion window increases, the average throughput of the system also increases. But, the average delay of the system also increases due to increase of data traffic in the network. The average PDF and fairness index of the LTE network is better when the threshold limit of cwnd is set to 3000 bytes compared to other threshold limits.

Case 2. Upper target limit ($\text{targetlimit}_{\text{high}}$) for increasing congestion window.

Table 4 shows the experimental results for varying upper target limits of cwnd. The average PDF, average delay, and fairness index of the network are better when the cwnd target limit is set to 10000 bytes compared to other target limits. But, the average throughput of the system slightly varies and depends on the number of users in the network.

4.3.5. Variation in the Number of Users (Background Traffic). This experiment investigates the effect of background data traffic density on performance of the system by varying the number of users in the eNodeB coverage area. In this scenario, the total number of users is varied from 20 to 100 in eNodeB, out of which 50% of users are TCP application users and remaining 50% of users are UDP application users. The users are in stationary position, and 51200 bytes is fixed as user queue size at eNodeB. The remaining simulation parameters and settings are the same as given in Section 4.1.

Table 5 shows the simulation results for varying number of users in eNodeB. When the number of users (20–40 users) is less in the eNodeB range, the throughput of the system is also less because the amount of data produced by the users is less compared to the execution capacity of the eNodeB. As the number of users is increased from 40 to 80 users in eNodeB network, the throughput of system also increases, due to the effect of multiuser diversity gain. But, once the number of users in the eNodeB coverage area crosses beyond 80 users, the throughput of the network decreases because users generate more data traffic beyond the execution capacity of the system as shown in Figure 5(a).

When users produce more data traffic (80 to 100 users scenarios) beyond the execution capacity of the eNodeB, few user packets get piled up in the user queue at eNodeB because of resource starvation. Hence, the user packets have to wait in the queue until the eNodeB resources are allocated to transfer the user packets. This process negatively impacts the packet delay of the users as shown in Figure 5(b).

The packet delivery fraction of the LTE network is good when data traffic of the users (number of users) is less in eNodeB network compared to the capacity of the system. As the number of users increases from 20 to 100 in LTE network, the packet delivery fraction of the system gradually decreases. This is because more data traffic is produced by users compared to the availability of resource blocks in eNodeB. As a result of this, the user packets get piled up in user queue at eNodeB and tend to queue overflow.

As observed from Table 5, the fairness index is less when the users are in the range of 20 to 40 in eNodeB network due to constant data traffic generation from video application users. The video application users use UDP protocol, where data are generated at constant bit rate and the FTP application users use the TCP protocol. When the data traffic from video application users is less, most of the eNodeB

Table 3: Performance of the system for varying lower target limit of cwnd.

Lower target limit	Average throughput (Kbps)		Average delay (ms)		Average PDF (%)		Fairness index	
	40	80	40	80	40	80	40	80
1000	9928.30	10984.08	14.92	26.44	99.83	99.75	0.84829	0.79059
2000	9915.07	10912.64	15.11	36.0	99.79	99.85	0.85983	0.81908
3000	9983.43	11738.27	15.2	55.17	99.82	99.97	0.88021	0.85614
4000	9990.70	12692.83	18.8	85.14	99.80	99.94	0.87335	0.8368
5000	10092.97	12733.44	25.37	105.67	99.75	99.93	0.91064	0.83787

Table 4: Performance of the system for varying upper target limit of cwnd.

Upper target limit	Average throughput (Kbps)		Average delay (ms)		Average PDF (%)		Fairness index	
	40	80	40	80	40	80	40	80
5000	9849.95	12511.00	16.75	69.00	99.83	99.95	0.87517	0.84125
10000	9983.43	11738.27	15.2	55.17	99.82	99.97	0.88021	0.85614
15000	9957.87	11690.08	17.02	55.90	99.81	99.92	0.86584	0.84284
20000	9775.26	11675.08	15.7	55.94	99.79	99.90	0.85070	0.84068

Table 5: Performance of the system for varying number of users in LTE network.

No. of users	Average throughput (Kbps)	Average delay (ms)	Average PDF (%)	Fairness index
20	7489.45	240.33	93.63	0.72445
40	10245.23	437.92	88.01	0.821573
60	11320.52	766.08	79.59	0.833096
80	12615.25	920.34	71.60	0.840856
100	12306.41	1275.72	57.40	0.828035

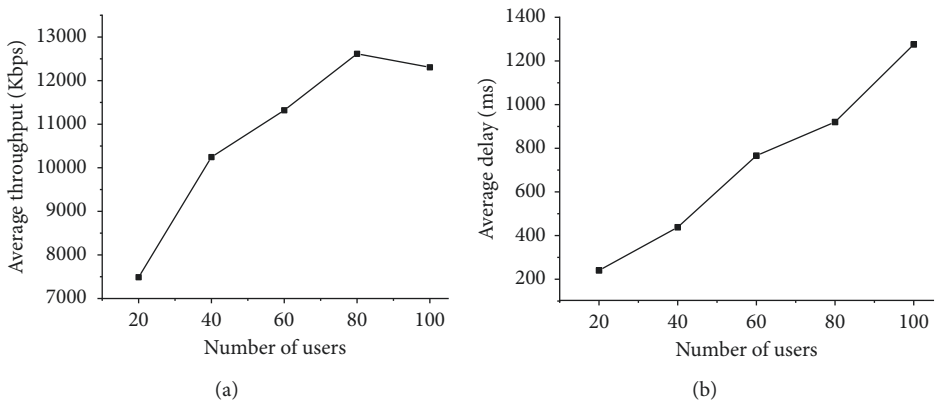

Figure 5: Performance of the system for varying number of users in LTE network. (a) Average throughput. (b) Average delay.

resources are consumed by FTP application users and as a result of this, fairness index is less. The fairness of the system increases as the user count increases above 40 in eNodeB network. This is due to increase in data traffic at eNodeB, and each user tries to grab the eNodeB resource. The eNodeB assigns the resources to the users based on resource allocation history. As a result of this condition, the eNodeB resources are shared equally among the active users in eNodeB network which slows down the high resource-consuming users.

Table 5 indicates that increase in traffic above the system execution capacity leads to high delay in packet transmission and increase in packet loss.

4.3.6. Estimation of Network Congestion at the Receiver Side. In this experiment, the performance of the network is studied by analyzing the network with and without estimation of network congestion at the receiver side. The scenario consists of one eNodeB, and users are varied from 20 to 100 in eNodeB network. Three TCP advertising

window methods are used at the receiver side to study the behavior of the algorithms. The methods are dynamic receiver-window adjustment (DRWA) [14], autotuning (conventional TCP NewReno), and adaptive receiver-window adjustment (the proposed ARWA method). The position of users is stationary, and 51200 bytes is fixed as user queue size at eNodeB. All users use TCP application. The remaining simulation parameters and settings are the same as given in Section 4.1.

The performance of DRWA, ARWA, and autotuning methods for varying number of users is shown in Table 6, and the corresponding graphs are given in Figure 6. Figure 6(a) indicates that the average throughput of the autotuning method is better compared to DRWA and ARWA methods due to the aggressive nature of packet generation rate at the sender side. The average throughput of ARWA is better than the DRWA method, and as the user count is increased in the network, the throughput of ARWA is almost equal to the throughput of autotuning. Another interesting observation is that the autotuning method suffers from huge delay (bufferbloat problem) as shown in Figure 6(b), especially when the number of users is increased in the network. These results again prove that the aggressive TCP sender can suffer from bufferbloat problem. Both DRWA and ARWA take less time to transfer a packet from the source to destination and maintain a less number of packets in user queue at eNodeB. The average packet delay of the ARWA method is slightly more than DRWA, but the packet delay is within the standard QCI limit [5]. The throughput of ARWA is improved by 23.68–35.41% compared to the DRWA method, and the delay of ARWA is reduced by 78.87–90% compared to the autotuning method.

The average PDF of various TCP advertising window methods is compared by varying the number of users in LTE network as shown in Figure 6(c). As the users are increased in the network, the PDF of the autotuning method is less compared to the ARWA and DRWA methods due to an aggressive data generation rate which exceeds system capacity (bandwidth). Therefore, the packets get piled up in queue, which tend to queue overflow. However, the increase in users at eNodeB network does not have much impact on the performance of DRWA and ARWA methods, due to the estimation of network congestion at the receiver side and control of the sender-side congestion window based on the receiver-advertising window value. These methods avoid the sender from overshooting packets to the network beyond the network capacity. The PDF of ARWA is improved by 0.36–2.17% compared to the autotuning method.

Figure 6(d) shows that the ARWA method achieves better fairness compared to other methods, while the DRWA method achieves lower fairness due to its conservative property. The reason is that when the user is in the low signal strength region of eNodeB, it can receive/transmit only two bits per frame element, and when the user is in good signal strength region of eNodeB, it can receive/transmit six bits per frame element. Hence, the user in low signal strength location can transmit/receive less data. The remaining packets get piled up in the queue of the user at eNodeB and wait for their turn to get eNodeB resource, which leads to rise in delay of packet transmission. When the packet delay is increased, the receiver reduces its rwnd value and intimates the sender to reduce its packet generation rate. Therefore, the user in the low signal strength region transmit/receive at lower data rate compared to users in good signal strength, which directly affects the fairness in sharing of eNodeB resources among the users. But, the ARWA method considers the data rate while calculating the rwnd value and maintains the minimum cwnd value (cwnd = 3000 bytes) at the sender side in order to avoid this problem. The fairness of ARWA is improved by 36.43–79.77% compared to the DRWA method.

4.3.7. Coexistence with Different Flows. In this section, the performance and behavior of coexistence of different flows of DRWA, ARWA, autotuning, and UDP users in LTE network are investigated. In this experiment, the eNodeB consists of 12 users in one sector and 51200 bytes is fixed as user queue size at eNodeB. The users are placed in three different locations based on signal strength (low, medium, and high CQI value) received from the eNodeB, and in each location, four users are placed with different flow methods (UDP, DRWA, ARWA, and autotuning). The remaining simulation parameters and settings are the same as given in Section 4.1.

Figure 7(a) shows the throughput of each user in different locations with different methods of flow process. When the users receive poor signal strength from eNodeB, the throughput of all the methods is less compared to the performance of the users at good signal strength locations. The throughput of the DRWA method is very less compared to other methods and the throughput of DRWA user in low signal strength location is almost nil. The DRWA method induces a fairness problem because of its conservative behavior, and it works like TCP Vegas [20], which suffers from grabbing the network resources in a competitive environment. The throughput of the proposed ARWA is almost equal to that of autotuning and UDP methods because it maintains the congestion window of the sender based on the data rate of the user at the receiver side and also ensures to maintain the minimum congestion window at the sender side when it goes down. In addition to that, ARWA maintains balanced throughput among the users in different signal strength locations.

Figure 7(b) shows the delay of each user in different locations with different flow methods. The users in location with good signal strength (high CQI) from eNodeB have less delay compared to users in poor signal strength (medium and low CQI) location. Overall, the delay of users using the DRWA method is less compared to other methods due to its delay-based congestion window adjustment. The packet delay of the UDP user is more compared to other methods because it does not bother about congestion in the network and the vast user queue space at eNodeB also affects. The delay of the proposed ARWA method is almost equal to the delay of the DRWA method and is better than the autotuning method because it maintains the congestion window of the sender-side based on delay estimation information from the receiver side.

Table 6: Performance of DRWA, ARWA, and autotuning methods for varying number of users in LTE network.

No. of users	Average throughput (Kbps)			Average delay (ms)			Average PDF (%)			Fairness index		
	DRWA	ARWA	Autotuning	DRWA	ARWA	Autotuning	DRWA	ARWA	Autotuning	DRWA	ARWA	Autotuning
20	6763.47	8863.27	10357.38	5.74	8.95	32.90	99.92	99.88	99.52	0.4169	0.6559	0.6551
40	6060.05	9985.29	10887.63	6.0	12.91	366.66	99.90	99.90	98.71	0.3694	0.8396	0.8411
60	7005.80	10824.92	11798.53	6.21	24.72	617.37	99.94	99.90	98.33	0.3624	0.9139	0.8156
80	6840.72	11738.27	12791.88	6.34	55.17	632.75	99.93	99.97	98.44	0.2066	0.8561	0.8295
100	7903.46	12238.15	12389.87	6.17	87.34	873.62	99.93	99.95	97.78	0.1698	0.8397	0.8329

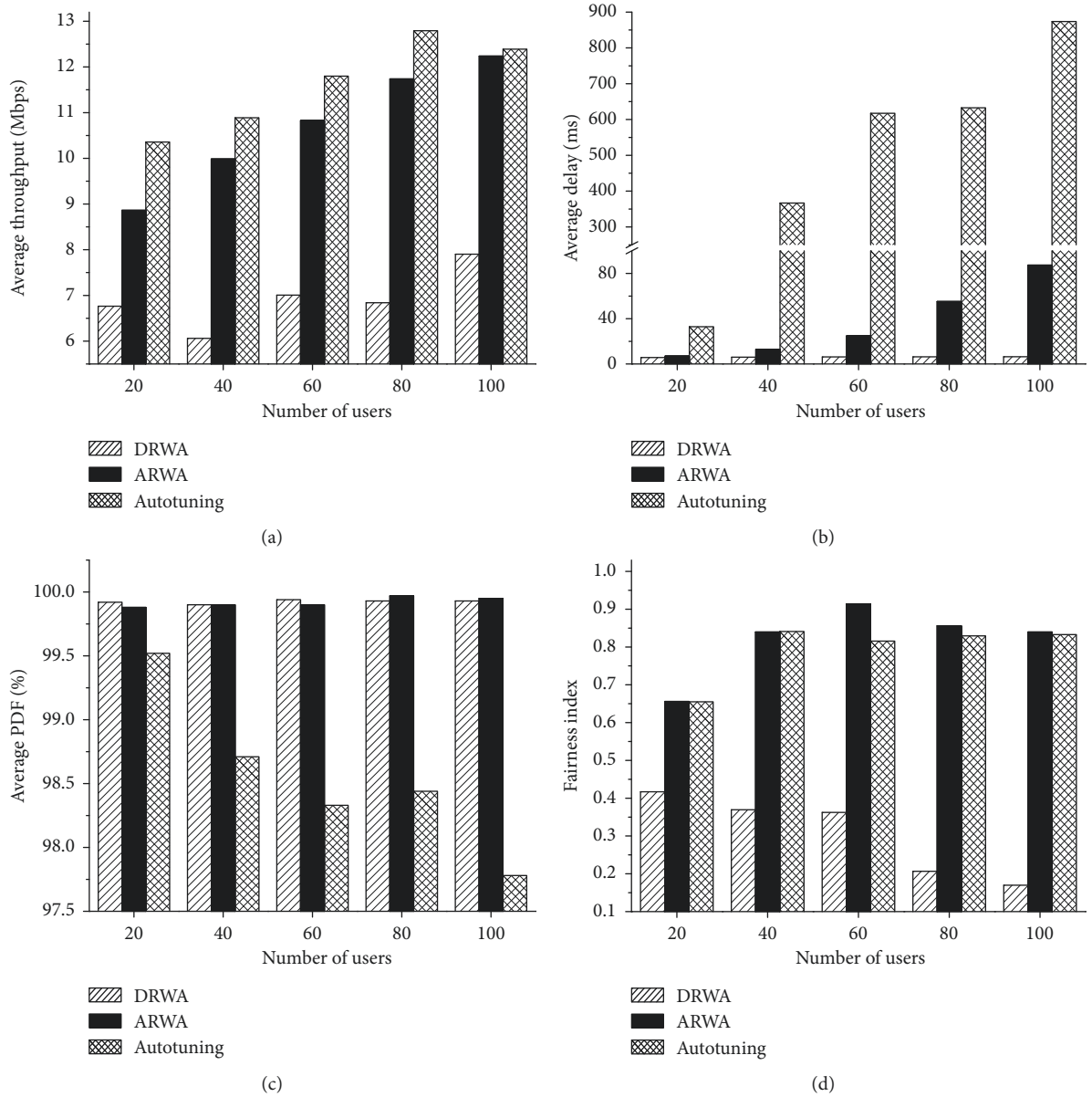

Figure 6: Performance of DRWA, ARWA, and autotuning methods for varying number of users in LTE network. (a) Average throughput. (b) Average delay. (c) Average PDF. (d) Fairness index.

Figure 8 illustrates the throughput of different flow methods for various signal strengths in LTE network. In Figure 8(a), the throughput of the DRWA method has high fluctuation compared to other methods due to the conservative behavior of the algorithm. The throughput of the autotuning method is better compared to other methods because the UDP method generates a constant number of bits per second and the ARWA and DRWA methods are

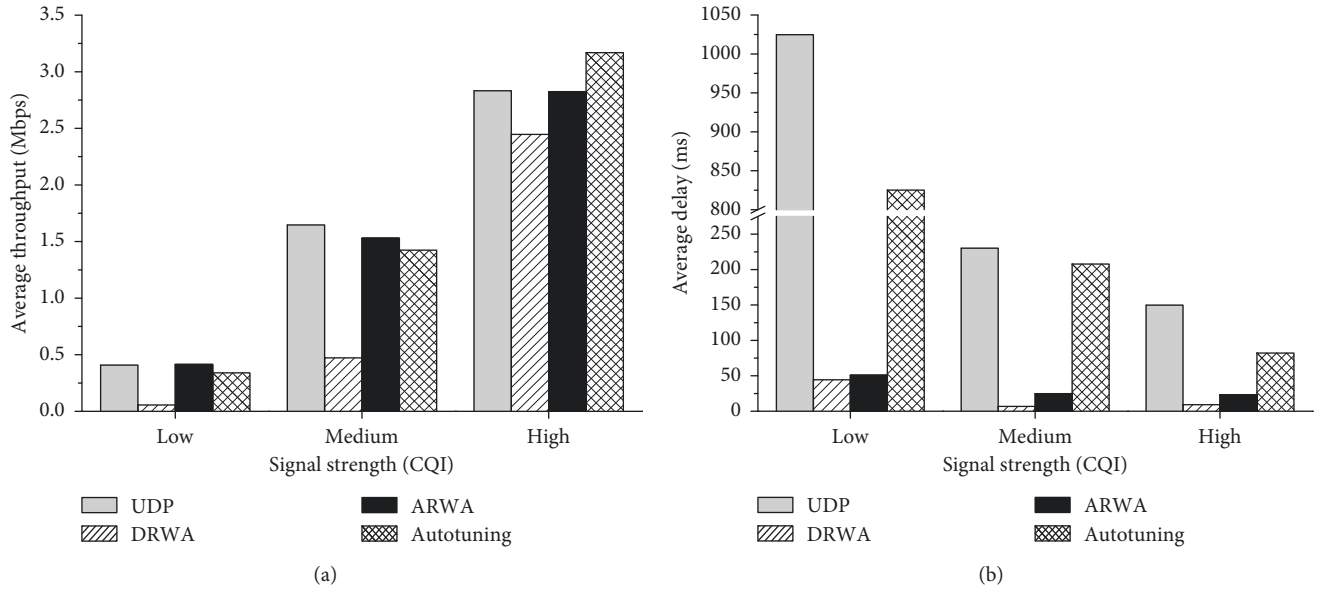

FIGURE 7: Performance comparison of different flow methods for various signal strengths in LTE network. (a) Average throughput. (b) Average delay.

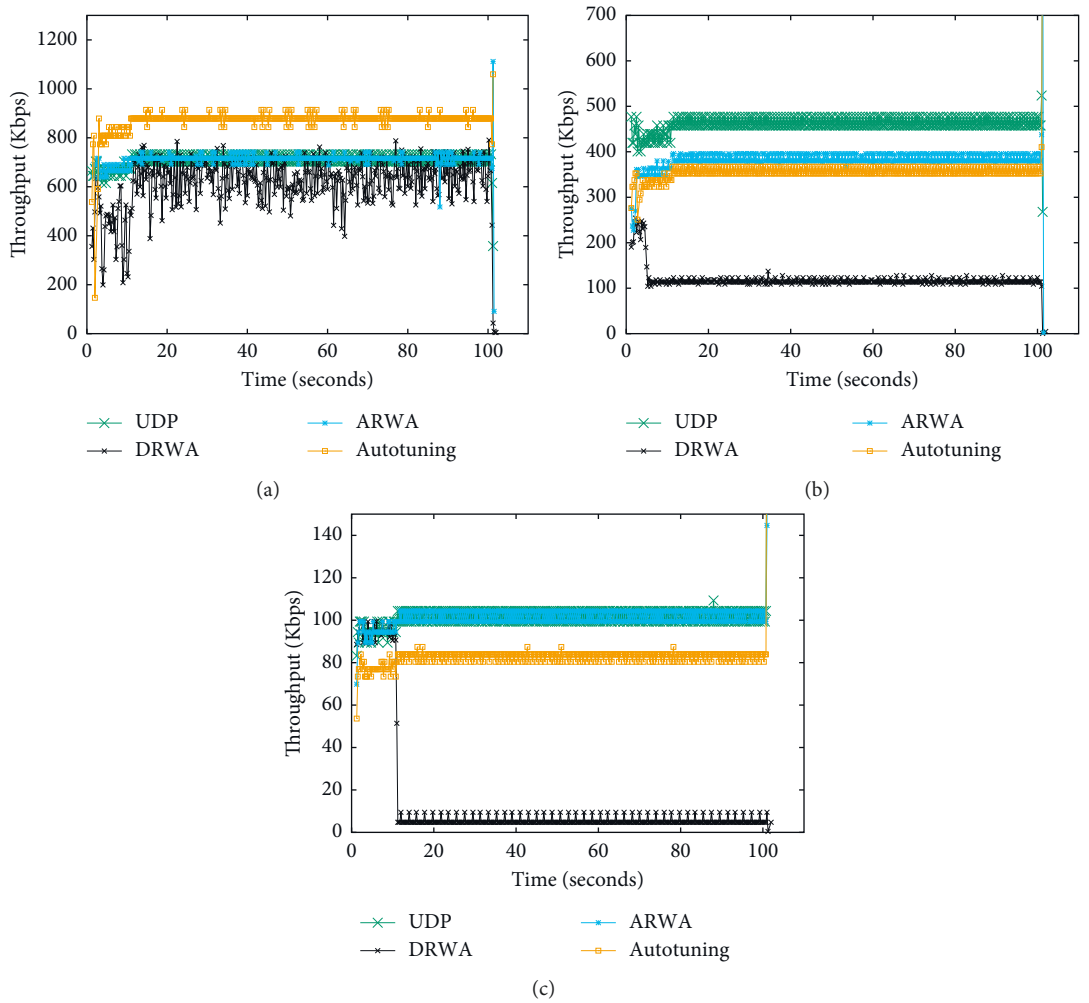

FIGURE 8: Throughput of different flow methods for various signal strength locations in LTE network. (a) High signal strength location. (b) Medium signal strength location. (c) Low signal strength location.

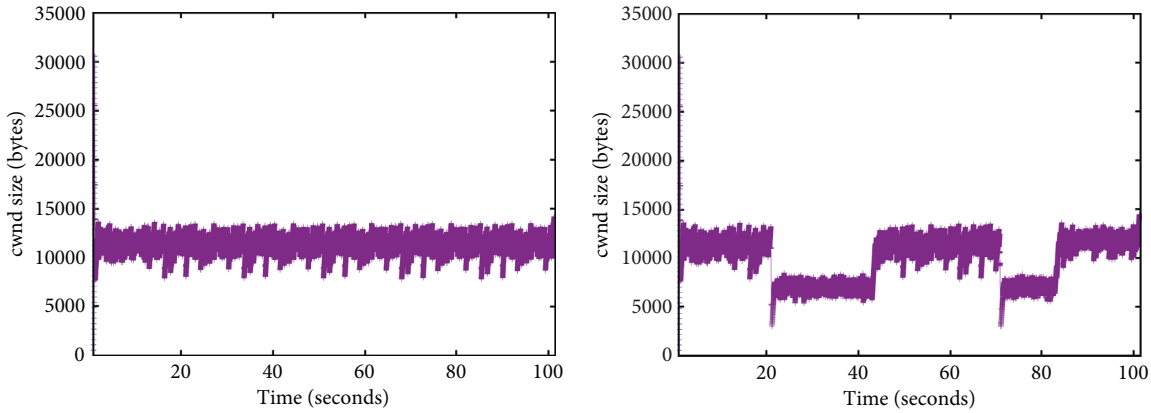

Figure 9: Performance of the ARWA method (b) with burst of data traffic and (a) without burst of data traffic in LTE network.

delay-based congestion control algorithms. So, the auto-tuning method goes on increasing the packet generation rate until the packet loss occurs in the network. In Figure 8(b), the throughput of UDP user is better than other methods due to the constant rate of generation of packets and also it does not bother about the packet loss (not aware of congestion), but other methods are aware of congestion control mechanism. The performance of the user of the DRWA method is lower compared to other methods because of the conservative behavior. In Figure 8(c), the performance of DRWA reaches almost the halt state, and the user suffers from resource starvation.

From the above experiment, it is observed that the overall performance of the ARWA method is better compared to other methods. It maintains the fairness among the users and suits for competitive environment, while DRWA severely suffers from resources starvation.

4.3.8. Variation of Congestion Window with Time. This experiment is conducted to study the variation of congestion window with respect to time, based on the intimation sent by the receiver, and also to study the reaction of the receiver and the sender (ARWA method) to sudden change in network traffic. In this scenario, the eNodeB consists of 20 users in one sector, out of which 19 users are UDP application users and one user is the TCP application user. The UDP users are used in scenario to generate constant data traffic in network. The user queue size at eNodeB is fixed to 51200 bytes. Initially, in order to study the variation of congestion window with time, without burst in traffic, out of 19 UDP application users, 10 users generate traffic throughout the simulation time but rest 9 users do not generate any data. In the next scenario, burst of data traffic is generated by the remaining 9 users in the network at simulation time from 20 seconds to 40 seconds and from 70 seconds to 80 seconds. This is used to study the variation of congestion windows when there is sudden burst of traffic. The remaining simulation parameters and settings are the same as given in Section 4.1.

Figures 9(a) and 9(b) shows the congestion window of the ARWA method without and with burst of data traffic in LTE network, respectively. When there is no burst of data traffic in network, the congestion window of the ARWA method slightly varies but maintains the constant range of variation. The traffic burst is generated from 20 seconds to 40 seconds and from 70 seconds to 80 seconds in the experiment, and these effects can be observed in Figure 9(b). The congestion window of the ARWA method is reduced due to the conservative behavior, to reduce the delay of packet delivery at the destination, and also due to the eNodeB resources being shared among the active users in the eNodeB coverage area. An interesting observation is that the ARWA method sender has not gone to halt state of packet generation. The congestion window size has been reduced to 3000 bytes but not below that value because of setting the lower target limit in the ARWA method. Then, it starts to grab the eNodeB resources and competes with other users. When data burst is reduced in the network after 40 seconds till 70 seconds and after 80 seconds, the ARWA method sender grabs the available resources in network. From this experiment, it is observed that the ARWA method responds to the burst of data traffic in network without degrading its overall performances.

5. Conclusion

The paper investigates the behavior of TCP in congested network over a large user queue space at the base station (eNodeB) and its effect on the performance of the system. The DRWA method [14] addresses the bufferbloat problem in cellular network, but the conservative behavior of the algorithm makes it less attractive in resource competitive environment. The user with the DRWA method suffers from resource starvation while competing with conventional TCP users. The throughput of the network also gets drastically reduced due to the proactive nature and the fairness problem while sharing resources among the users.

To overcome these problems, we have developed the ARWA method for LTE downlink communication. First, the ARWA receiver estimates the congestion level in the network by considering the packet delay information and its data rate. The estimated rwnd value is transmitted to the

sender via an advertised receiver window. Further, with the help of an estimated rwnd value, the sender side controls the packet generation rate. In addition to that, the sender side checks the cwnd threshold level before increasing or decreasing the congestion window size in order to utilize the network resources efficiently and maintain fairness among the users.

The proposed method is evaluated, and its performance is compared with the DRWA and autotuning methods in various scenario settings by using an ns-3 simulator. Simulation results illustrate that the ARWA algorithm prevents the bufferbloat problem and reduces the packet delay without reducing the throughput of the system. Moreover, the resource starvation and fairness problem is solved while competing with coexisting conventional TCP users and UDP users in the LTE network.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, "Downlink packet scheduling in LTE cellular networks: key design issues and a survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 678–700, 2013.

[2] S. Carson, P. Jonsson, J. S. Sethi et al., *Ericsson Mobility Report*, Ericsson White Paper, 2017, https://www.ericsson.com/en/mobility-report/reports/november-2017/key-figures.

[3] Cisco, *Cisco Visual Networking Index: Forecast and Methodology, 2016–2021*, Cisco White Paper, 2017, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf.

[4] A. S. A. Salam, M. Luglio, C. Roseti, and F. Zampognaro, "TCP wave: a new reliable transport approach for future internet," *Computer Networks-Elsevier*, vol. 112, pp. 122–143, 2017.

[5] R. Zhu and J. Yang, "Buffer-aware adaptive resource allocation scheme in LTE transmission systems," *EURASIP Journal on Wireless Communications and Networking- Springer*, vol. 2015, p. 176, 2015.

[6] M. A. Ali, A. Esmailpour, and N. Nasser, "Performance evaluation for LTE applications with buffer awareness consideration," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–7, IEEE, Doha, Qatar, April 2016.

[7] A. Zolfaghari and H. Taheri, "Queue-aware channel-adapted scheduling and congestion control for best-effort services in LTE networks," *Canadian Journal of Electrical and Computer Engineering*, vol. 38, no. 2, 2015.

[8] P. Imputato and S. Avallone, "An analysis of the impact of network device buffers on packet schedulers through experiments and simulations," *Simulation Modelling Practice and Theory*, vol. 80, pp. 1–18, 2018.

[9] K. Nichols and V. Jacobson, "Controlling queue delay," *Queue*, vol. 10, no. 5, p. 20, 2012.

[10] M. P. Tahiliani, K. C. Shet, and T. G. Basavaraju, "CARED: cautious adaptive RED gateways for TCP/IP networks," *Journal of Network and Computer Applications*, vol. 35, no. 2, pp. 857–864, 2012.

[11] R. Pan, P. Natarajan, C. Piglione et al., "PIE: a lightweight control scheme to address the bufferbloat problem," in *Proceeding of IEEE 14th International Conference on High Performance Switching and Routing, HPSR*, pp. 148–155, IEEE, Taipei, Taiwan, July 2013.

[12] Y. Tan, R. Susitaival, and J. Torsner, "Active queue management for LTE uplink in eNodeB," September 2010, http://www.google.com/patents/WO2010107355A1?cl=en.

[13] H. Im, C. Joo, T. Lee, and S. Bahk, "Receiver-side TCP countermeasure to bufferbloat in wireless access networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, pp. 2080–2093, 2016.

[14] H. Jiang, Y. Wang, K. Lee, and I. Rhee, "DRWA: a receiver-centric solution to bufferbloat in cellular networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 11, pp. 2719–2734, 2016.

[15] S. C. F. Chan, K. M. Chan, K. Liu, and J. Y. B. Lee, "On queue length and link buffer size estimation in 3G/4G mobile data networks," *IEEE Transactions in Mobile Computing*, vol. 13, no. 6, pp. 1298–1311, 2014.

[16] R. ROBERT, "Comparative study of the performance of TCP congestion control algorithms in an LTE network," Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2016.

[17] P. J. Argibay-Losada, K. Nozhnina, A. Suárez-González, C. López-García, and M. Fernández-Veiga, "Loss-based proportional fairness in multihop wireless networks," *Wireless Networks*, vol. 20, no. 5, pp. 805–816, 2013.

[18] B. Qureshi, M. Othman, S. Subramaniam, and N. A. Wati, "QTCP: improving throughput performance evaluation with high-speed networks," *Arabian Journal for Science and Engineering*, vol. 38, no. 10, pp. 2663–2691, 2012.

[19] N. Iya, N. Kuhn, F. Verdicchio, and G. Fairhurst, "Analyzing the impact of bufferbloat on latency-sensitive applications," in *Proceedings of IEEE International Conference on Communications (ICC)*, pp. 6098–6103, IEEE, London, UK, June 2015.

[20] L. S. Brakmo and L. L. Peterson, "TCP Vegas: end to end congestion avoidance on a global Internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, 1995.

[21] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP westwood: bandwidth estimation for enhanced transport over wireless links," in *Proceedings of ACM the Seventh Annual International Conference on Mobile Computing and Networking MOBICOM*, pp. 287–297, ACM, Rome, Italy, July 2001.

[22] A. A. Salman, "Delay Aware and users categorizing-based call admission control for multi-services LTE-A networks," *Arabian Journal for Science and Engineering*, vol. 41, no. 9, pp. 3631–3644, 2016.

[23] M. Casoni, C. A. Grazia, M. Klapez, and N. Patriciello, "How to avoid TCP congestion without dropping packets: an effective AQM called PINK," *Computer Communications*, vol. 103, pp. 49–60, 2017.

[24] Y. Zaki, T. Weerawardane, S. Hauth, E. Wallmeier, and C. Gorg, "Intelligent traffic enforcement for LTE backhaul," in *Proceeding of IEEE 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, pp. 3077–3082, IEEE, London, UK, September 2013.

[25] J. Huang, F. Qian, Y. Guo et al., "An in-depth study of LTE: effect of network protocol and application behavior on

performance," in *Proceeding of ACM conference on SIGCOMM*, ACM, Hong Kong, China, August 2013.

[26] R. Kwan, R. Arnott, R. Trivisonno, and M. Kubota, "On preemption and congestion control for LTE systems," in *Proceeding of IEEE 72nd Vehicular Technology Conference Fall (VTC 2010-Fall)*, pp. 1–5, IEEE, Ottawa, ON, Canada, September 2010.

[27] X. Wang and S. Konishi, "A novel TCP-oriented multi-layer packet scheduling algorithm," in *Proceedings of IEEE 73rd Vehicular Technology Conference (VTC Spring)*, IEEE, Budapest, Hungary, May 2011.

[28] N. Kuhn, D. Ros, A. B. Bagayoko, C. Kulatunga, G. Fairhurst, and N. Khademi, "Operating ranges, tunability and performance of codel and PIE," *Computer Communications*, vol. 103, pp. 74–82, 2017.

[29] P. C. Lin, R. G. Cheng, X. Wang, and P. A. R. Suryadhi, "PFCS: pre-buffering-aware flow control scheme for LTE-advanced relay networks," in *Proceedings of 11th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE*, pp. 155–159, IEEE, Taipei, Taiwan, August 2015.

[30] E. Atxutegi, F. Liberal, K.-J. Grinnemo, A. Brunstrom, A. Arvidsson, and R. Robert, "TCP behaviour in LTE: impact of flow start-up and mobility," in *Proceedings of 9th IFIP conference on Wireless and Mobile Networking (WMNC)*, pp. 73–80, IEEE, Colmar, France, July 2016.

[31] Z. Wang, X. Zeng, X. Liu, M. Xu, Y. Wen, and L. Chen, "TCP congestion control algorithm for heterogeneous internet," *Journal of Network and Computer Applications*, vol. 68, pp. 56–64, 2016.

[32] S. K. Bisoy and P. K. Pattnaik, "Design of feedback controller for tcp/aqm networks," *Engineering Science and Technology, an International Journal*, vol. 20, no. 1, pp. 116–132, 2017.

[33] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[34] S. Floyd, "TCP and explicit congestion notification," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 5, pp. 8–23, 1994.

[35] K. Liu and J. Y. Lee, "On improving TCP performance over mobile data networks," *IEEE Transaction on Mobile computing*, vol. 15, no. 10, 2016.

[36] A. K. Paul, H. K. A. Tachibana, and T. Hasegawa, "An AQM based congestion control for eNB RLC in 4G/LTE network," in *Proceedings of Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–5, IEEE, Vancouver, BC, Canada, May 2016.

[37] K. Shi, Y. Shu, O. Yang, and J. Luo, "Receiver-assisted congestion control to achieve high throughput in lossy wireless networks," *IEEE Transactions on Nuclear Science*, vol. 57, no. 2, pp. 491–496, 2010.

[38] 3rd Generation Partnership Project, "Policy and charging control (release 13)," Technial Report 3GPP TS 29.212 V13.4.0, European Telecommunications Standards Institute, Sophia Antipolis, France, 2016.

[39] X. Jiang and G. Jin, "Adaptive low-priority congestion control for high bandwidth-delay product and wireless networks," *Computer Communications*, vol. 105, pp. 44–52, 2017.

[40] T. Henderson, "ns-3 Tutorial," 2018, https://www.nsnam.org/docs/release/3.27/models/html/index.html.