Buffer De-bloating in Wireless Access Networks

by

Yuhang Dai

A thesis submitted to the University of London for the degree of
Doctor of Philosophy

School of Electronic Engineering & Computer Science
Queen Mary University of London
United Kingdom

Sep 2018

TO MY FAMILY

Abstract

Excessive buffering brings a new challenge into the networks which is known as Bufferbloat, which is harmful to delay sensitive applications. Wireless access networks consist of Wi-Fi and cellular networks. In the thesis, the performance of CoDel and RED are investigated in Wi-Fi networks with different types of traffic. Results show that CoDel and RED work well in Wi-Fi networks, due to the similarity of protocol structures of Wi-Fi and wired networks.

It is difficult for RED to tune parameters in cellular networks because of the time-varying channel. CoDel needs modifications as it drops the first packet of queue and the head packet in cellular networks will be segmented. The major contribution of this thesis is that three new AQM algorithms tailored to cellular networks are proposed to alleviate large queuing delays.

A channel quality aware AQM is proposed using the CQI. The proposed algorithm is tested with a single cell topology and simulation results show that the proposed algorithm reduces the average queuing delay for each user by 40% on average with TCP traffic compared to CoDel.

A QoE aware AQM is proposed for VoIP traffic. Drops and delay are monitored and turned into QoE by mathematical models. The proposed algorithm is tested in NS3 and compared with CoDel, and it enhances the QoE of VoIP traffic and the average end-to-end delay is reduced by more than 200 ms when multiple users with different CQI compete for the wireless channel.

A random back-off AQM is proposed to alleviate the queuing delay created by video in cellular networks. The proposed algorithm monitors the play-out buffer and postpones the request of the next packet. The proposed algorithm is tested in various scenarios and it outperforms CoDel by 18% in controlling the average end-to-end delay when users have different channel conditions.

Delaration

I, Yuhang Dai, confirm that the work presented in this thesis is my own work and it is original except where due reference has been mentioned.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis. I confirm that this thesis has not been previously submitted of for the aware of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be publish without the prior written consent of the author.

Signature:

Date:

Acknowledgments

First of all, I would like to express the depth of my gratitude to my supervisor, Dr. Vindya Wijeratne, for he constant encouragement and guidance. She has walked me through all the stage of my Ph.D. study.

I would also like to express my gratitude to Dr. John Schormans for his priceless advices in my research and effort in my thesis writing. Without his consistent and illuminating instructions, this thesis could not have its present form.

I would also like to thank Prof. Yue Chen for her valuable suggestion and discussions in the group meeting.

Meanwhile, I would like the thank you all the college and friends at Queen Mary. It is a great to meet all of you.

I would also like to thank my family and girlfriend, who gives me support and courage through my Ph.D.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

3GPP	Third Generation Partnership Project
4G	Fourth Generation
ACK	Acknowledgement
AM	Acknowledged Mode
AP	Access Point
AQM	Active Queue Management
ARED	Adaptive RED
ARPANET	Advanced Research Projects Agency Network
AWND	Advertised Window
BS	Base Station
BBU	Baseband Units
BER	Bit Error Rate
CA	Congestion Avoidance
CAGR	Compound Annual Growth Rate
CoDel	Controlled Delay
CWND	Congestion Window
CPRI	Common Public Radio Interface
CQI	Channel Quality Index

CRRA	Cooperative Radio Resource allocation
CSCF	Call Session Control Function
CSI	Channel State Information
DRR	Deficit Round Robin
DRWA	Dynamic Receive Window Adjustment
DWRR	Deficit Weighted Round Robin
eNB	eNode Base Station
EPS	Evolved Packet System
E-RAB	E-Radio Access Bearer
FCFS	First Come First Serve
FIFO	First In First Out
FQ	Flow Queuing
FRED	Fair RED
FWMRED	Fair Weighted Multi-Level Random Early Detection
GBR	Guaranteed Bit Rate
HSS	Home Subscriber Server
IoT	Internet of Things
ISP	Internet Service Provider
IID	Independent and Identical Distributed
IP	Internet Protocol
IMS	IP Multimedia Subsystem
I-CSCF	Interrogating CSCF
KPI	Key Performance Indicator
LEDBAT	Low Extra Delay background Transport
LTE	Long Term Evolution
MPTCP	Multi Path TCP

MB	Megabyte
MOS	Mean Opinion of Source
MPD	Media Presentation Description
mmWave	Millimeter Wave
NIC	Network Interface Card
NGN	Next Generation Network
NS3	Network Simulator 3
P-CSCF	Proxy CSCF
PDN	Packet Data Network
PIE	Proportional Integral Enhanced
PSTN	Public Switched Telephone Network
PUCCH	Public Uplink Control Channel
PUSCH	Public Uplink Share Channel
QoS	Quality of Service
QoE	Quality of Experience
RCPQD	Rate Controlled Priority Queuing Discipline
RED	Random Early Detection
RLC	Radio Link Control
RR	Round Robin
RTAC	Receiver-side TCP Adaptive queue Control
RTT	Round Trip Time
RTP	Real Time Transport Protocol
RTSP	Real Time Streaming Protocol
RWND	Receiver Window
SABRE	Smooth Adaptive Bit Rate
S-CSCF	Serving CSCF
SIP	Session Initiation Protocol

SGW	Service Gateway
TCP	Transmission Control Protocol
TID	Traffic Identifier
TM	Transient Mode
TTI	Transmission Time Interval
TPP	TCP Packet Pacing
UDP	User Datagram Protocol
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
VoIP	Voice over IP
WRR	Weighted Round Robin
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network

Chapter 1

Introduction

Mobile devices are becoming increasingly powerful these days. It is quite common that one person has multiple devices connected to the network, which leads to the rise of global data traffic. Smart phones and tablets facilitates people's life. Nowadays, people can have access to the Internet with different type of equipment and almost whenever and wherever they want. Increasing numbers of devices accessing to the Internet causes the growth of Internet traffic. As a result, delay of packets increases as packets waiting to be transmitted in the queue along the Internet path. Increasing queuing delay of packets has drawn researchers' attentions. The phenomenon here referred to as Bufferbloat.

Three conditions are essential to trigger Bufferbloat. First is the bursty traffic which makes it too hard to transmit packets at the intermediate node in the network in time; second is that the node has a large buffer; third is that packets in the buffer are for different clients. Bufferbloat potentially causes high latency and jitter, as well as reducing the overall network throughput as a consequence of TCP synchronization.

Bufferbloat can happen in both wired and wireless networks. The access points (AP) in wireless networks are often considered as the bottleneck of the whole path of a connection. The unique features of wireless access networks, such as the time-varying channel, multi-path fading, high bit error rate and etc. [CT14][JBT14], make it more challenging to solve Bufferbloat issues.

1.1 Motivation

Ludwing, R., et al.[LRK+99] first pointed out the potential issue of over-buffering in cellular networks in 1999, focusing on GPRS networks. Studies [JBT14][JWLR12] have also confirmed that Bufferbloat indeed exists not only within cellular networks but also in Wi-Fi networks and pointed out that it can lead to RTT in the order of seconds for cellular networks. However, recent studies [HKT+17][GPKC17] focus on WiFi and Satellite Networks but few papers focus on cellular networks. With the growth of hand-held devices (e.g., smartphones and tablets) and with most devices equipped with both Wi-Fi and cellular interfaces, wireless access to the Internet is growing fast. A forecast [cic17] from CISCO shows that the global mobile data traffic of 2021 will be aroud 7 times of that of 2016 due to growth of smart mobile devices, data-demanding applications and services such as gaming, video and VoIP. Such kind of interactive applications, are delay-sensitive. It can be seen that wireless networks are becoming an increasingly important technology to access the Internet. Different from wired network with a fixed bandwidth, wireless access networks have variable bandwidth as the number of users are changing and there are interferences. As pointed out by [AGG+13], wireless networks have become an integral part of day-to-day-life and suffer the most from large RTT, while minimal work is done so far in solving long RTT in cellular networks. It is truly significant and timely to focus on buffer de-bloating in wireless environments.

1.2 Aims and Objectives

The aim of this research is to mitigate Bufferbloat in wireless access networks by devising AQM algorithm tailored to the target environment. The objective is

1. Design AQM algorithms for wireless access networks.

2. Evaluate AQM algorithms in wireless access networks.

3. Reduce the queuing delay of each packets

The measurable outputs are end-to-end delay, packet loss probability, link utilization and Jain's Fairness Index. Jain's Fairness is defined in terms of throughput. Results are discussed in later chapters.

1.3 Novelty and Contributions

The novelty of the research presented in this thesis mainly comes from the three novel AQMs proposed and deployed in wireless access networks. Existing works to solve the Bufferbloat issues mainly focus on wired networks and Wi-Fi networks. This thesis focuses on cellular networks. The contributions are:

1. Evaluate DropTail, CoDel and RED in Wi-Fi access networks with FTP and VoIP Traffic.

2. Involve cross layer information in dropping decisions of AQMs.

3. Involve the Quality of Experience (QoE) metric to balance the drop and delay of packets.

4. Propose AQMs for specific delay sensitive applications, VoIP and Dynamic Adaptive Stream over HTTP (DASH).

5. Involve queuing theory, G/G/1 queue, to forecast the trend of the queue.

1.4 Outline of the Thesis

Chapter 2 reviews the techniques of network congestion control algorithms and state-of-the-art technologies fighting against Bufferbloat issues. **Chapter 3** evaluates and analyses the performance of RED and CoDel in Wi-Fi Networks with FTP and VoIP traffic. **Chapter 4** proposes a novel AQM to alleviate congestion in cellular networks. The proposed algorithm takes the channel conditions into consideration, which improve the performance of the proposed algorithm. Compared with CoDel, the proposed algorithm can further reduce the delay and keeps fairness among different users. **Chapter 5** proposes a novel AQM helping to enhance the Quality of Experience (QoE) of VoIP traffic in cellular networks. The proposed algorithm uses QoE as the performance metric and is based on queuing theory. Compared with CoDel, the proposed algorithm makes a balance between drop and delay, hence improving the QoE of VoIP traffic. **Chapter 6** proposes a novel AQM for video traffic. The video contents nowadays are mainly based on DASH. The proposed algorithm is deployed in the DASH client. It monitors the play out buffer of the client and reduces queuing delay of packets by delay the request of following video contents. The proposed algorithm is compared with CoDel and it outperforms CoDel in controlling the queuing delay. **Chapter 7** summaries the work in this thesis and discuss the future work.

Chapter 2

Background and Literature Review

2.1 Introduction

In 1984, congestion in communication networks was first pointed out by John Nagle in [Nag84] and two years later, the Advanced Research Projects Agency Network (ARPANET), which is considered as the original of the Internet, suffered "congestion collapse" [Jac88]. Congestion collapse occurs when huge amounts of traffic is injected into the networks, which results in large numbers of packets being dropped and few of packets being delivered. Congestion often happens at bottleneck links where the incoming traffic rate is higher than the outgoing traffic. The initial solution that engineers came up with at that moment is to increase the bandwidth. However, the "congestion collapse" came over again and again until 1988 when Van Jackson enhanced the Transport Control Protocol (TCP) [Jac88] where the Additive Increase and Multiplicative Decrease (AIMD) of the window size were proposed. Although "congestion collapse" is gone, researchers and engineers never stop fighting with congestion as the traffic keeps increasing (see Cisco's forecast [cic17]). Meanwhile, the number of devices is also increasing, as shown in Fig 2.1. With the increasing number of devices and traffic, congestion happens when

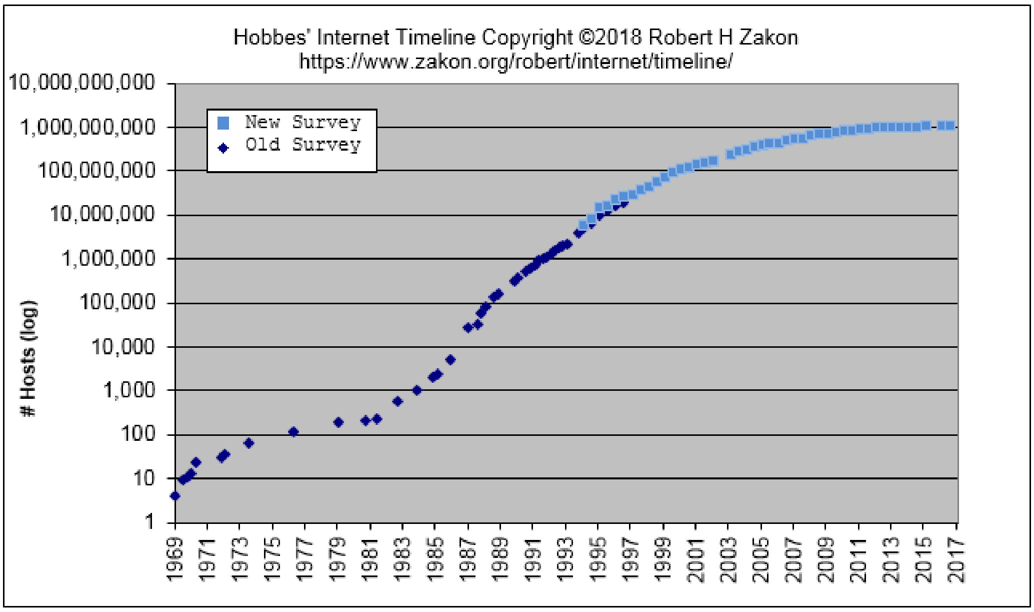

Figure 2.1: Internet Host

the total number of packets in flight approaches the processing capacity of the network. Congestion often happens at bottleneck links where the traffic comes faster than the depletion rate of the packets in the buffer as shown in Fig 2.2. Packets are injected into

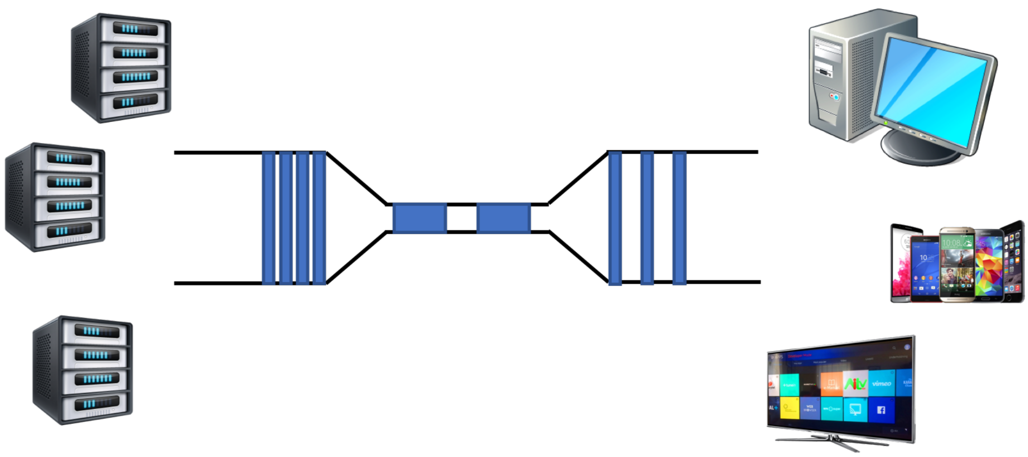

Figure 2.2: Bottleneck Link

the Internet and accumulated in the buffer where there is a bottleneck link. According to the locations of the congestion control algorithms, they can be either end-to-end congestion control algorithms, such as TCP and its variants, or hop-by-hop congestion algorithms, such as AQMs. These are discussed in Section 2.2 and 2.3.

2.2 End-to-end Congestion Control

End-to-end congestion control algorithms are implemented at the end host, which detect the congestion of the network according to the data transmitted or acknowledgements (ACKs) received. TCP and its variants play an important role in end-to-end congestion control. TCP tries to make full use of the bandwidth by increasing the sending rate when the ACKs of packets sent before are being received. Defined in [Pos03], TCP is a protocol for reliable transmission. TCP is connection oriented by establishing a flow between tow machines. Each TCP flow is identified by 5-tuples which are source IP address and port, destination IP address and port and the protocol. The reliability of TCP is assured by sending ACKs back to the sender confirming the received packets. Each packet has a unique sequence number in its header hence the receiver can respond to the sender exactly which packet it receives. Since October 1988, when TCP was first improved [Jac88], many efforts have been made to further enhance the performance of TCP. According to the type of the mechanisms, there are now loss based, delay based and hybrid TCP.

2.2.1 Loss based TCP

Loss based TCP controls congestion by decreasing the sending rate when the loss of packets occurs. Duplicate (usually 3) ACKs or timeout of waiting for the ACK is used as a signal of congestion in loss-based TCP, such as TCP Tahoe and Reno as shown in Fig 2.3. Except for the mentioned AIMD algorithm, they both have Slow start (SS), Congestion Avoidance (CA), and fast retransmission. In the SS phase, the Congestion Window (CWND) size increases exponentially. In the CA phase, the CWND is increased by $2/CWND$ each time an ACK is received. When a TCP connection is established, the CWND is initialized with 2 and TCP enters SS phase. A threshold, ssthresh, is the upper bound of the SS phase. Once CWND exceeds ssthresh, TCP enters CA phase. A timer will expire if the ACK of a packet is not received, which is called timeout. Timeout usually means the loss of a packet or congestion in the networks. Once timeout happens, CWND will be reset and TCP enters SS phase. Timeout is a way to detect the loss of a packet, however it will take too long for TCP to react to the loss. Fast retransmission is proposed to take advantage of three duplicate ACK. When three duplicate ACK are received, the TCP sender will retransmission the lost packet without waiting timeout. The difference between TCP Tahoe and Reno is that Reno also has a Fast Recovery algorithm. Fast Recovery sets the ssthresh to half of the current value of the CWND and the CWND is set to ssthresh plus 3 more segments. There are other loss based

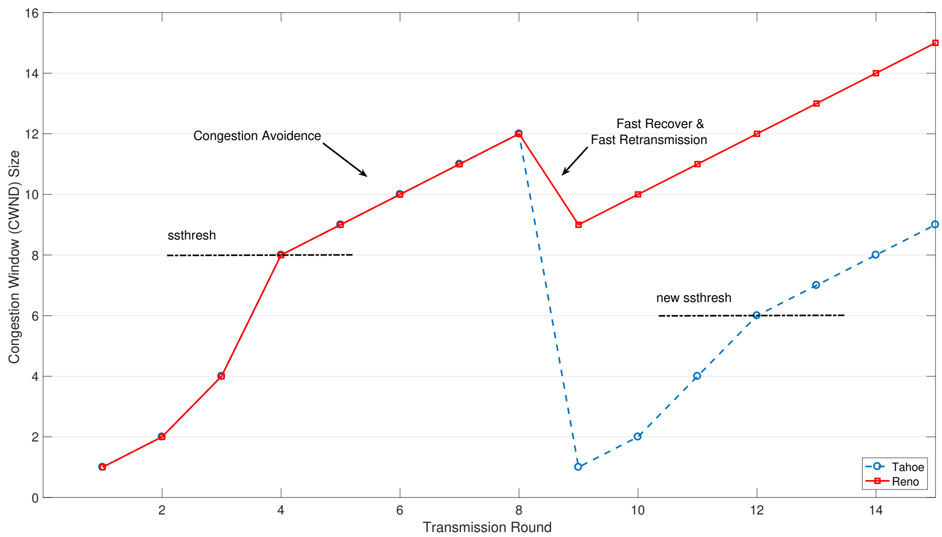

Figure 2.3: Congestion Window of Tahoe and Reno

TCP with different algorithms to adjust the CWND, such as Cubic TCP [HRX08], which uses a cubic function to adjust the CWND. The idea of loss based is simple and straightforward. However, due to the abuse of buffering in today's networks, excessive packets are kept in the buffer in case of packet dropping due to bursts of traffic, which results in the failure to detect congestion in loss based TCP.

2.2.2 Delay based TCP

In order to overcome the issues brought about by loss-based TCP, delay-based TCP (which uses delay as the symptom) are designed, and TCP Vegas [BOP94] is the most typical one. Vegas monitors the change of Round Trip Time of the packets in flight and adjusts the CWND according to the difference between the expected throughput and actual throughput. The expected throughput is calculated by Eq. 2.1 where $BaseRTT$ is the minimum RTT when there is no congestion in the network. The actual throughput is calculated by $Actual = \frac{CWND}{RTT}$ where RTT is the actually measured RTT value of each packet. The difference is calculated by $Diff = (Expected - Actual) * BaseRTT$. Then, the difference is compared with α and β. α and β are predefined constant values.

$$Expected = \frac{CWND}{BaseRTT} \tag{2.1}$$

$$Actual = \frac{CWND}{RTT} \tag{2.2}$$

$$Diff = (Expected - Actual) * BaseRTT \tag{2.3}$$

$$CWND = \begin{cases} CWND + 1 \;\; if \; Diff < \alpha \\ CWND - 1 \;\; if \; Diff > \beta \end{cases} \tag{2.4}$$

Delay-based TCP is effective when dealing with large buffers. However it is not as competitive as loss-based TCP, because when the packets accumulate in the buffer, delay-based TCP begins to decrease the sending rate while loss-based TCP still keeps increasing the sending rate, which will cause a fairness problem [JCH84] considering the throughput of each user.

2.2.3 Hybrid TCP

Hybrid-TCP uses both packet loss and delay as a combined symptom, such as TCP-FIT [WWZH11]. TCP-FIT is designed to solve the fairness issue of delay-based TCP. TCP-FIT adjusts CWND by Eq. 2.5, where N is the number of TCP flows using TCP-FIT algorithm. From Eq.2.5, it can be seen that CWND is updated every RTT. However, the flow with larger RTT will have a lower frequency of updating their CWND. To solve this issue, $\gamma = RTT/RTT_0$ is introduced as the normalization factor and RTT_0 is the statistical “floor” of the RTT values in the network. The number of flows is estimated by Eq. 2.6 where α is a preset parameter and Q is the estimation of the packets queued in the networks. Q is estimated as Eq. 2.7.

$$\begin{aligned} Each\ RTT: \ CWND &\leftarrow CWND + \gamma N \\ Each\ Loss: \ CWND &\leftarrow CWND - (CWND/2N) \end{aligned} \tag{2.5}$$

$$N_{i+1} = max\{1, \ N_i + (\alpha - \frac{Q}{CWND} N_i)\} \tag{2.6}$$

$$Q = (avg\_rtt - rtt\_min)\frac{CWND}{avg\_rtt} \tag{2.7}$$

There are other Hybrid TCPs with different algorithms to adjust the CWND, such as Compound TCP [TSZS06] specific for Windows Operating System, and TCP Westwood [CGM+02].

2.2.4 Multi Path TCP

Multi Path TCP (MPTCP) is proposed to allow a single TCP connection to take advantage of multiple paths to transmit data simultaneously, as shown in 2.4. Compared with traditional TCP flows, sub-flows are established to transmit packets from the same

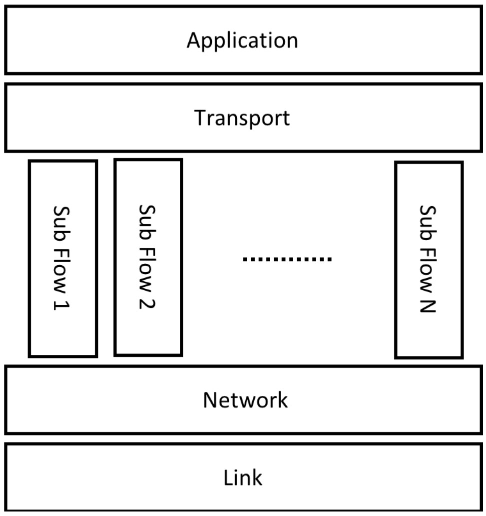

Figure 2.4: Multi Path TCP Illustration

flow. MPTCP has advantages in competing the bandwidth and MPTCP can be used to stablize a connection. Nowadays, mobile devices are often equipped with two different types of interfaces accessing the Internet Wi-Fi and Cellular. As is shown in Figure 2.5, MPTCP establishes multiple paths instead of a single path. MPTCP wants to distribute the load onto other available paths, hence improve the link utilization of network resources [FRHB13]. Each sub-connection behaves like a standard TCP. MPTCP

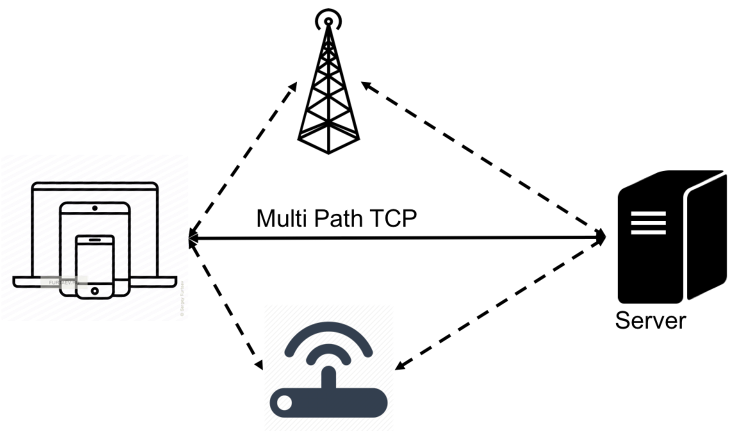

Figure 2.5: Multi Path TCP Illustration

requires sub-flow management and congestion control in sub-flows to work well [ZDZ+17]. As there are multiple connections between the server and the client, MPTCP needs to carefully schedule packets into different sub-flows. Each sub-flow is possibly in different network conditions. Take Figure 2.5 as an example. The sub-flow with Wi-Fi connection has lower Round Trip Time (RTT) compared with the RTT of sub-flows with cellular connection. It will bring the Head-of-Line (HOL) blocking issue [SK] as TCP guarantees in-order delivery. As shown in Figure 2.6, packets are transmitted through 3 paths and packet 1 is lost somehow and hence being retransmitted. Assuming that the buffer can only contain 9 packets, before packet 1 is received correctly by the receiver, other packets will be held in the receiver's buffers. If packets with larger sequence number

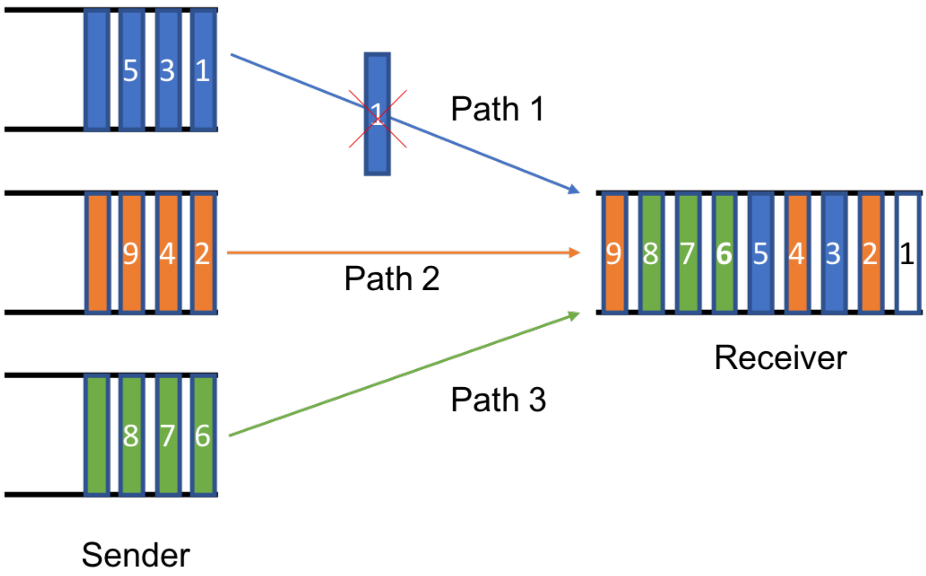

Figure 2.6: Head-of-Line Blocking

are scheduled into sub-flows with lower RTT, they have to wait in the buffer until the arrival of packets with smaller sequence number. HOL blocking will increase delay of packets and hurt delay sensitive applications. To solve HOL blocking issues, different schedulers can be used. A round robin (RR) scheduler equally shares packets into all sub-flows which can be used if all the sub-flows are homogeneous. If the sub-flows are heterogeneous, priority scheduler can be used to put more packets into the sub flow with lower RTT. Schedulers will be discussed in section 2.3.3. As there are many versions of TCP algorithms, different flows can use different kinds of TCP, which makes it hard to control congestion on the Internet.

2.3 Hop-by-Hop Congestion Control

In hop-by-hop algorithms, the intermediate network routers are responsible for detecting congestion and providing feedback to the sender indicating network conditions [Jac88]. End-to-end algorithms work well for traffic that is responsive to congestion e.g. TCP traffic. However, non-responsive traffic, e.g., UDP traffic, may still cause congestion since it does not react to congestion. Hence, hop-by-hop congestion control algorithms have drawn increasing attention [JWLR12][FODA14][PKTH16][GQC$^+$16].

2.3.1 Understanding Traffic Control (TC) within a Node

In the operating systems of a electronic device, traffic control is a set of technology, including queuing systems, scheduling schemes and etc. To control congestion within a device, the mechanism of how TC works is introduced. Taking Linux OS for example, figure 2.7 shows how packets go through the kernel stacks. When packets are received by

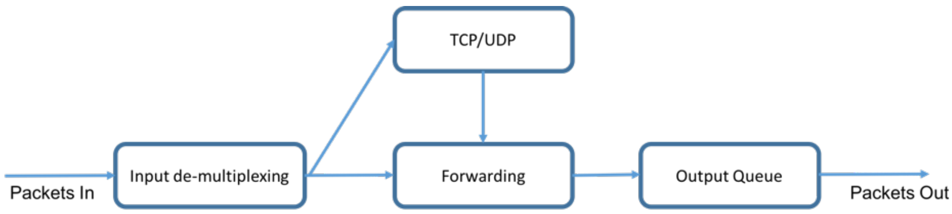

Figure 2.7: Traffic Flow Chart in Linux Kernel

Network Interface Card (NIC), a de-multiplexer is involved to decide whether a packet is for the local node or not. After de-multiplexing, forwarding decide where the packet is forwarded. The output queue model is where packets is waiting to be sent. Since the first AQM algorithm was proposed [FJ93] in 1993, a variety of components have been proposed to improve the performance of queue management algorithms. As shown in Figure 2.8, traffic control nowadays has multiple components which allow devices to give more specific control of the traffic. When packets arrive at the output queue, they can be firstly classified into different sub-queues where different AQMs can be applied. Then

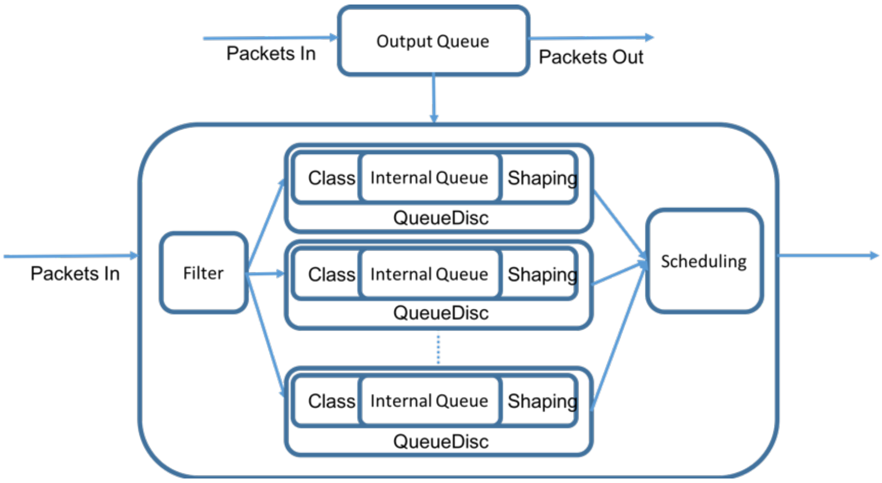

Figure 2.8: TC with different Components

traffic shapers, such as Token Bucket, will allocate bandwidth to each traffic. Finally, the scheduler can give priority to certain types of traffic or improve throughput fairness among different traffic flows.

2.3.2 AQM Algorithms

2.3.2.1 Random Early Drop

Random Early Drop (RED) [FJ93] works by monitoring the average queue length. As shown in Figure 2.9, RED has two thresholds: the minimum threshold (min_{th}) and the maximum threshold (max_{th}). When the average queue length is smaller than the min_{th}, no packets are dropped. When the average queue length is bigger than the max_{th}, each arriving packet is dropped. When the average queue size is between the two thresholds, each arriving packet is marked with probability p_a, where p_a is a function of the average queue length as shown in Figure 2.9. A general discard probability function is shown as Eq 2.8.

$$p_a = p_{max} * \frac{qlen_{avg} - min_{th}}{max_{th} - min_{th}} \tag{2.8}$$

where $qlen_{avg}$ is the moving average queue length; min_{th} is the minimum threshold and max_{th} is the maximum threshold; p_{max} is the maximum drop probability. To calculate

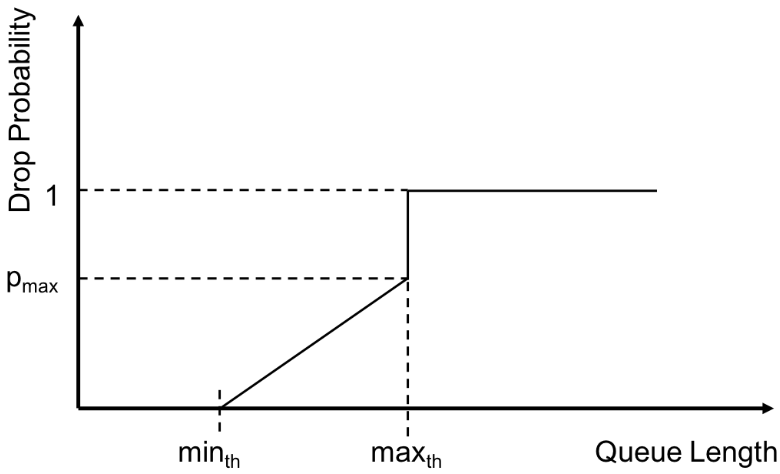

Figure 2.9: RED AQM

the moving average queue length, RED has a fixed parameter, called queue weight (w_q). Then, the average queue size is calculated by Eq 2.9 if the queue is not empty.

$$qlen_{avg} = (1 - w_q) * qlen_{avg} + w_q * qlen \tag{2.9}$$

If the queue is empty, the average queue length is calculated by Eq 2.10, where *time* is the current time and *q_time* is the start of the queue idle time.

$$qlen_{avg} = (1 - w_q)^{time - q\_time} * qlen_{avg} \tag{2.10}$$

As RED is easy to implement and effective, many variants of RED have been proposed such as Adaptive RED (ARED) [FGS+01], Modified RED (MRED) [DK13] and SmRED [PKTH16]. They will be discussed Section 2.4.

2.3.2.2 CoDel AQM

Aiming to solve the parameter setting issue and the Bufferbloat, a novel AQM was proposed in 2012, called CoDel. CoDel is different from prior AQMs [NJ12] as it does not use queue length or as the parameter. Instead, CoDel uses the waiting time of the first packet in the queue, i.e. HOL queuing delay. CoDel uses fewer parameters compared with prior AQMs. The threshold in CoDel is called target, which is the minimum HOL delay that the algorithm wants to keep. Another parameter is the interval to decide whether a packet needs to be dropped. CoDel monitors the sojourn/waiting time for each packet through the standing queue. The sojourn time is then compared to the target (5ms by default). If the sojourn time is above the target, it will trigger the dropping algorithm which will be introduced later. It also explains why the target is the minimum HOL delay as all the value above the target will trigger the dropping algorithm. Although queuing delay can be calculated by queue length, it depends on the bandwidth, which results in the issues of tuning parameters. By using HOL delay, the performance metric is directly related to user-perceived performance.

CoDel works in two phases. When a packet arrives at the buffer, CoDel enqueues the packet if the buffer is not full, and adds a timestamp to it. On depature of each packet, the timestamp is extracted and the sojourn time for the packet is obtained by calculating the difference between the current time with the time recorded. This is the first phase. The second phase is about making drop decisions. CoDel operates over a certain period, which is called *interval* (100 ms by default) as mentioned above. At the departure of the last packet during the interval, if the lowest queuing delay is above the target, then the packet is dropped and the next interval is shortened according to Eq. 2.11

$$interval = \frac{100}{\sqrt[2]{n}},\ n = 1, 2, 3, ... \tag{2.11}$$

where n is the sum of packets being dropped. It can be seen that the more packets are dropped, the shorter the interval becomes, and CoDel can drop packets more quickly. Once the sojourn time goes below the target value or there is no packet in the queue, the counter, n, will be reset to 0 and the interval is reset to 100ms. CoDel leaves the dropping mode.

CoDel is a promising method as it controls the latency directly to address the Bufferbloat problem; however, CoDel provides poor link utilization compared with RED and ARED [RAT13].

2.3.2.3 PIE AQM

Similar to most AQMs, PIE [PNP+13] drops packets randomly from the tail of the queue. The drop probability p_{drop} is updated every $\lambda = 30ms$ by Eq 2.12. $E[T]$ and $E[T]_{old}$ are the current and previous estimate of queuing delay. τ is the target as defined in CoDel. α is a tuning parameter that controls how deviation of current queuing delay from τ affects p_{drop}. β controls how the trend of the queuing delay affects p_{drop}, i.e., the queuing delay is increasing or decreasing.

$$p_{drop} \leftarrow p_{drop} + \alpha * (E[T] - \tau) + \beta * (E[T] - E[T]_{old}) \tag{2.12}$$

PIE consists of 3 components, as shown in Figure 2.10. The first part of PIE is the dropping algorithm, which drops a packet randomly on the arrival of a packet. The second part is the algorithm that updates the dropping probability periodically. And the third part is the algorithm that estimates the departure rate of packets. The dropping

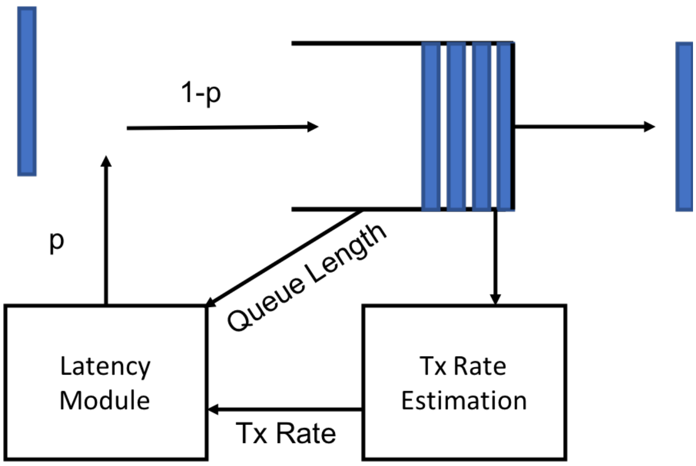

Figure 2.10: PIE AQM

probability of PIE is updated every 15ms by default, which is the *interval* in PIE. The calculation of the dropping probability is based on the dropping probability from the last period, the deviation of the current delay from the target value and the tendency of the latency, e.g. going up or down. In this way, PIE adjusts the dropping probability dynamically according to how severe the congestion is seen to be.

2.3.3 Scheduling Scheme

First-in-first-out (FIFO), also known as First-Come-Fist-Serve (FCFS), is the simplest scheduling scheme. In FIFO, traffic flows from different source or applications share the same buffer and as its name implies, it does not give any priority to any flows. Packets are stored in the queue according to the order of arrivals. The contents in the queue are depleted from the head to the tail of the queue. FIFO queue are widely deployed in the networks. However, delay-sensitive applications such as Voice over IP (VoIP), Video and gaming, requires higher level of QoS which allows packets pass the queue first. Non-FIFO scheduling enables this kind of operation. Non-FIFO scheduling uses multiple sub-queuing systems and by adopting different kinds of schedulers, it allows packets from different sub-queues to leave the queue first. The most common schedulers are discussed below.

2.3.3.1 Round Robin

The Round Robin (RR) scheduler is shown in Figure 2.11. A device with RR scheduler has separate queues for different traffic flows. Using this scheduler, the first packet in each queue takes turn to be served. The predefined value of the number being served in each turn is one. If the packets size of each traffic flow is different, then different amount of data is served during each cycle, which causes a fairness issue.

2.3.3.2 Weighted Round Robin

The Weighted Round Robin Scheduler is shown in figure 2.12, where w is the weighting factor. WRR is first proposed in [KSC91]. Static weights are assigned to different queues. It cycles through queues transmitting amount packet from each queue according to its weight. It was designed especially for ATM networks where the size of packet is fixed. In today's IP networks with variable size packets, the weighting factor needs to

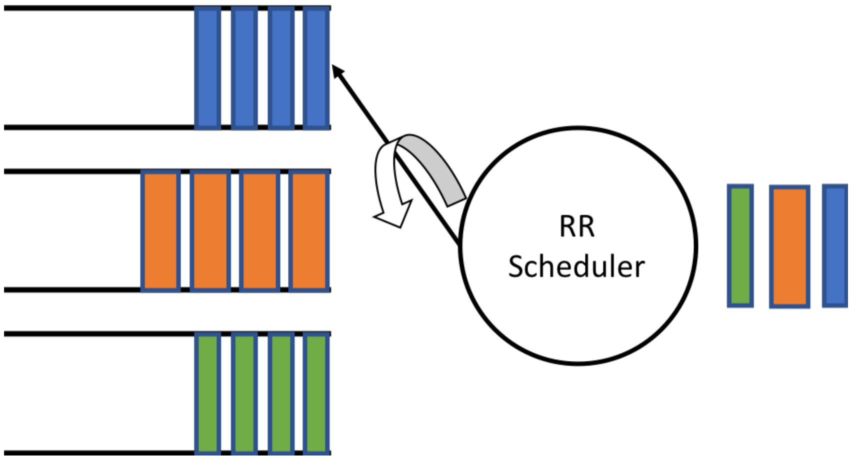

Figure 2.11: Round Robin Scheduler

be normalized.

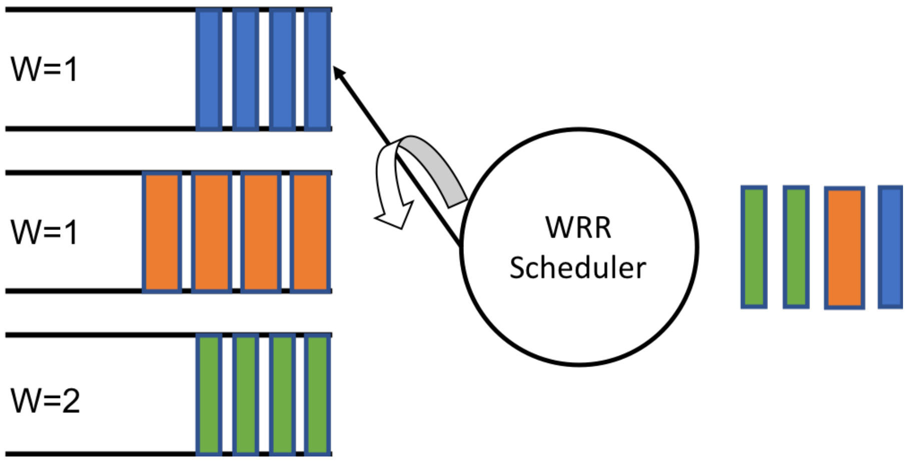

Figure 2.12: Weighted Round Robin Scheduler

2.3.3.3 Deficit Round Robin

To solve the drawbacks of RR scheduler, an enhanced round robin called Deficit Round Robin (DRR) [SV95]. A DRR scheduler is shown in Figure 2.13. As its name suggests, DRR improves fairness with the help of a counter called Deficit Counter. The counter has a predefined value and each queue is served. Or else the packets are queued and will be served in the next round. In the meantime the counter is increased by a quantum value. When a queue being served has no packets, the counter is reset to its initial value.

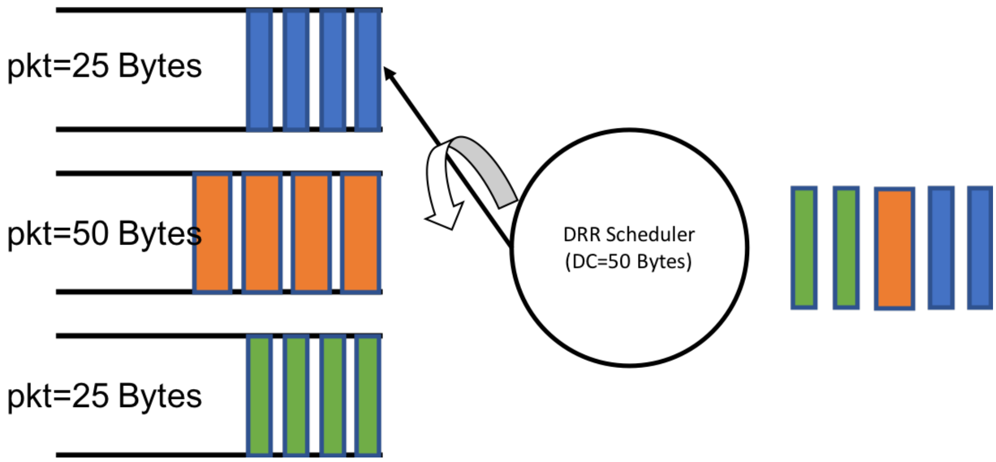

Figure 2.13: Deficit Round Robin Scheduler

2.4 Literature Review

This section introduces the effort researchers have made to fight against delays in a chronological order. This section reviews the state-of-the-art technologies for designing the mechanisms to reduce the delay of packets.

As previously mentioned, after the “congestion collapse” [Jac88], TCP is enhanced by Van Jackson with the AIMD algorithm. Five years later in 1993, RED [FJ93], the first AQM, was proposed. Giving a set of fine-tuning parameters, RED provides good performance, however with the fast advance of both hardware and software, new services and equipment keep emerging in the Internet, which makes it hard to tune RED. Inspired by RED, variants versions of AQM are proposed such as Fair RED (FRED) [LM97] and Adaptive RED (ARED) [FGS+01]. Compared with RED, FRED is more fair to different types of flows. FRED tracks the flows that have packets in the buffer and for each active flow, FRED has 2 parameter, min_q and max_q which are the minimum and maximum packets that can be buffered. ARED aims to solve the issue of RED that parameters need to be pre-configured. It monitors the average queue length to make the decision whether ARED should be aggressive or conservative, i.e., drop more or less packets. It uses AIMD mechanism to adjust the maximum drop probability, max_p, instead of using a static value. Users define the target delay and min_{th}, the minimum threshold, is chosen according to the target delay and the link bandwidth. A recent study [TSH+18] compares RED and ARED in the Benchmark Scenarios from RFC 7928 and the results show that ARED outperforms RED if there are no abrupt changes in traffic load. [RAT13] compares ARED with RED and CoDel in wired networks. Results in [RAT13] show that the advantage of ARED is less drop of packets but lower link utilization.

AQMs are initially proposed for wired networks. For wireless networks, the potential issue of over buffering in wireless link was first pointed out in [LRK+99] in 1999. The work traces packets at different layers and finds the mismatch between the load and the link capacity, which is caused by the improper setting of the buffer size. This work shows the negative effects of over-sized buffers such as inflated RTT and degraded user experience of web browsing which may take several seconds due to the standing queue. The over buffering issue becomes severe as the link capacity of cellular network increases rapidly with the advance of transmission technology, as larger buffer are required in the networks to absorb the burst of traffic.

In [SCHW01], a Markov model for the Rayleigh fading channel and the link layer is built to evaluate the buffer overflow issue. The work takes the standing queue into consideration and reveal the relation of TCP sending rate, queue length and queuing delay.

In [SLMP03], a new queue management technology, called Packet Discard Prevention Counter (PDPC) is proposed by Ericsson and it is accepted as the default queue/buffer management by 3GPP. Compared with RED, it exploits the link's capacity to set the minimum threshold of the instantaneous queue size rather than the average queue size. The PDPC algorithm has a counter and for each packet drop will initialize the counter with the value of n. When the queue length exceeds the minimum queue length, one single packet is dropped and the counter is triggered. For each arriving packet the counter will be down counted and as long as the counter is with positive value and the queue length is below maximum threshold, the arriving packets will be accepted. If the queue is empty, the counter is set to be 0 to recover the discard policy. By doing this, PDPC prevents losing multiple packets from certain traffic flow and ensures the congestion is notified by dropping a packet immediately after the queue builds up.

Research work [AGG+13] investigates the interaction between different TCP versions and buffering used in cellular networks. Their results are convincing since the measurements are performed in the commercial 3G, 3.5G and 4G cellular networks with the mixture of long and short TCP flows. The TCP versions tested are CUBIC, New Reno and WestWood+. They have claimed that the excessive buffers occupied by long flows will obviously increase the delay of short flows.

Research work [SJS14] analyses the Bufferbloat in IEEE 802.11n wireless networks, which points out that the excessive buffers occupied by long flows, such as transferring files by FTP, can lead to the RTT exceeding 4.5 seconds. This work proposes MAC-layer protocol to alleviate the issue but more measurements are required for real time flows, such as VoIP. Another research work [HJKT+17] explores the 802.11 performance anomaly. Based on the conclusion from previous work [SJS16] that neither deploying AQM or reducing the buffer size to Wi-Fi interface can provide same delay reduction as for wired links, it points out that the queuing in lower layers is the reason for the limited performance of the mentioned algorithms. The work makes modifications to the queuing protocol stacks in the Access Point (AP) as it is easier to deploy and it also takes the aggregation mechanism into consideration. The algorithm it proposed contains three part. The First part is the AQM at the enqueue stage. If the queue limitation is reached, the arriving packet is dropped. Otherwise, the arriving packets are hashed into a sub queue according to the DiffServ markings, Traffic Identifiers (TIDs). The second part is at the dequeue stage. When a sub queue needs to dequeue a packet, a DRR scheduler is applied to the queue. The third part is the scheduling algorithm which focus on airtime fairness. Different from Jain's Fair Index which is based on throughput, airtime fairness is based on time. In wired network, the airtime is equal to transmit same amount of data for the links with same speed as the connection is stable. However, for wireless access networks, the connection between each station and the AP are with variable link speed due to variable distance, interference from other devices and the multipath fading. The scheduling algorithm in this work aims to achieve airtime fairness among different connections. The proposed algorithms in this research work can reduce the latency however it sacrifice the throughput of the users with poor connection.

Another research work in Ericsson [Tan09] focus on the uplink of LTE and proposed a delay based AQMs specifically for LTE scenario. The proposed AQM is deployed at the Radio Link Layer (RLC) layer of eNode Base stations (eNB), named Receiver-AQM. The receiver refers to the base station and it aims to reduce the queuing delay in uplink direction. Compared with AQMs deployed in each UE (transmitter), Receiver-AQM is easier to deploy as it only requires the eNB to install the AQM.

In 2009, Dave Reed reported large RTT with low packet loss rate in 3G networks during daytime in the end-to-end mailing list. In the night, the RTT became much short as the number of flows reduced. The large RTT was believed to be caused by excessive buffering. However, it didn't draw too much attention until in 2011, Bufferbloat [CJWG11] in wired networks was pointed out. The author noticed a latency up to 1.2 seconds in his home network. The high latency is captured in both cable modems and fibre networks. It reveals the fact that abuse of buffer will cause high latency. This work indicate that essence of Bufferbloat is the latency under load. Bufferbloat will not be noticed if researchers only test the latency when the link is idle. As is pointed out by [Get11], the side effect of Bufferbloat appears every time the link is saturated. Network services, such as DNS, will fail due to the large latency induced by Bufferbloat. High frequency traders treasure 1ms advantage to their competitors. Web browsing will be annoying as contents need seconds to be loaded. [CJWG11] also points out that although there are AQM algorithms such as RED, few networks are running with them.

The abuse of buffer is actually everywhere. TCP needs to keep unacknowledged packets in the buffer. Buffering is necessary to absorb burst of traffic and even the device drivers also have large buffers. If looking inside the operating systems, Bufferbloat will be found at multiple layers, as pointed out by [CJWG11]. Hence, there is no single solution to solve the issue. To mitigate Bufferbloat, it needs the cooperation of Internet Service

Providers (ISPs), application vendors and device manufactures. One of the most effective way to alleviate Bufferbloat is deploying AQM in home routers. RED, mentioned previously, is a good starting point. RED makes the drop decision according to the average queue length. However, as the packets size and link capacity differs from each other, average queue size cannot give enough information to predict the latency.

In 2012, CoDel, mention in section 2.3.2.2, was proposed to solve the issue in wired networks. CoDel uses a local minimum queue as a more reliable and accurate measure of standing queue and hence is able to use a single state variable to show whether the latency is below or above the *target* value. As is mentioned in the previous section, *target* is the minimum delay that CoDel wants to keep. CoDel labels a queue "good" or "bad". A "good" queue accommodates incoming packets and turns bursty arrivals into smooth departures. While a "bad" queue accumulates packets and create unnecessary delay.

[KLM14] compares the performance of CoDel and RED in the most common topology as shown in Figure 2.14, where A and B are the servers and E and F are the client. C and D are routers and the bottleneck is the link between C and D. This work simulates

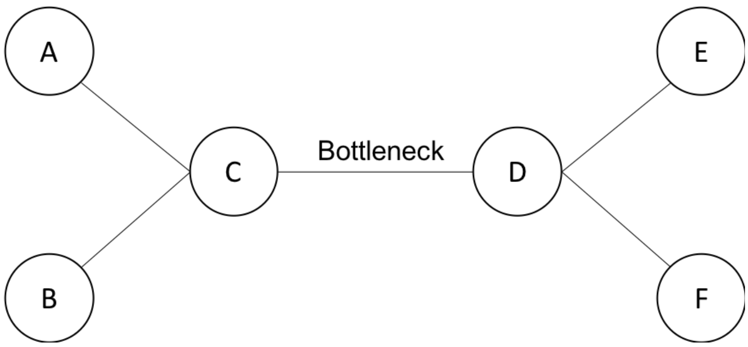

Figure 2.14: Common Topology

the process of downloading a file of 10MB and points out that CoDel outperforms RED in reducing the latency. However the downloading process with CoDel costs 42% longer than that with RED. Hence, RED can also be considered as a candidate to mitigate Bufferbloat.

[Cer14] confirms that Bufferbloat is a performance challenge at the edges of the Internet. [RAT13] evaluates the effectiveness of CoDel with different versions of TCP in wired networks. In the simulation, the bottleneck bandwidth and number of TCP flows are varied. CoDel is compared with RED and ARED. The results show that CoDel achieves high link utilization and reduces the queuing delay. This work also points out that the effectiveness of CoDel in wireless networks remains to be evaluated.

PIE, mentioned in the previous section, is proposed after CoDel. [CR15] evaluates the performance of Compound TCP with PIE. It uses a fluid model of TCP and points out PIE is unstable when the queuing delay gets larger. [Whi15] evaluates PIE in cable networks. Compared with DropTail queue, PIE reduces the upstream latency by hundreds of milliseconds which can significantly improve the user experience of web browsing, VoIP and online games.

Flow Queue (FQ) CoDel [HJMT+14] is proposed then providing isolation among different flows. FQ-CoDel uses a hashing algorithm to distinguish packet from different flows. The hashing is based on a 5-tuple of source IP, destination IP, source port, destination port and protocol number. Byte-based DRR scheduler is used to keep fairness among all the sub queues. CoDel algorithm is applied in each sub queue. [KKFR15] evaluated the performance of CoDel and FQ-CoDel in capacity-limited networks with large RTT. The results show that both CoDel and FQ-CoDel have difficulties in reducing latency properly with default configurations. In rural broadband scenario, CoDel and FQ-CoDel with default settings drop too many packets resulting in a very low link utilization. This work proposes a new set of parameters that can provide higher link utilization while maintain a lower queuing delay. Although CoDel is claimed to be parameterless and insensitive to link capacities and RTTs [NJ12], it is not the case according to the test of this work.

[JK17] proposes a new AQM that can handle rapid changes of load by predicting the load in the near feature. Compared with CoDel and PIE, the proposed AQM provides shorter flow completion time. The work also points out that CoDel and PIE cannot handle queues containing packets with small RTTs.

[ASAB17] evaluates the Low Extra Delay background Transport (LEDBAT) performance through bottlenecks with PIE, FQ-CoDel and FQ-PIE. LEDBAT [SHIK12] stands for Low Extra Delay Background Transport. It is based on UDP and reacts to both delay and loss. LEDBAT aims to help bulk transfer applications, such as software updates. LEDBAT utilizes available bandwidth with a delay threshold of 100 ms. Default setting of CoDel allows a minimum delay of 5 ms which is much lower than 100 ms. Results in [ASAB17] show that AQM algorithms lead to poor link utilization of LEDBAT.

[KNAB17] explores the effective of AQM algorithms on Internet of Things (IoT) application flows in home broadband networks. FQ-CoDel, FQ-PIE and PIE are tested in this work. With the emergency of smart devices, such as sensors and health monitors, IoT application flows are quite common in home broadband networks. IoT flows [PA14] are often require low bandwidth as the packet size is small, but they often have strict QoS requirements. For example, dropping packets from a health monitor device is very dangerous as important information is possibly dropped. Competition with other flows over a shared bottleneck will hurt IoT flows as single-queue AQM algorithm will distinguish packets from different flows. [KNAB17] points out that FQ-AQM algorithms provide good flow isolation and hence protect IoT flows.

[AOAL17] makes modification to the famous RED algorithm. As different classes of flows have different QoS requirements, this work proposes a Fair Weighted Multi-Level Random Early Detection (FWMRED). The definition of fair weighting factor is defined according to the quality of server, in terms of delay and bandwidth. Multiple-Level RED means a RED algorithm with adaptive parameters which are decided by average queue length. The results from this work show that the proposed algorithm improves bandwidth fairness and reduce the latency compared with RED and ARED. However, RED and ARED were proposed quite a long time ago, hence the proposed algorithm needs to be compared with new emerging algorithms such as CoDel and PIE.

[KRB$^+$17] explores the operating range of CoDel and PIE, in terms of RTT, bottleneck bandwidth and congestion level. PIE allows more queuing delay with default settings, but both CoDel and PIE result in poor performance in data center and rural broadband networks. Although CoDel claims parameterless and try to be adaptive, manual tuning is unavoidable.

As video streaming traffic accounts for a large proportion of Internet traffic, extensive research work has been done on optimization on video delivery technology. [JDSL14] focus on the coding scheme when the content is dominated by different colours. [LCTA12] proposed a network management algorithm to improving the quality of mobile IPTV. It takes advantage of network performance metrics such as bandwidth, jitter, delay and packet loss to decide the coding schemes of the delivered video. [SVS14] proposed a QoE-driven framework for RTP based video due to the limitation of available QoE model. Although QoE on DASH video has drawn the attention of researchers, [HSVT12] points out the challenge in developing a proper QoE model. The retransmission mechanism in TCP makes it complex when it comes to packet loss and other network impairments. User activities such as Pause, Resume, Switching among different resolutions and time shifting when watching the video will influence the QoE [MCLC11]. Some research work point out that initial delay, stalling and rate changing are the main factors in evaluating the QoE of video content [SCW+16][HSBP13].

- Initial delay refers to the time between content request and start of the actual playout of the content. The QoE of live steams is more sensitive to initial delay.
- Stalling refers to the phenomenon when the frame freezes which is due to lack of contents in the buffer. Playback will be resumed when the buffer is refilled.
- Rate changing refers to the phenomenon that the resolution of the video changes during playback. It is normally due to the change of network conditions such as the bandwidth. Rate changes when congestion happens in the network, which can avoid stalling.

[CJWG11] points out one video stream from YouTube and one video stream from Netflix can completely saturate the buffer of a home router. [MVSA13] proposed a Smooth Adaptive Bit RatE (SABRE) algorithm to mitigate the Bufferbloat issue in residential networks. This work tests the performance of RED and points out that RED outperforms DropTail but the queuing delay still remains hundreds of milliseconds. According to the results from the work, it is because the threshold of RED is not properly set. RED allows too many packets in the buffer before it starts to drop a packet. The proposed SABRE algorithm monitors the socket buffer level at the client and it changes the value of RWND to reduce the number of packets sent by the erver. If the buffer level is reaches 75%, SABRE will reduce the RWND and if the buffer level is lower than 75%, it will increase the RWND. However, the results in this work only compares DropTail queue after it points out the drawbacks of RED.

With the deployment of modern AQMs in home router, the interaction between DASH and AQMs is complex as DASH will try to adapt to the networks conditions and AQMs will trigger the adaptive nature of DASH by dropping incoming packets. [KAB16] studies the impact of AQMs on DASH delivery. Results shows that a flow queuing scheduling scheme with PIE algorithm in the sub queues gives the best throughput. [AG16] improves DASH performance by using multiple TCP connections and measures the occupancy of the buffer in the home router. It can be seen that the queue keeps full which causes the Bufferbloat issue. [KA17] proposes a chunklets algorithm to optimize DASH over AQM enabled gateways. The essence of chunklets is to use parallel TCP connections for DASH content which makes DASH more aggressive in competing for bandwidth with other flows.

Lots of work have been done in wired networks. Bufferbloat in cellular networks has also drawn the attentions from both researches and device manufactures. Compared with Wi-Fi and wired network, the protocol stack of cellular networks is different to those of wired and Wi-Fi networks. Data transmitted between the base station and the UE is carried by a virtual concept, "bearer" [CPG+13], which means each UE has a dedicated buffer for communication in the base station. As shown in Figure.2.15, the RLC Layer and PDCP Layer, where the queuing of packets for different UE happens. Each UE, when connected to an eNB, will be allocated a dedicated PDCP and RLC buffer for downlink data transmission. RLC layer has three different transmission modes, Transient Mode (TM), Unacknowledged Mode (UM) and Acknowledged Mode (AM) mode.

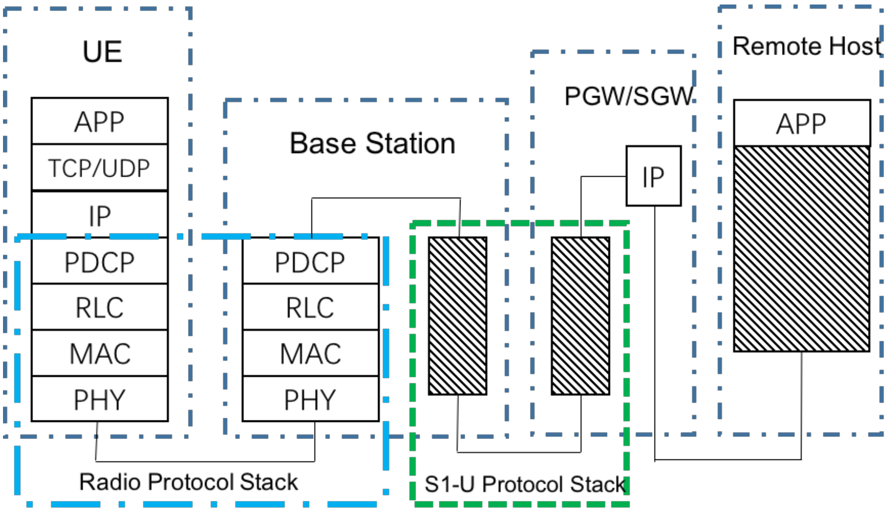

Figure 2.15: Protocol Stacks of Cellular Network

- TM Mode: it does not make any modifications to data, which means no headers are added or removed, no segmentations created and no aggregations. And it does not require any ACK/NACK from the receiver.
- UM Mode: it is similar to TM mode but the difference is that UM mode has its own headers and it can segment or concatenate data.
- AM Mode: it is used to guarantee reliable transmission which requires ACK/NACK from the receiver. It can also segment or concatenate data and has its own headers. Moreover, it will make a copy of the transmission buffer for a possible retransmission.

TM mode and AM mode are used in the control plane and UM mode is used for transmitting data. Our algorithm is deployed in the UM buffers so that it does not affect the control and signaling messages. An AQM tuned for cellular networks is proposed in [PKTH16] and this is a variation of RED and it is implemented in the RLC layer. The authors change the control function from linear function to non-linear and simulation results show that it outperforms RED from the aspect of end-to-end average delay.

However, it is based on RED and cannot solve the tuning issue as the length of the queue is not directly related to delay.

Queue aware scheduling, such as reported in [AEN16], gives priority to real time (RT) traffic. However, most of the traffic in cellular networks are RT traffic, so giving priority is not easy and it can be conflict with the priority settings of 3GPP specifications.

[JLW+12] and [JWLR12] exploit the Bufferbloat phenomenon in 3G/4G networks. Both of the works are taken in real cellular networks which are operated by the 4 main carriers in the USA. [JLW+12] confirms the Bufferbloat issue is in the cellular part instead of the Internet part. They also point out the abnormal behaviour of TCP in smart phones as shown in Figure 2.16. It can be seen from the figure that TCP in iPhone and Android Phone behave differently compared with that in Windows Phones. Knowing the TCP

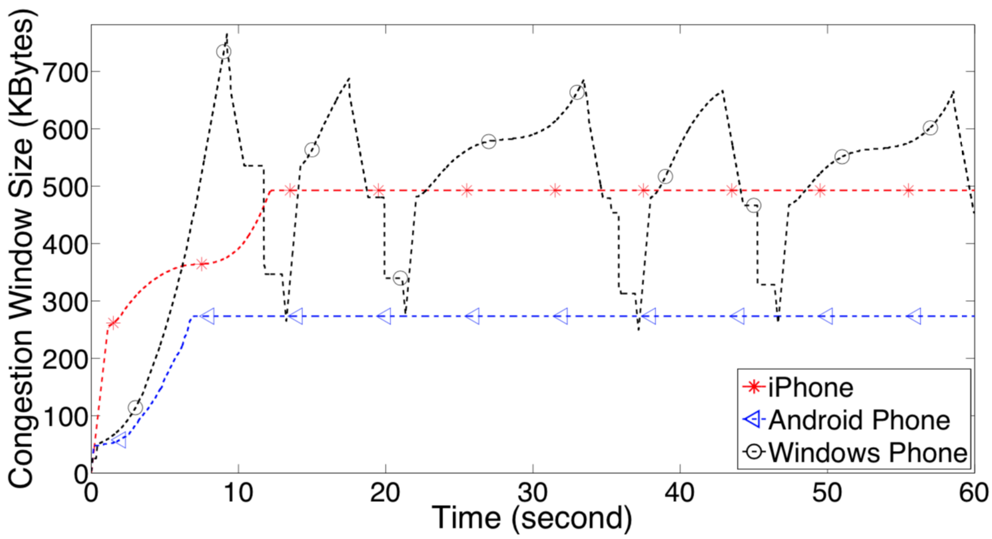

Figure 2.16: Abnormal TCP Behavior (Figure 7 in [JLW+12])

congestion control mechanism, [JLW+12] reveals the reason why TCP in iPhone and Android can give a flat CWND. To prevent the side-effects of large buffers, ISP gives hard limit of the Receiver Window (RWND) size over the air interface to stop the packets accumulating in the buffer, hence stops the inflations of TCP flows. Such limit of RWND will result in the degradation of throughput for long-lived TCP flows. Then, a light-weight AQM called Dynamic Receive Window Adjustment (DRWA) [JWLR12] is proposed which makes modifications to the size of RWND. It uses the timestamp option in TCP to get accurate estimation of the RTT and the estimation of RTT is used to control the RWND size. The proposed algorithm reduces latency at the sacrifice of throughput of high speed links with small RTT. However, it improves the throughput iofn high speed links with large RTT.

Research work [ZMZ+17] evaluates the performance of CoDel and DRWA in millimeter wave (mmWave) links. As there are massive spectrums available in mmWave, the bandwidth is much higher than existing networks. Although using mmWave can significantly increase the bandwidth, there are also side effects. MmWave are easily blocked or attenuated by building materials such as brick [PK11][ZMS+13][AR04][ASC08], and even a human body can cause 35dB attenuation [LSCP12]. Hence, mmWave has very high data rate but high variability. This work suggests that a cross layer design that MAC and RLC layer can cooperate with each other is necessary.

In the Packet Data Convergence Protocol (PDCP) buffer in each UE, there is a timer to control whether to drop a packet or not. When the PDCP buffer receives a packet from higher layers, the discard timer is started. If the packet is not sent by the UE when the timer expires, the packet will be dropped. To dig out the potential of the default queue management, [TWTGG11] tries to optimize the performance of the discard timer by figuring out the proper setting. It does a series of the test however fixed settings cannot meet the requirement of today's complicated mixture of traffic. the optimized performance under one circumstance is sub-optimized in another scenario.

A research work in Nokia [SV15] also contributes in managing the buffer at PDCP layer. They propose two mechanisms. One is called TCP Packet Pacing (TPP) and another is Advertised Window (AWND) management. The TPP has three components, first is to measure the real depletion rate of the PDCP buffer and then a ACK shaper shapes the rate of ACKs and hence the rate of ACK matches the depletion rate of TCP. Seconds is to adopts a virtual discard mechanism at the reverse ACK flow. It will insert duplicated ACKs (identical copies of already transmitted ACKs) in the reverse ACK flow to give the sender a fake signal of congestion. The third component is to deploy fair queuing at the PDCP buffer to further improve the fairness among different flows. The AWND management also calculates the depletion rate of the PDCP buffer and uses the reverse ACK flow to feedback the AWND information to the sender. Research papers [JWLR16] and [IJLB16] try to control the traffic sending rate by making modification on the congestion RWND. Both of the works take advantage of (RTT) and aim to solve Bufferbloat in cellular networks. [JWLR16] is primarily based on the estimated RTT and the minimum RTT value. The estimated RTT is the average of RTT value from all the samples of RTT and if the estimated RTT is larger than the minimum RTT, the RWND will be reduced. Work [IJLB16] controls RWND by monitoring queue states. The queue state is estimated using the difference between the minimum RTT value and the real RTT value. It assumes that the minimum RTT value is the RTT when there is no queue in the buffer. However, when there is no queue in the buffer, the dynamic nature of wireless channel and the number of UEs competing for the bandwidth will also affect RTT. Additionally, the calculation of RWND in their work is a function of the dropping function of AQM deployed in the router and different AQMs may behave very differently. [PKTH16] proposed a variation of RED specially for LTE networks. Based on the conclusion from [PZR02], they modified the drop function of RED. When the traffic load is low and the link bandwidth is not fully utilized, a small drop probability is needed to prevent further reducing the link utilization. When the traffic load is high and the link is fully utilized, a large drop probability is need to prevent the latency. The proposed smart RED algorithm uses different functions to calculate the dropping probability when the traffic load changes. This work also emphases that a cross layer design is needed to make AQM in LTE scenario more practical. This work compares the proposed algorithm with RED which is proposed long time ago. Comparison with recent algorithms need to be done.

Work [AEN16] focus on the real time application in LTE scenario. It classified applications into categories, namely real time (RT) applications, such as VoIP and video, and non-real time (nRT) applications such as web browsing and file transfer. It then proposes a two-level scheduling scheme which is called Rate-Controlled Priority Queuing

Discipline (RCPQD). The proposed algorithm is based on the deficit weighted round robin (DWRR). The RT traffic are classified into high priority queues and the nRT traffic are classified into low priority queues. The scheduler will keep processing the high priority queues as long as they have packets in the queue. The low priority queues will be processed if the high priority queues are empty. To keep the fairness, the high priority queues has a counter. When certain number of packets from high priority queues are processed, low priority queues will be served for a period of time which is decided by the delay requirement of the RT traffic.

A recent study [BNPF17] evaluates two promising AQM algorithms, i.e. CoDel and PIE, in cellular networks by simulation. The results show that CoDel and PIE lower the link utilization and induce too many losses of packets when the bottleneck link is of high variation. This work proposes a new metric, queue balance, for designing effective congestion control in cellular networks. Queue balance is the difference between arrival and departure rates f packets, and there is work [ALLY01] trying to find the mismatch between the arrival and departure rate. The effective of the congestion control using queue balance still remains to be seen.

With the prevalent of distributed systems, in the future 5G network, traditional base stations are decoupled into two components. One is called distributed radio heads (RRHs) which are with radio frequency functions supporting high capacity in hot spots, and the baseband units (BBUs) clustered as a BBU pool providing large-scale collaborative processing cooperative radio resource allocation (CRRA) in a centralized location. The BBU pool communicates with RRHs via common public radio interface (CPRI) protocol. When it refers to resource allocation, the most common assumption is the infinite queue backlogs and stationary channel conditions such as [BHL+14] and [KWL15], as pointed out in [PSL+16]. However, the time-varying radio channel and standing queue in the devices have significant effect to the performance. Parameters from the physical layer play an important role in cellular communications, e.g. CQI which indicates the quality of the wireless channels for data transmission. CQI is measured at the UE side and reported to eNB using the Public Uplink Control Channel (PUCCH). The Modulation and Coding Scheme is selected at eNB according to the CQI reported by a specific user which reduce the Bit Error Rate (BER) [KHH+12]. CQI is also used in the data link layer as a parameter in scheduling schemes. More resource blocks are allocated as a compensation of bad channel quality so as to achieve fair throughput and low delay. It can also be used in another way such as maximizing the overall throughput by giving priority to UEs with good channel quality. Research work [ZNSH12] exploits the average arrival rate, service rate and delay via packet flow model of queue theory and the numerical results in [CL13] show the trade off between the Channel State Information (CSI) and the average queuing delay. It can be seen that research work at MAC layer has begun to consider the effect of queuing in cellular networks. In conclusion, quite a lot receiver-side mechanism focusing on the TCP flows, however many real time application traffic does not use TCP and such traffic are often with high QoE requirement. The existing AQMs or just modifications to TCP are not enough.

More and more functions are integrated on mobile devices and with the rapid increase of smart devices such as smart watches, monitors, sensors together with phones and tablets, mobile traffic will be inflated obviously. Cellular networks are becoming an increasingly important technology to access the Internet.

As pointed out by [AGG+13], wireless networks have become an integral part of day-to-day-life and suffer the most from performance issues, while there minimal work done so far. Even though a whole variety of AQMs and variation exist, most of these focus on wired networks such as adaptive RED (ARED) [SJS16], flow-queue CoDel (fq-CoDel) [HJMT+14]. Evaluations and tests have also been done in wireless networks such as in [HJHB15], [JBT14] and [SJS14] but none of them considers the effect of wireless features. Existing AQMs are primarily based on the status of the queue (queue size or the delay each packet suffers in the queue). Hence, it is truly significant and timely to focus on buffer de-bloating in wireless environments.

2.5 Summary

This chapter summarize existing technology against Bufferbloat phenomenon. TCP and its variants try to control congestion at the end point by adjusting the sending rate. AQMs control congestion at the point congestion happens by pro-actively dropping packets before the queue is full. Table 2-A summarize the state of the arts. It is worth mentioning that it is not possible to improve all the metrics (goodput, delay, loss and fairness) at the same time. This thesis focuses on reducing queuing delay by AQMs and maintain similar throughput and fairness among different users. The link utilization and resource allocation are beyond the thesis.

Table 2-A: Summary of State of the Arts

Methods / Networks Scenarios	Modifications of TCP	AQMs
Wired	[KA17], [Pos03], [SHIK12], [ASAB17], [CR15], [Cer14], [SK], [FRHB13], [ZDZ+17], [PFAB14], [TSZS06], [BOP94], [Nag84], [Jac88], [HP92], [BP95], [VS94], [HRX08]	[SVM16], [JK14], [KAB16], [MVSA13], [ALLY01], [PZR02], [KRB+17], [AOAL17], [Kli17], [JK17], [IKVF15], [KKFR15], [RAT13], [KLM14], [AL18], [TT17], [Cha17], [LZGS84], [TSH+18], [FKSS01], [FSKS02], [LM97], [KSC91], [DKS89], [SV95], [ACA96], [FJ93], [NJ12], [FGS+01], [HK04], [DK13], [FHXC15], [HJMT+14], [Whi15]
Wireless	[AG16], [LL15], [CBHB16], [ZMZ+16], [CT14], [FODA14], [HXT+10], [ZMZ+16], [ZMZ+17], [CGM+02], [JWLR12], [SCHW01], [WWZH11], [JWLR12], [CT14], [CLG+13], [AGG+13], [SV15], [IJLB16], [JWLR16]	[KNAB17], [BNPF17], [FSKN17], [YLLL18], [FODA14], [GQC+16], [HKT+17], [GPKC17], [TGWTG11], [AC17], [CL13], [TWTGG11], [HJKT+17], [SLMP03], [JBT14], [SJS14], [Tan09], [HJHB15], [PKTH16], [AEN16], [LHG15]

Chapter 3

Performance Evaluation of Active Queue Management on Wi-Fi Access Networks

3.1 Introduction

Wireless access networks are divided into two categories according to the technology used. Wi-Fi is commonly seen at home or a limited size of area (e.g.within a building). Wi-Fi uses a router to provide Internet access to the devices connected to it. Cellular networks are usually used in a larger area (e.g. within a city). A base station has more powerful antennas compared with a router and hence covers lager areas. Wi-Fi is an important way for mobile users to connect to the Internet. Wi-Fi is more convenient for users to join the networks compared with wired networks. And compared with cellular networks, Wi-Fi provides stable and fast connection with lower cost when downloading files or making VoIP call compared with cellular networks and it helps to save the energy of you devices. Traditional AQMs design for wired networks are tested in wired connection and have not been fully study in Wi-Fi networks. To study the performance of AQMs in wireless access networks, this will work start with evaluating existing AQMs in Wi-Fi networks. Evaluations of new proposed AQMs, i.e., CoDel and PIE, have been done in wired networks in [RAT13][JK14][SVM16]. The results of these research works show that comparing with PIE, RED and variants of RED, CoDel maintains lowers queuing delay and the link utilization is higher. Meanwhile, CoDel has fewer parameters compared with PIE and RED. Therefore, CoDel is chosen as the baseline through this research work. In this chapter, we evaluate CoDel under heavy traffic load with different traffic types. We focus on the delay and the drop probability. The load of the traffic, number of different traffic flows and the effect of different parameter of CoDel are tested to give a insight into the performance of CoDel.

3.2 Background of Traffic Pattern

VoIP traffic is used in Chapter 3, Chapter 4 and Chapter 5, which is generated using ON-OFF traffic generator. This section introduces the the principle and the validation of VoIP traffic.

ON-OFF model is used to model traffic in the Internet [ML97]. As shown in Figure 3.1, ON-OFF traffic source sends packets with a static rate during “ON” period and keep silence during “OFF” period. The time of “ON” period and “OFF” period can be fixed values or random values with certain distribution. The mean arrival rate is calculated

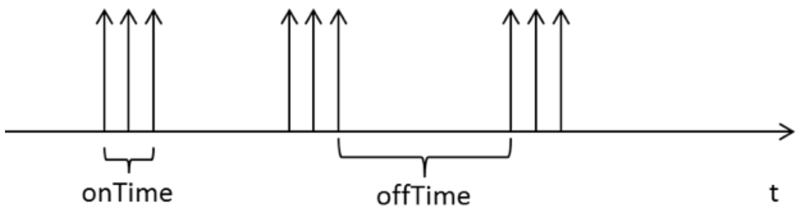

Figure 3.1: ON-OFF Traffic Source.

by Eq 3.1, where λ_{on} is the packet generate rate during “ON” period. T_{on} and T_{off} are the length of "ON" period and "OFF" period.

$$\lambda = \frac{\lambda_{on} T_{on}}{T_{on} + T_{off}} \tag{3.1}$$

The network load of a single ON-OFF source is given by Eq 3.2, where C is the bandwidth of the bottleneck link.

$$\rho = \frac{\lambda}{\mu} = \frac{\lambda_{on} T_{on}}{C(T_{on} + T_o ff)} \tag{3.2}$$

The queue length for bursty traffic consists of two parts. The first part is packet-scale queuing, which is Poisson distributed, and the second part is bursty scale queuing, which is exponentially distributed, as shown in Figure 3.2. The schematic picture of the

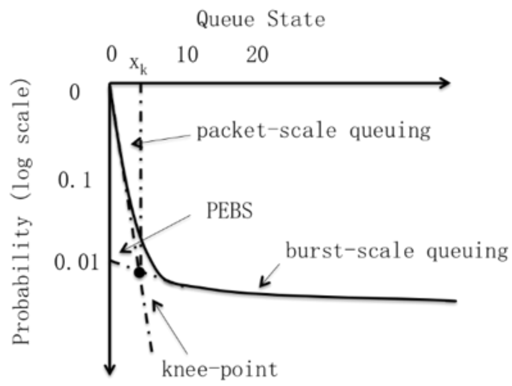

Figure 3.2: Bursty Traffic Queuing

distribution of packet-scale queuing is shown in Figure 3.3. According to [PS01], the

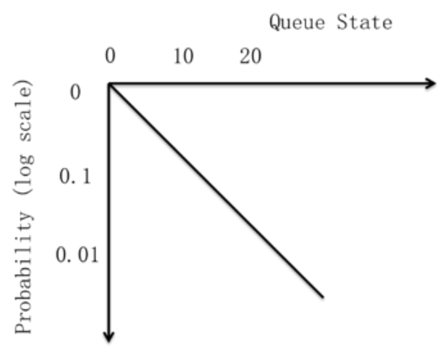

Figure 3.3: Packet Traffic Queuing

probability of n packets in the queuing system is given as Eq 3.3 for Poisson traffic.

$$P(n) = \begin{cases} 1-\rho, & n = 0 \\ (1-\rho)(e^{\rho}-1), & n = 1 \\ (1-\rho)\sum_{i=1}^{n} \frac{(i\rho+n-i)e^{i\rho}(i\rho)^{n-i+1}(-1)^{n-i}}{(n-1)!} & n > 1 \end{cases} \tag{3.3}$$

For bursty traffic, the average queue length in the packet-scale queuing is given by Eq 3.4, where C_p is packet-scale decay constant. η is packet-scale decay rate and it is given by Eq 3.5.

$$q(x) = C_p * \eta_n^p \tag{3.4}$$

$$\eta = \frac{\rho e^{\rho} - e^{\rho} - \rho^2 + \rho + e^{-\rho}}{\rho - 1 + e^{-\rho}} \tag{3.5}$$

For burst-scale queuing, the average queue length is given in Eq 3.6, where η_b is the burst-scale decay rate and can be approximated as Eq 3.7. T_{on} is the on period in the ON-OFF application to simulate the burst traffic; ρ is the utilization of the queue or the load of the link. μ is the service rate of the queue and λ_{on} is the service rate during the on period.

$$q(x) = C_b * \eta_b^x \tag{3.6}$$

$$\eta_b = 1 - (1 + \frac{\rho T_{on} \lambda_{on}^2}{\mu(1-\rho)^2})^{-1} \tag{3.7}$$

Figure 3.4 shows an arrival process of a random distribution. If the inter-arrival times of

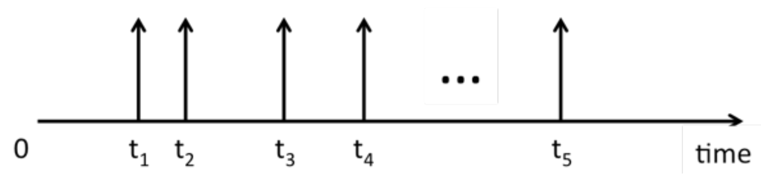

Figure 3.4: Poisson Arrival Process Illustration.

a process (such as $t_2 - t_1$, $t_3 - t_2$ and $t_4 - t_3$) are independent and identically distributed (IID) and exponential distributed, then it is called Poisson process. Poisson processes are often used to model the number of arrivals over a given time interval, e.g. number of packets arriving at a queue. Traffic following a Poisson process is called Poisson traffic. Thus, the probability of k packets arriving during interval t is given by Eq 3.8.

$$P_t = \frac{(\lambda t)^k}{k!} e^{-\lambda t} \tag{3.8}$$

The well-known property of Poisson process is that the merging of multiple Poisson process is also a Poisson process. The number of arrivals in a unit time is called the mean arrival rate and for the new merging Poisson process, the mean arrival rate is given by Eq 3.9, where λ_i is the arrival rate for the i_{th} Poisson process. This property is used throughout the work.

$$\lambda = \sum_{i=1}^{N} \lambda_i \tag{3.9}$$

3.2.1 Validation of Traffic Pattern

3.2.1.1 Simulation Set Up in NS3

The topology used in single source validation is shown in Figure 3.5. Packets are generated from the server to the client using ON-OFF application. The bandwidth from the server to the router is 100 Mbps and the link between the router to client is 10 Mbps. The queue in the router is checked periodically (every 0.01 seconds) and the queue state

Figure 3.5: Topology for single source validation.

is recorded. The parameter used in the ON-OFF application is as shown in Table 3-A. The topology used in multi-source validation is shown as Figure 3.6, i.e. 1 ON-OFF application is installed on each server shown on the left. Packets are generated from these sources and sent to the server shown on the right. The queue in the router is checked using the same method and queue state in recorded. The parameters used is

Load	λ_{on}(pps)	T_{on}(s)	T_{off}(s)	Packet Size(Byte)
0.6	300	0.000033	0.00164	1250
0.7	300	0.000033	0.00140	1250
0.8	300	0.000033	0.00122	1250
0.9	300	0.000033	0.00108	1250
0.95	300	0.000033	0.00102	1250

Table 3-A: Parameters of the ON-OFF source for single source validation over UDP

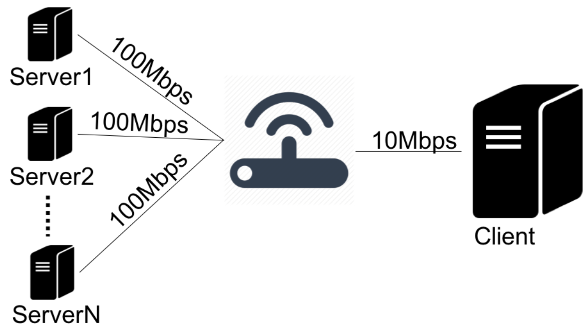

Figure 3.6: Topology for multiple source validation.

shown in Table 3-B and Table 3-C. To validate the VoIP traffic. 10 ON-OFF sources

Load	λ_{on}(pps)	T_{on}(s)	T_{off}(s)	Packet Size(Byte)
0.6	300	0.000033	0.0033	1250
0.7	300	0.000033	0.0029	1250
0.8	300	0.000033	0.0025	1250
0.9	300	0.000033	0.0022	1250
0.95	300	0.000033	0.0021	1250

Table 3-B: Parameters of the ON-OFF source for two source validation over UDP

Load	λ_{on}(pps)	T_{on}(s)	T_{off}(s)	Packet Size(Byte)
0.6	300	0.000033	0.0083	1250
0.7	300	0.000033	0.0071	1250
0.8	300	0.000033	0.0062	1250
0.9	300	0.000033	0.0055	1250
0.95	300	0.000033	0.0052	1250

Table 3-C: Parameters of the ON-OFF source for five source validation over UDP

are installed in the server in Figure 3.5. The queue state is checked and recorded using the same method. The parameters of 10 ON-OFF sources are shown in Table 3-D.

Load	Service Rate (Mbps)	λ_{on}(pps)	T_{on}(s)	T_{off}(s)	Packet Size(Byte)
0.6	1.80	0.1075	0.96	1.69	218
0.7	1.54	0.1075	0.96	1.69	218
0.8	1.35	0.1075	0.96	1.69	218
0.9	1.20	0.1075	0.96	1.69	218
0.95	1.14	0.1075	0.96	1.69	218

Table 3-D: Parameters of the ON-OFF source for validation of VoIP Traffic

3.2.1.2 Validation Results

The validation results of Poisson traffic from a single traffic source are shown in Figure 3.7. The queue states measured from simulation match well with the theoretical

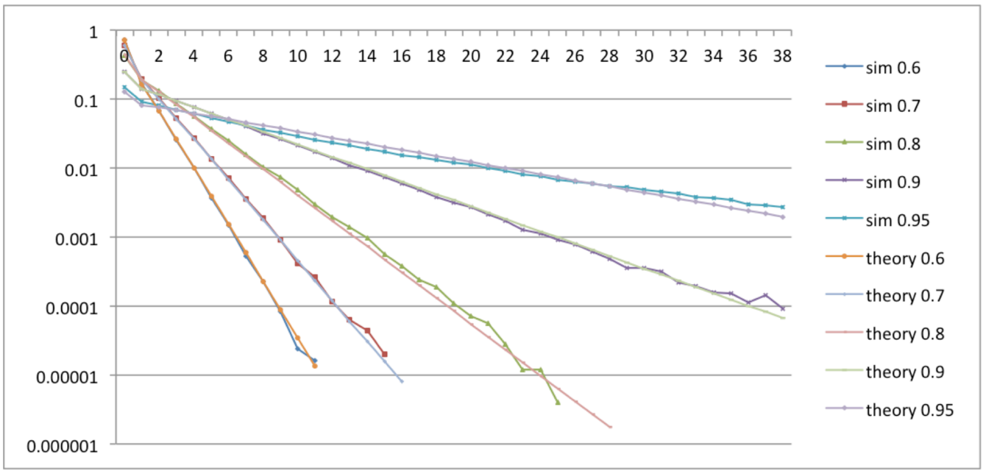

Figure 3.7: Single Source Validation over UDP.

values.

The multiple sources Poisson traffic validation is done by extending the topology in Figure 3.6. 2 sources and 5 sources are created respectively. The results are shown in Figure 3.8 and Figure 3.9. The multiple validation results shows that simulation results match well with the theoretical results.

Figure 3.10, 3.11, 3.12, 3.13, 3.14 compares the simulation results of VoIP traffic with the theoretical values. Figure 3.10 to 3.11 show that the simulation results go above the theoretical value with same decay rates for burst scale. Figure 3.13 and 3.14 show that the simulation results go below the theoretical values with same decay rates for burst

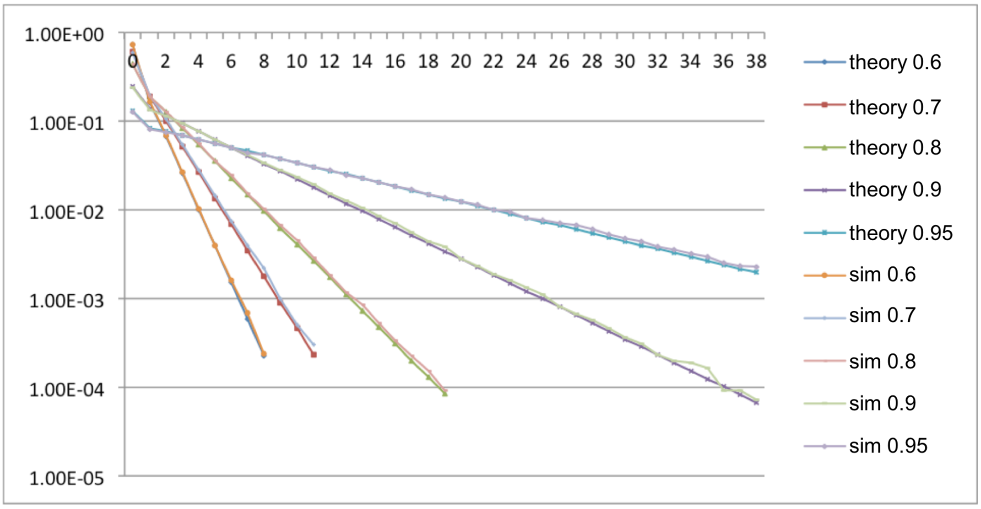

Figure 3.8: Two Sources Validation over UDP.

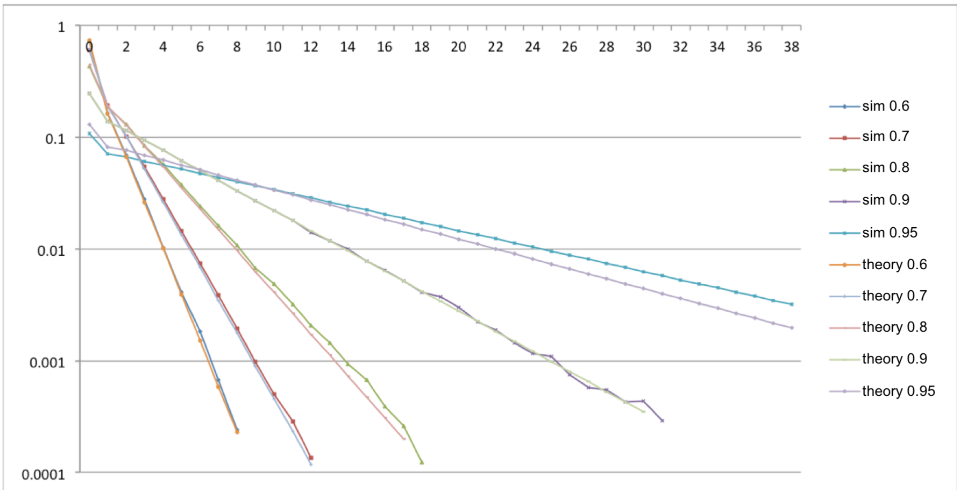

Figure 3.9: Five Sources Validation over UDP.

scale. Figures above are considered to be validated to the acceptable extent. Figure 3.12 shows that the simulation results matches with the theoretical values for burst scale. From all the results of Poisson and VoIP traffic validation, it implies that the simulation tool and traffic model implementation and simulation topology are validated.

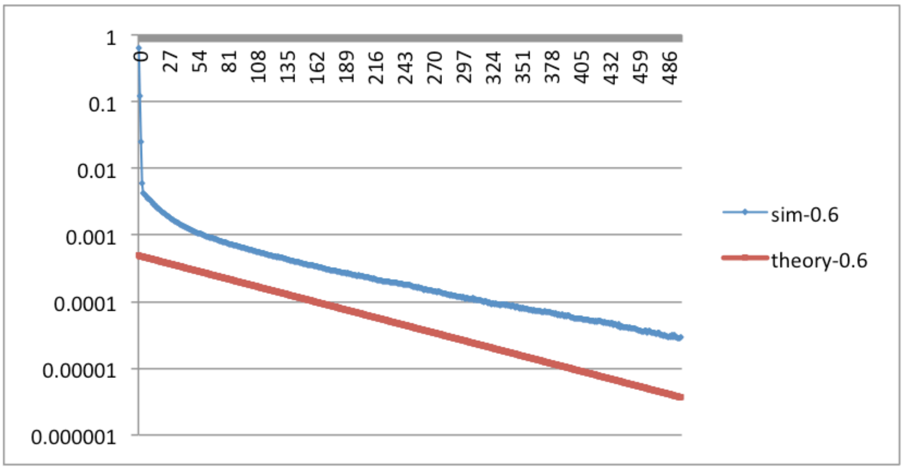

Figure 3.10: VoIP Traffic Validation (Load=0.6).

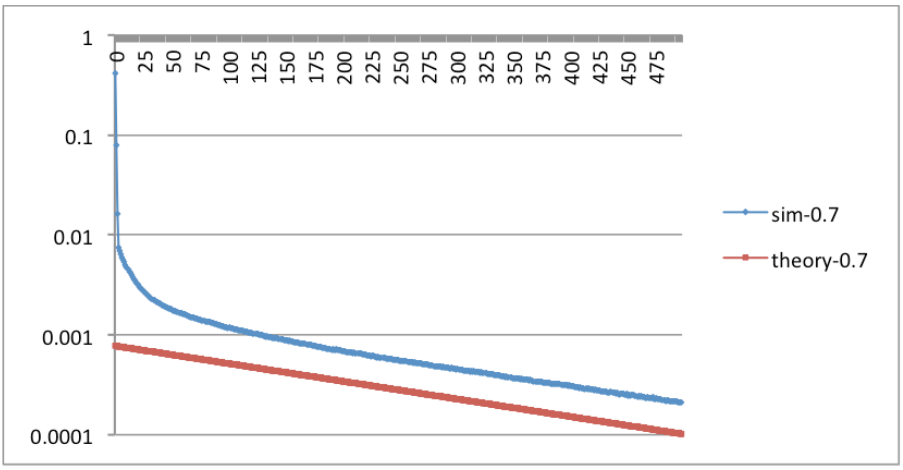

Figure 3.11: VoIP Traffic Validation (Load=0.7).

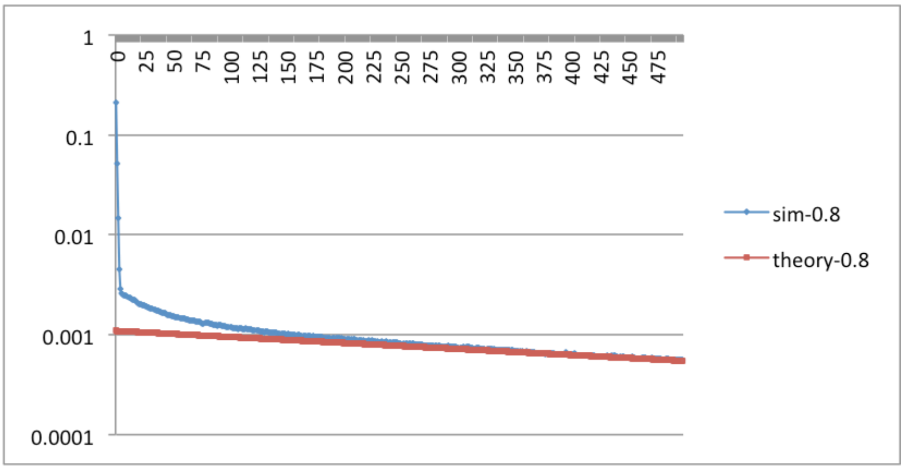

Figure 3.12: VoIP Traffic Validation (Load=0.8).

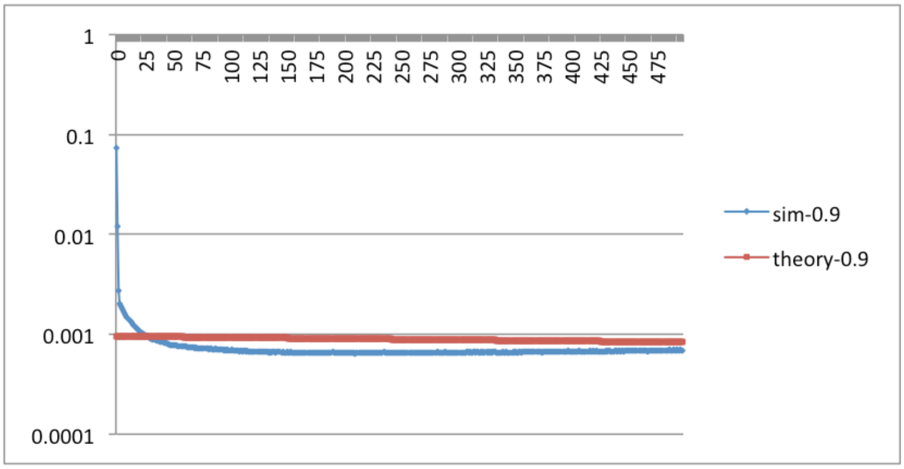

Figure 3.13: VoIP Traffic Validation (Load=0.9).

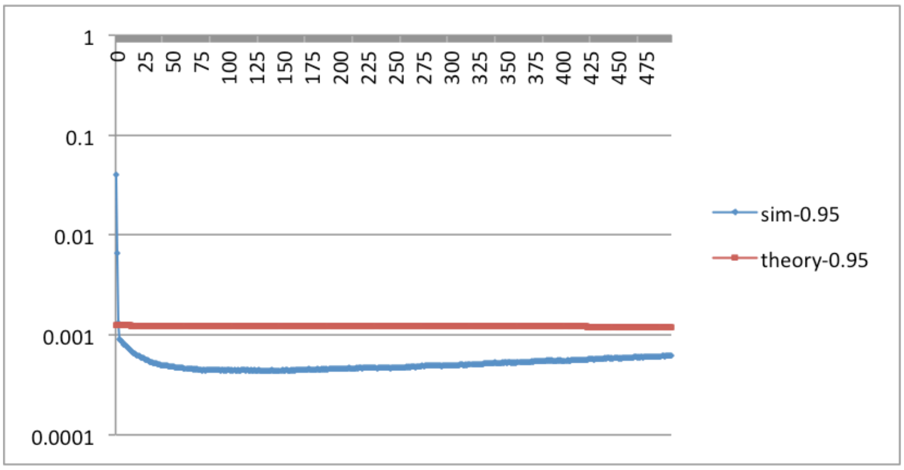

Figure 3.14: VoIP Traffic Validation (Load=0.95).

3.3 Evaluation Design

Existing work only test AQMs with FTP traffic, which is not sensitive to delay. In this chapter, I evaluate performance of two prevalent AQM algorithms, CoDel and RED, in Wi-Fi networks using two different kind of traffic scenarios. Traffic scenario one is dominated by FTP traffic with single VoIP flow and Traffic scenario two is dominated by VoIP flow with single FTP traffic. Then, conduct a thorough performance analysis on CoDel under different network scenarios.

The topology used in the simulation is shown in Figure 3.15. Ten clients are connected to the gateway over a wireless connection. The gateway is connected to the server over a wired link. The wired connection is set to be the bottleneck in the simulation and the speed of it is set depending on the load needed. The speed of the wireless access network is higher and is related to the protocol used in the MAC layer. As 802.11a is used in the simulation, the theoretical speed is up to 54 Mbps. Two traffic scenarios are used in

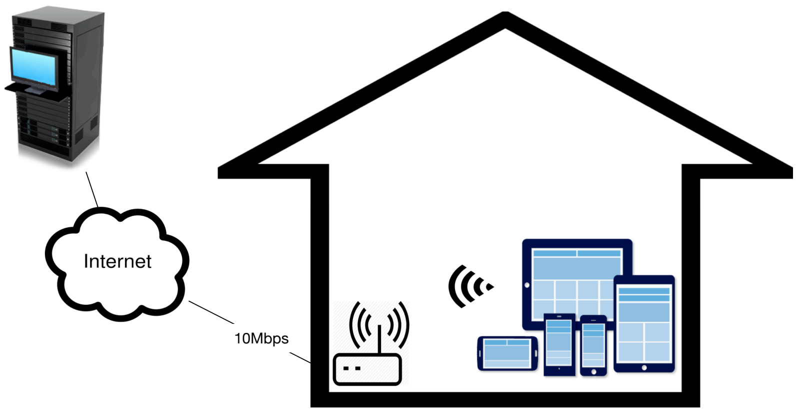

Figure 3.15: Simulation Topology.

the simulation experiments. The details of the parameters in each scenario are shown in Table 3-E. To simulate large file transfers, the FTP application will keep sending data trying to utilize the bandwidth and the sending rate is limited by the sliding window size in TCP. A standard On-OFF traffic model is used to simulate the VoIP traffic. Experiments are carried out with three different queue disciplines, i.e. DropTail, CoDel

Table 3-E: Parameters in Simulation

	Scenario I	Scenario II
No. of FTP Flows	10	1
No. of VoIP Flows	1	10
Mean On Time	0.96s	0.96s
Mean Off Time	1.69s	1.69s
Load of VoIP	4%	48%
Bottleneck link speed	10Mbps	2.6Mbps
VoIP Packet Size	218B	218B
FTP Packet Size	1000B	1000B

and RED. For each traffic scenario, the length of the queue, the waiting time of the packets in the queue, the loss rate and the goodput are monitored. The physical queue limit is set to 1000 packets (default value in Linux) for all the three cases. The default target value of CoDel is commonly set to 5 ms for wired access networks. To fit the wireless scenario in the simulation, the target is set to 50 ms [JBT14] The minimum threshold and maximum threshold of the RED queue are set to 50 packets and 100 packets respectively.

3.4 Simulation Results and Discussion

3.4.1 Traffic Scenario I (10 FTP Flows and 1 VoIP Flow)

Figure 3.16 shows the cumulative distribution of the queue states. With DropTail, packets are accumulated in the buffer and the queue can be as long as 1000 packets which means the buffer is physically full. The mean queue length is 91% shorter with CoDel

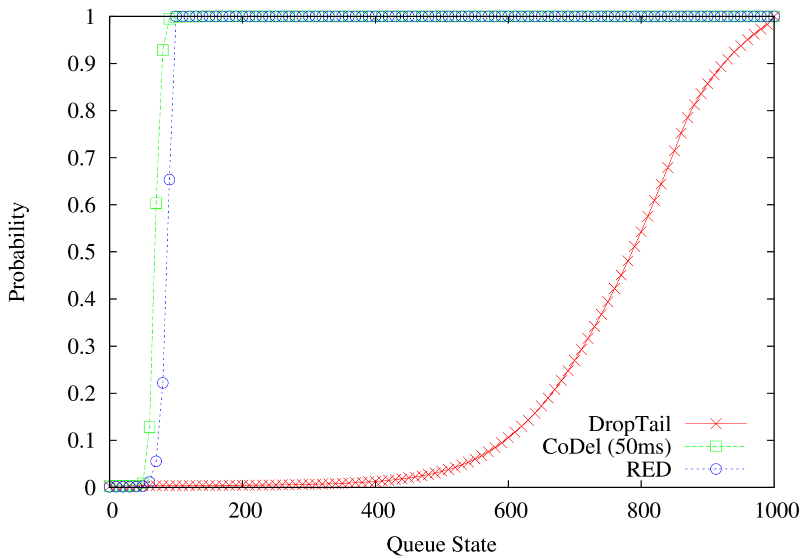

Figure 3.16: CDF of Queue State with Different Queue Management Techniques in Traffic scenario I.

and is 88% shorter with RED. It can be seen that the performance of CoDel is similar to RED from the perspective of queue depth.

Figure 3.17 shows the waiting time experienced by packets belonging to FTP and VoIP flow with different queue management techniques. As shown in Figure 3.17, it can be

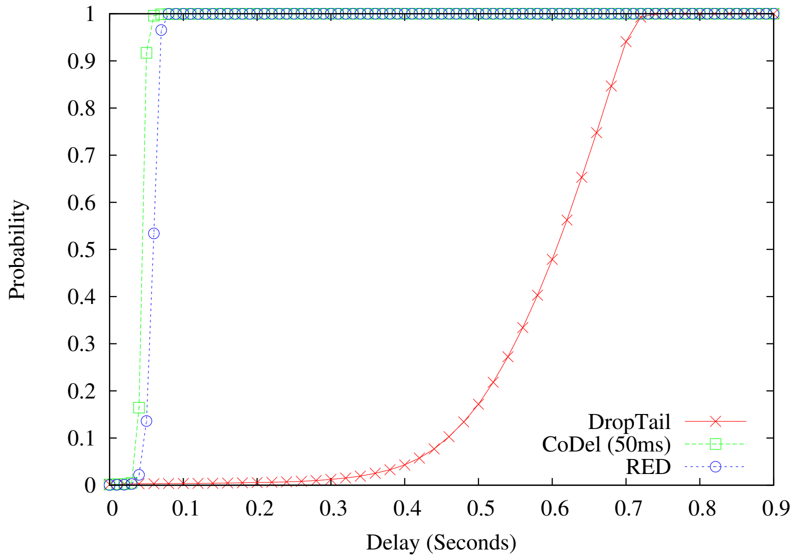

Figure 3.17: CDF of Delay of FTP and VoIP Flow with different Queue Technique.

seen that, for most of the packets, the waiting time with DropTail can be over 500 ms; however, with CoDel, the delay of packets in the queue is reduced to around 150 ms on average. However, RED is not always better than CoDel. The drop probability is shown in Table 3-F. It can be seen that CoDel drops more packets to maintain a shorter queue. DropTail queue has to drop some packets because it is physically full. To further explore the effects of the target value, CoDel with other target values, such as 100 ms and 150 ms, are also tested, as shown in Figure 3.20. The performance of CoDel with differet target will be discussed later. It is true that different performance level of RED can be achieved by tuning the parameter of RED and sometimes it can provide good performance. As discussed in Section II, although a lot of effort has been made to

Table 3-F: Drop Rate in Traffic Scenario I

	DropTail	CoDel (50 ms)	RED
FTP	0.0279%	0.9498%	0.8120%
VoIP	0.0563%	1.1507%	1.110%

enhance the performance of RED, properly tuning RED is challenging. CoDel, on the contrary, can be easily tuned by different requirement of Quality of Service (QoS), as the only metric concerned is delay rather than minimum/maximum threshold, which is much more promising.

3.4.2 Traffic Scenario II (10 VoIP Flows and 1 FTP Flow)

The queue length CDF is shown in Figure 3.18. It can be seen that the behavior of

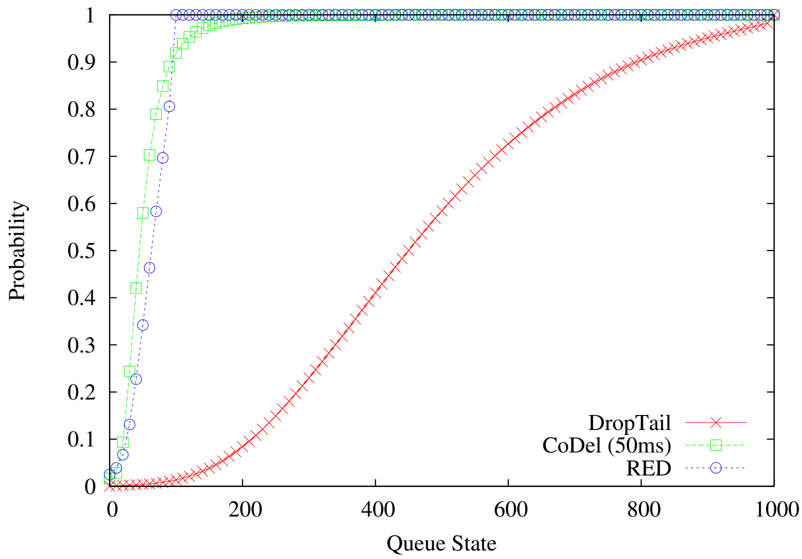

Figure 3.18: CDF of Queue State with Different Queue Management Techniques in Traffic scenario II

queue techniques is quite different (except for RED) compared with that in Figure 3.16. The reason RED provides similar performance from the queue depth aspect is that it counts the queue by packet. As the maximum threshold is set to 100 packets, it will drop all the coming packets when the queue depth reaches the threshold. CoDel is less affected by the queue size as it is based mainly on waiting time. Although the target value remains the same (take 50 ms target for example), CoDel can accept more packets as the proportion of small packet is increased. This is why CoDel (50 ms target) and RED are crossed. Figure 3.19 shows the delay of FTP and VoIP flows. CoDel (50 ms target) provides the shortest waiting time. Considering drop rate, as shown in TABLE

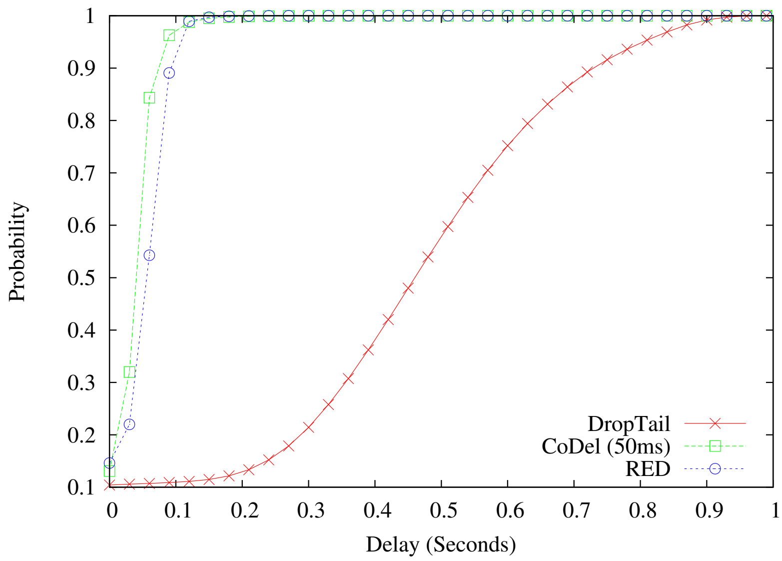

Figure 3.19: CDF of Delay of FTP Flow and VoIP Flow with differnt Queue Techniques.

3-G, it can be seen that CoDel (50 ms target) outperforms RED and DropTail overall as it provides lower delay for VoIP packets compared with DropTail and lower drop rate compared with RED. Among the 3 cases shown in TABLE 3-G, RED has the highest drop rate for both FTP flows and VoIP flows. For VoIP flows, the drop rate with RED for VoIP flows is 8 times, 2.5 times, compared with that of DropTail and CoDel (50 ms) respectively and it gives the lowest goodput.

Table 3-G: Drop Rate in Traffic Scenario II

	DropTail	CoDel (50 ms)	RED
FTP	0.0345%	0.2092%	0.3288%
VoIP	0.3469 %	0.9594%	2.4499%

3.4.3 CoDel with Different Target

In this section, CoDel with different target values, i.e. 50 ms, 100 ms and 150 ms are tested. Figure 3.20 shows the CDF of queue depth of CoDel with different target values. As expected, the higher the target is, the longer the queue is, which also gives a larger queuing delay. In Scenario I, the maximum queue depth of CoDel with different

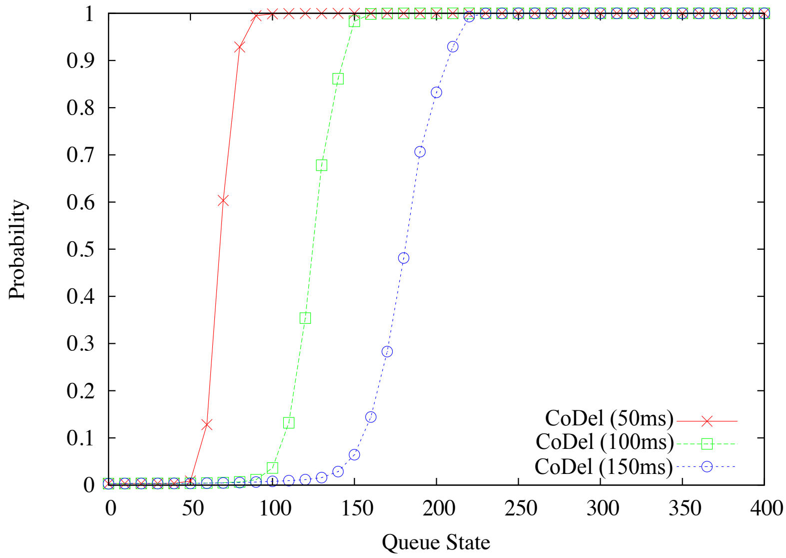

Figure 3.20: CDF of Queue State of CoDel with Different Target in Traffic Scenario I

targets can reach 100, 150 and 200 packets respectively, as shown in Figure 3.20 while in Scenario II, the maximum queue depth of CoDel with different targets can reach 200, 300 and 400 respectively. Different traffic type can lead to different maximum queue length is because the packet size of VoIP is smaller than FTP, as shown in Figure 3.21. And due to the different traffic mix, the proportion of VoIP packets increases. As the key metric of CoDel is the waiting time, it can keep the waiting time of an enqueued packet around the target value regardless of the number of packets in the queue and the size of the packet, as shown in Figure 3.22 and Figure 3.23. With regards to drop rate, the higher the target is, the fewer packets are dropped, as can be seen from TABLE 3-H and TABLE 3-J. Considering the goodput, when traffic is dominated by TCP flows such as in Scenario I, CoDel with a smaller target value will drop packets earlier and hence prevent TCP flows from increasing their sending rate too much, which works well for delay-sensitive applications: this is why VoIP flow has higher goodput, as shown in Table3-I, when the target is set to 50 ms. It's also worth mentioning that although increasing the target value of CoDel can decrease the drop rate, the goodput for VoIP application is decreased three times with target of 100 ms and 150 ms compared with

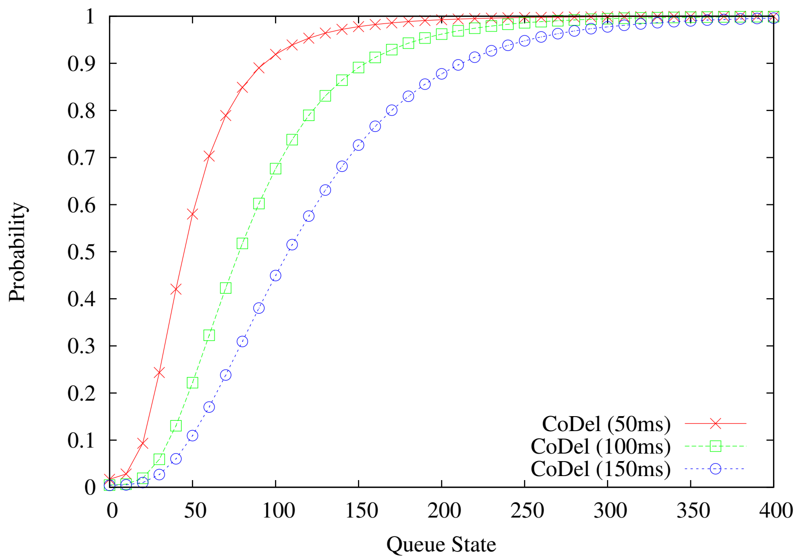

Figure 3.21: CDF of Queue State of CoDel with Different Target in Traffic Scenario II

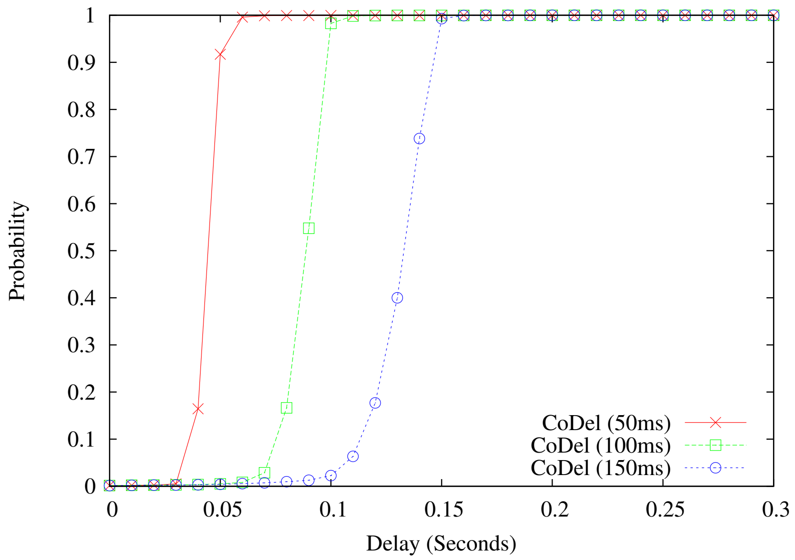

Figure 3.22: CDF of Delay of CoDel with Different Target in Traffic Scenario I

that with target of 50 ms. It is because that FTP takes most of the buffer and when VoIP traffic arrives at the buffer, CoDel already enters the drop state. It can be seen that short bursty traffic has a poor share of the bandwidth. In this chapter, we evaluated the performance of the prevalent AQMs (CoDel and RED) in Wi-Fi access networks. Two traffic scenarios are tested. Scenario I is dominated by TCP traffic, which contain

Table 3-H: Drop Rates in Traffic Scenario I

	CoDel (50 ms)	CoDel (100 ms)	CoDel (150 ms)
FTP	0.9498%	0.3978%	0.2020%
VoIP	1.1507%	0.5072%	0.2827%

Table 3-I: Goodput in Traffic Scenario I

	CoDel (50 ms)	CoDel (100 ms)	CoDel (150 ms)
FTP (Mbps)	9.3939	9.3802	9.3913
VoIP (Mbps)	0.3806	0.1184	0.1186
Total (Mbps)	9.7745	9.4986	9.5099

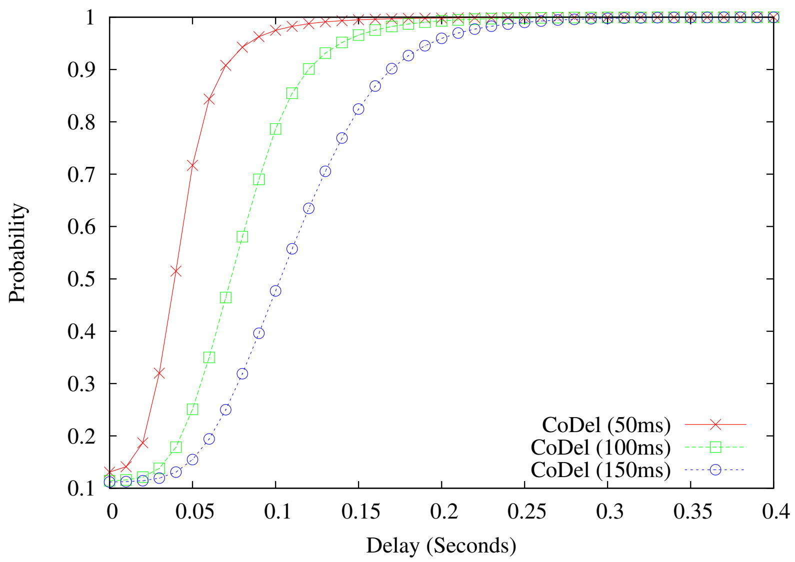

Figure 3.23: CDF of Delay of CoDel with Different Target in Traffic Scenario II

Table 3-J: Drop Rate in Traffic Scenario II

	CoDel (50 ms)	CoDel (100 ms)	CoDel (150 ms)
FTP	0.2092%	0.1124%	0.0745%
VoIP	0.9594%	0.6789%	0.5013%

10 FTP flows and 1 VoIP flow. Scenario II has a higher proportion (nearly 50%) of traffic from delay-sensitive applications, which contains 10 VoIP flows and 1 FTP flow. Simulation results show that no matter which traffic is dominating, all AQMs can effectively control the queue length around the respective threshold or target values and thus

reduce the delays experienced by packets. However, the trade-off is that more packets are lost. It can be seen that for the scenario with a higher proportion of TCP flows, the drop rate with CoDel (50 ms) for VoIP packets is nearly 1% as shown in TABLE 3-G while the drop rate with RED for VoIP packets can be as high as 2.45%. And the drop probability with RED for FTP packets is around 0.33% while it is 0.21% for FTP packets with CoDel (50 ms). As the delay metric is quite straightforward, tuning CoDel is easier than tuning RED.

For the scenario dominated by TCP traffic, it can be seen from TABLE 3-F that the drop rate with CoDel (50 ms) is higher than that of RED, which is due to the greedy characteristic of TCP flows. An AQM with adaptive drop probability can allow the congestion window of TCP to increase further although congestion has already occurred. Compared with DropTail, CoDel provides some packet loss to signal congestion but allows the congestion become more serious especially when there are long-lived TCP flows. On the contrary, RED has a hard limit which stops the congestion window increasing further and hence drops less packets than CoDel. When bursty traffic compete for bandwidth with long-lived TCP flows, performance will be degraded if AQMs do not tune properly, such as lowering the goodput for VoIP flows.

3.5 Summary

The target environment for this chapter is Wi-Fi access networks where the performance of DropTail, RED and CoDel (with different target values) are evaluated. A deep insight on how CoDel performs when facing the Bufferbloat phenomenon is revealed based on a thorough performance evaluation under different traffic scenarios. Simulation results shows that AQMs help prevent long standing queues thereby mitigating the “Bufferbloat” phenomenon although at the expense of higher packet drop rates. CoDel is more friendly to VoIP traffic when the traffic is dominated by VoIP traffic. On the contrary, RED drop 2.45% of VoIP packets which is very high for VoIP services. Moreover, although CoDel claims to be parameterless, there are actually 2 parameters, target and interval, that can be adjusted. A larger target value decreases the drop probability for both FTP and VoIP traffic in both traffic scenarios. However, the goodput decrease with the increase of target value when the traffic is dominated by FTP applications. It shows that dropping packets not only help to reduce queuing delay, but also helps to improve the goodput. Based on the evaluation results of this chapter, CoDel is chosen as the benchmark in the following research work. Mobile devices are more and more powerful and integrated with multiple functions. With the expansion of mobile networks, people can have access to Internet almost everywhere. Little work has so far been done in addressing the Bufferbloat issue in cellular networks via AQMs. In next chapter, an channel quality aware AQM is proposed.

Chapter 4

Channel Quality Aware Active Queue Management in Cellular Networks

This chapter proposes a novel AQM algorithm tailored to cellular networks, mainly by utilizing the Channel Quality Indicator (CQI) periodically reported by user equipment, in order to mitigate Buffer bloat and maintain acceptable levels of performance. Simulation results show that the proposed algorithm reduces the average queuing delay of packets for each user by 40% with TCP traffic compared with the CoDel algorithm. Meanwhile, the goodput is minimally affected.

4.1 Introduction

The emergence of powerful smart devices and their integration in people's daily lives place huge strains on networks. As forecast by Cisco [cic17], global mobile traffic will increase eight folds by 2020 and the link speed will increase by three fold. Since 4G, cellular networks have been all IP-based while widely used by applications such as video streaming, gaming and online chatting. A cellular communication system is shown in Figure 4.1. Although link speed of the last hop has increased with the advance of tech-

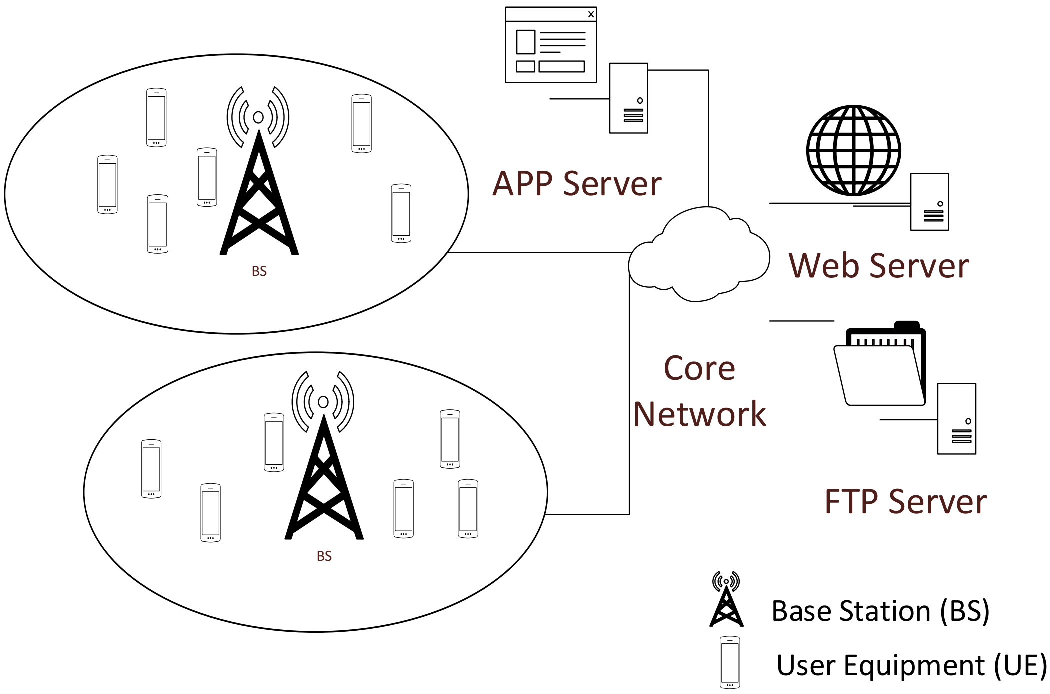

Figure 4.1: Cellular Network Structure

nology, the access networks are still considered as the bottleneck as there large number of UEs sharing the bandwidth, while other links within the core network and the link between servers and the core network is fast. The idea of the proposed algorithm is straightforward. In cellular networks, unique features of the underlying cellular networks such as the channel conditions can have a considerate impact on the behaviour of the queue. The time-varying channel is affected by the mobility of users, the density of users in one cell etc. Good channel conditions mean advanced MCS are chosen, and so users can achieve faster Internet connections. And with poor channel conditions, lower rate MCSs are chosen, which results in slow Internet connections in order to achieve reliable transmission. A faster connection will transmit the packets in the queue faster while a slower connection means extra queuing delays occur. The channel condition can be easily obtained as UEs will measure it and report to the eNB every Transmission Time Interval (TTI). Each UE has a unique buffer in the eNB and the respective UE can only transmit data when resource blocks are allocated to it. With the increasing number of devices and much more powerful devices, the competition of wireless resource is fierce. The number of mobile devices will keep increasing at a CAGR of 8% between 2016-2021 [cic17]. Increasing the capacity of served UEs or allocating more resources to UEs will not solve the issue of congestion in the buffer. In the proposed algorithm, UEs with poor channel quality may drop more packets compared with the ones with good channel quality. We also take delay into consideration as UEs with bad channel quality is more likely to suffer from large queuing delays.

In this Chapter, a CQI-aware AQM algorithm is proposed which is light-weight but can effectively control the delay. Section 4.2 describes how CoDel is tailored to cellular networks. Section 4.3 gives the details of the proposed algorithm. Section 4.5 gives the simulation results and discussion. Section 4.6 gives the conclusion.

4.2 Implementation of CoDel in RLC Layer

CoDel makes the decision about whether to drop a dequeuing packet periodically. If the waiting time of a dequeuing packet goes over the minimum allowed value (target), it starts to count certain time period which is called interval. During this interval, if the waiting time of all the dequeuing packets goes over the target, it will drop one packet when the interval ends.

The enqueuing algorithm is the same as standard algorithm. When a packet is received by RLC layer, RLC layer will add a timestamp to the packet. The enqueue algorithm is shown in Algorithm 1 However, the dequeue algorithm needs to be chagend a bit from the original version. Drop from head makes CoDel not suitable for cellular networks

Algorithm 1 CoDel in the Base Station - Enqueue

On the arrival of each packet:
if $qlen_{current} < qlen_{limit}$ **then**
    Add timestamp, T_{in} to the packet
    Enqueue the packet
else
    Drop the packet
end if

due to the limitations from MAC layer. The HOL packets might be segmented and the remaining part will be returned to the queue, as shown in Figure 4.2. If the returned

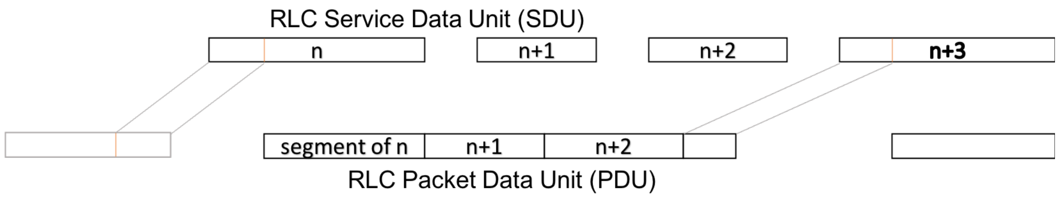

Figure 4.2: RLC Packet Segmentation

part is dropped, it will influence to entire flow as no retransmission mechanisms can help to recover the missing part. Hence, a change is needed for CoDel to work properly in cellular networks. Instead of examining the HOL packet in the buffer, CoDel will examine the second packet, every time the HOL packet is about to dequeue. In this way, CoDel is tailored for cellular networks. Details are given in Algorithm 2.

4.3 CQI-Aware Queue Management

One of the main differences between cellular network and Wi-Fi is that the channel condition in cellular networks is much more complex. For Wi-Fi networks, it often aims to server a closed space, such as in a house or a room. The number of equipment accessed is small and the strength of interference is low compared to the signal from wireless routers. However, for the cellular scenario, the eNB servers an open space. There will be blocks of buildings standing in the path of the of communication, crowded people moving in or out, strong interference from other equipment and etc.

Algorithm 2 CoDel in the Base Station - Dequeue

```
On the depature of each packet:
Calculate the sojourn time of the pakcet as:
    d_queuing = T_current − T_in
if dropping_state == 1 then
    if d_queuing < target OR q_len < MTU then
        Forward the packet
        Leave dropping state
        dropping_state = 0
    else
        while dequeue_time ≥ next_drop_time do
            if pktStatus == FULL then
                Drop the pakcet
            else
                if q_len > 2 then
                    Drop the second packet from the head
                    count+ = 1
                    Update next_d rop_t ime as next_drop_time+ = interval/√count
                else
                    Leave dropping state
                    dropping_state = 0
                end if
            end if
        end while
    end if
else
    if dropping_state == 0 AND
        d_q ueuing ≥ target AND
        firstAboveTarget == false then
        Drop the packet
        Enter dropping state
        dropping_state = 1
    else
        Forward the packet
    end if
end if
```

CQI [ETS] is the control message used between the UE and eNB. CQI is detailed illustrated in Table 4-A. CQI contains information to indicate the (MCS) value. UE assesses the CQI and send it to the eNB by Physical Uplink Control Channel (PUCCH) if there are no data transmission or by Physical Uplink Share Channel (PUSCH) if there are data transmission. Higher CQI value means better channel quality and hence can use higher modulation scheme. Packets are transmitted by resource blocks in cellular networks. A

CQI Index	**Modulation**	**Code Rate** * 1024	**Efficiency**
1	QPSK	78	0.1523
2	QPSK	120	0.2344
3	QPSK	193	0.3770
4	QPSK	308	0.6061
5	QPSK	449	0.8770
6	QPSK	602	1.1758
7	16QAM	378	1.4766
8	16QAM	490	1.9141
9	16QAM	616	2.4063
10	64QAM	466	2.7305
11	64QAM	567	3.3223
12	64QAM	666	3.9023
13	64QAM	772	4.5234
14	64QAM	873	5.1152
15	64QAM	948	5.5547

Table 4-A: 4-bit CQI Table [ETS]

resource block is the smallest unit of resource that can be allocated to a user. User with a high CQI can transmit more data in each resource block.

The proposed algorithm keeps tracking the HOL delay of each packet. When a packet arrives at the queue, a timestamp is added to the packet and when it leaves the queue, the waiting time experienced by the packet is calculated. The random dropping decision for next incoming packet is made according to channel quality of the user i.e. CQI, and the delay experienced by the packet about to leave the queue. The dropping probability is controlled by Eq (4.1), when $d_{min} \leq {d^i}_{queuing} \leq d_{max}$. $d_{queuing}$ is the HOL delay of each packet, and d_{max} is the maximum queuing delay.

$$P^i{}_{drop} = -\beta * (d_{max} - {d^i}_{queuing}) * k_i^3 + 1 \tag{4.1}$$

d_{max} can be preset and when ${d^i}_{queuing}$ reaches d_{max}, the drop probability equals to 1. When ${d^i}_{queuing}$ is below d_{min}, the drop probability will be 0, as shown in Eq (4.2).

$$P^i{}_{drop} = \begin{cases} 0, & \text{if } {d^i}_{queuing} \leq d_{min} \\ 1, & \text{if } {d^i}_{queuing} \geq d_{max} \end{cases} \tag{4.2}$$

k_i is the CQI index reported by the i_{th} UE to the BS. ${d^i}_{queuing}$ can range from tens of milliseconds to hundreds of milliseconds, hence the cube of CQI is adopted in the formula. β is the index normalization coefficient, as shown in Eq (4.3) where $k_{i_{max}}$ is the maximum CQI value can be achieved.

$$\beta = \frac{1}{(d_{max} - d_{min}) * k_{i_{max}}^3} \tag{4.3}$$

The range of the drop probability is shown in Figure 4.3. With the increase of the

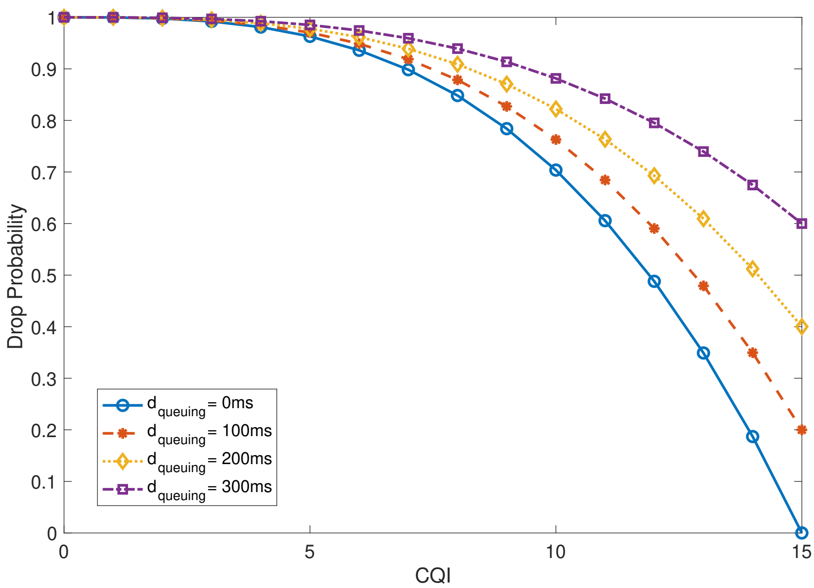

Figure 4.3: Dropping probability function of the proposed method.

queuing delay, the drop probability increases, which can help reduce the waiting time of each packet. Different UEs may experience different channel qualities and the CQI value determines the transmission rate. For the UEs with low CQI value, the queue will build up quickly as the depletion rate is low. Hence, UEs with low CQI value has a higher drop probability.

UEs with different CQI can suffer different delay, so we have to input these parameters into the algorithm that calculates the drop probability in order to give a quick response to congestion. The algorithm examines the waiting time for each packet. UEs with poor

CQI drops more packets. And even though the channel quality of the UEs are good, UEs suffering larger delay can drop more packets. In both cases, deploying the proposed algorithm will prevent the queue from further growth. Details of the proposed algorithm are shown in Algorithm 3.

Algorithm 3 CQI Aware Active Queue Management

Calculate the HOL queuing delay of the i_{th} UE

for each pkt arrives at the RLC of the i_{th} UE **do**
    Add a timestamp T_{in}
end for
for each pkt leaves RLC **do**
    Record the current time $T_{current}$
    $d^i{}_{queuing} = T_{current} - T_{in}$
end for

Packet dropping decision of the i_{th} UE

if $d_{min} < d^i{}_{queuing} < d_{max}$ **then**
    $P^i{}_{drop} = -\beta * (d_{max} - d^i{}_{queuing}) * k_i^3 + 1$
else if $d^i{}_{queuing} \leq d_{min}$ **then**
    $P^i{}_{drop} = 0$
else
    $P^i{}_{drop} = 1$
end if

4.4 Simulation Setup

The proposed algorithm is implemented in Network Simulator (NS) 3. The topology used in the simulation is shown as Figure 4.4. The UEs are randomly distributed within 2500 meters to 5000 meters. The number of UEs varies from 2 to 10. The Buffer at the RLC layer is set to 100 packets and TCP Cubic is used. The propagation delay is set to 50 ms and link rate is 10 Mbps for the connection between the core network and the server. Each Base Station (BS) has 15 resource blocks. According to the quality settings, Guaranteed Bit Rate (GBR) video traffic is generated from the server and sent

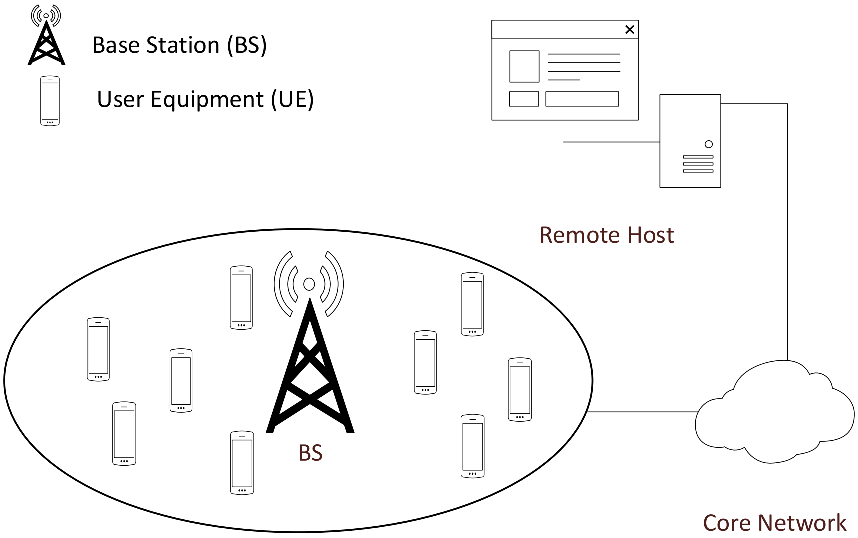

Figure 4.4: Simulation Topology.

to each UE.

The channel quality of a user can change rapidly. For example, a moving obstacle, such as a big truck stops in the way between the UE and the BS for several seconds. To make the simulation more realistic, a random movement model is applied to each UE. When the simulation starts, each UE will randomly choose a direction and a speed uniformly distributed from 50m/s to 100m/s. When they arrive at the edge of the cell, they will stop and choose a new direction and speed. In this way, the CQI of the UE changes with the movement of UE.

The sending rate of TCP traffic changes according to the congestion level of the networks. It will increase when an ACK is received and decrease when the ACK is lost or delayed (depending on different algorithms). Parameter used are listed in Table 4-B.

4.5 Results and Discussion

The performance of the proposed algorithm is evaluated in terms of the queuing delay, the average goodput and the loss probability given by the algorithm. The drop probability

Parameters	Value
Random Distribute Model	UniformRandomVariable
Server BulkSendApplication	Mean $T_{on} = 0.96, T_{off} = 1.69$
Client PacketSink	TCPSocketFactory
Path Loss Model	FriisPropagationLossModel
Scheduling Algorithm	PfFfMacScheduler
Number of UE	2, 4, 6, 8, 10
Distance to Base Station	2500 to 5000 meters
Server Bandwidth	10 Mbps

Table 4-B: Parameters in Simulation

is determined by both the queuing delay and the channel conditions i.e. CQI. When the queuing delay exceeds lower threshold, i.e. d_{min}, the drop probability is calculated by Eq 4.1. The next enqueue a packet will be randomly dropped accordingly. Average end-to-end delay is shown in Figure 4.5. It can be seen from the figure that the average

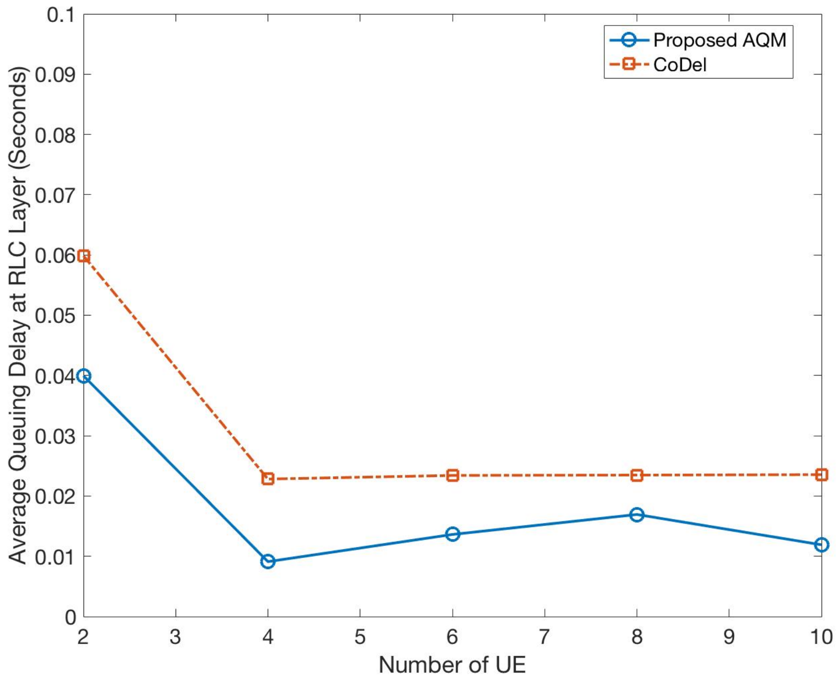

Figure 4.5: Average Queuing Delay at RLC Layer with Increasing Number of UEs.

end-to-end delay decreases when the number of users increases from 2 to 4. It is because the AQMs. When there are more packets waiting in the queue, AQMs drops more packets as shown in Figure 4.6. When the number of users increases from 4 to 8, the queuing delay increases because the resource blocks at MAC layer are shared by all the users. When there are more users, each user needs to wait to be scheduled and at the same time, the packets are also waiting in the queue. When the number of users increases to 10, the queuing delay goes down as more packets are dropped by the proposed AQM due to the increase of waiting time. Deploying AQM at the RLC layer helps reduce the average end-to-end delay, especially when there are more users in the system. Compared with CoDel, the proposed algorithm has better performance from the aspect of average end-to-end delay. Lower average end-to-end delay is achieved by dropping more packets reasonably. The drop probability is shown in Figure 4.6. As traditional cellular networks do not

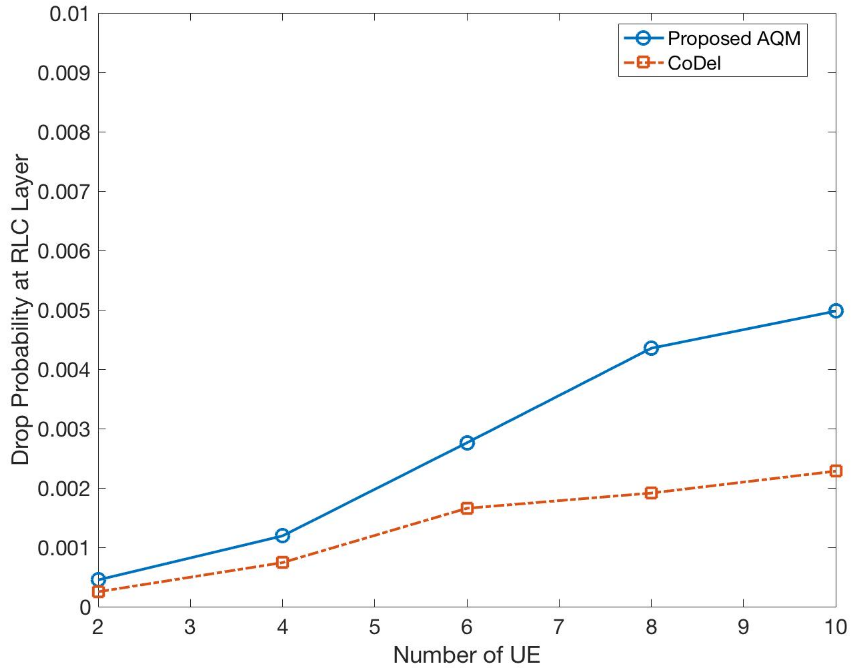

Figure 4.6: Drop Probability

actively drop packets during the transmission, only two curves are shown. Compared with CoDel, the drop probability of proposed algorithm is a bit higher.

Deploying AQM helps to increase the average goodput. As shown in Figure 4.7, both the proposed algorithm and CoDel have higher goodput compared with the scenario without

AQM deployed. Actively dropping a packet gives a signal to the sender that congestion

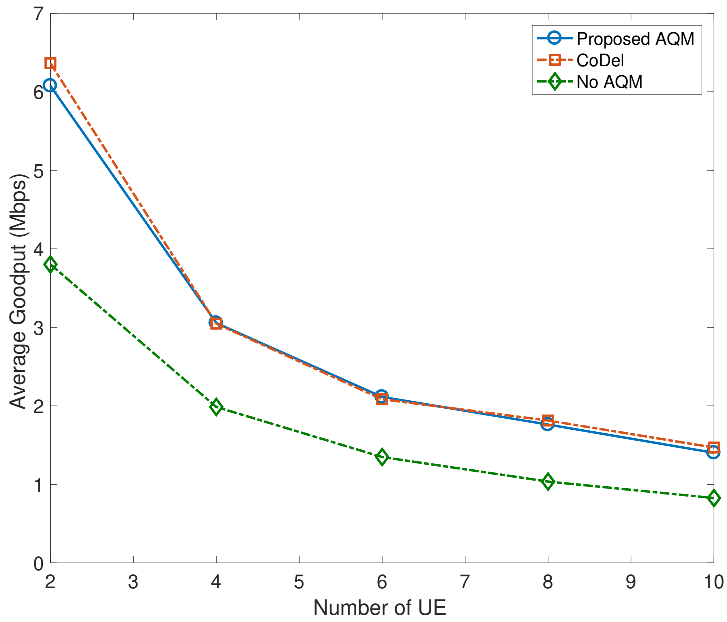

Figure 4.7: Goodput with Increasing Number of UEs.

happens. The sender will stop increasing the sending rate. In this way, AQM algorithms alleviate the congestion of the whole system.

Jain's fairness index is used to rate the fairness in a network when there are multiple users in the system [JCH84]. The Jain's fairness index is shown in Figure 4.8. The UEs are randomly distributed, hence when there only 2 users in the system and they are with different CQI, the fairness index of the proposed algorithm will be lower compared with CoDel. However, better fairness is achieved with the increasing number of UEs, which shows that the proposed algorithm is suitable for multi user scenarios.

4.6 Conclusions

In this Chapter, a CQI aware AQM algorithm is proposed for cellular networks with the goal of mitigating Bufferbloat and improving performance. This algorithm is implemented in the base station where all the connected UEs have a dedicated buffer. The

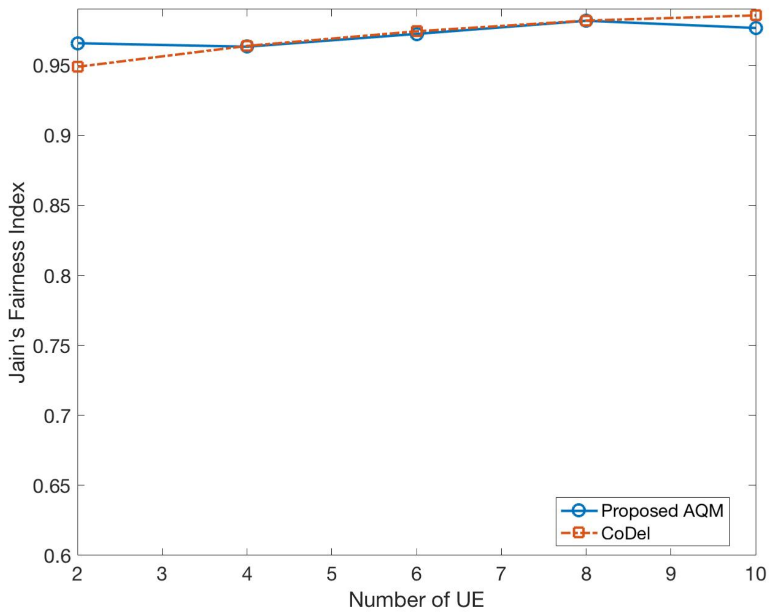

Figure 4.8: Jain's Fairness Index

proposed algorithm considers the channel quality of each user and actively drops packets in order to minimize overall delay and maximize goodput. The drop probability is determined by both the HOL queuing delay and the channel quality. Simulation results show that the proposed algorithm is able to control the delay regardless of the number of UEs in the system and does not harm the other metrics, such as fairness. Meanwhile, it improves the average goodput. It is able to achieve this performance due to the consideration of a specific wireless feature, i.e. CQI, unlike existing AQM algorithms which only consider network layer parameters. Nowadays, the radio environment is complex because of different types of interferences. CQI has a considerable effect on the higher layers hence need to be considered for an AQM to be successful in the cellular environment. The proposed algorithm can also be used in future generations of cellular networks as the CQI still plays an important role in cellular networks in general. In addition to CQI, other parameters such as Buffer State Report (BSR) and Discontinuous Reception (DRX) can also influence network delays and these will be considered in the future work in order to further fine-tune the proposed algorithm.

Chapter 5

User Experience Aware Active Queue Management in Cellular Networks

In Chapter 4, an CQI aware AQM algorithm tailored to cellular network is proposed to reduce the delay of packets. CQI is involved in the proposed algorithm to assist in the dropping decision. The performance is evaluated according to QoS metrics, such as delay, drop and goodput. However, to what extent the improvement means to a user? In this chapter, a novel AQM algorithm tailored to VoIP application in cellular networks is proposed, mainly by utilizing QoE metric in order to mitigate Bufferbloat. Simulation results show that the proposed algorithm provides a good balance between drops and delay hence successfully maintains expected levels of service.

5.1 Introduction

Wireless networks allow users to access the Internet at any time from any places around the world. In the past few decades, cellular networks have evolved rapidly. Nowadays, the cellular networks are IP based and billions of population use mobile phones everyday. The deployment of cellular networks and the expansion of Wi-Fi networks offer the facility of making VoIP calls. VoIP is a technology that allows people to make phone calls over packet switched networks (IP) instead of circuit switched networks, i.e., Public Switched Telephone Network (PSTN). Improving the performance of VoIP applications has drawn an increasing attention by research community. QoE and QoS are important metrics to quantify the performance of VoIP application. [JS17] evaluate the QoS over LTE networks. This work compares the Mean Opinion of Source (MOS), end-to-end delay, packet loss rate and jitter in three different scenario and points out the mobility of the UE is a key factor affecting the performance of VoIP. When the speed of UE changes from 0 to 100 km/h, the QoE value drops from 4.3 to 3.3. This work only considers several UE in the simulations which is not the case in real world. [HHS17] and [AB14] evaluate the performance of VoIP calls over MANET networks with a testbed and simulations. They conclude that different voice codecs affects the performance of VoIP and voice codec G.711 gives the best performance. [DYYL14] evaluates the performance of different voice codecs over Ad Hoc Networks in underground mines. Because of the change of networks and environment, this work gives the conclusion that voice codec G.723 gives the best performance. In Vehicular Ad Hoc Networks (VANETs), [EBBEBB13] shows that voice codec G.723 gives the best performance. Few research work focus on the cellular networks and the effects of AQM are not considered. Thus the user experience aware AQM is proposed aiming to improve the QoE of VoIP in cellular networks. In this chapter, G.711 is used as it gives the best performance in mobile networks as pointed in [HHS17] and [AB14].

5.2 QoE Estimation

5.2.1 Kingman Formula

The waiting line or queue management can be seen everywhere in our daily life. Bus stations have customers in line to get on the buses and customers queue up for seat in the restaurant. Wherever there is contest of resources, there are queues. In telecommunication systems, packets from different sources compete for the chance to be transmitted. The packets wait in the buffer and they can be described as a queue of customer waiting to be served. A general queuing system is shown in Fig. 5.1. Understanding the

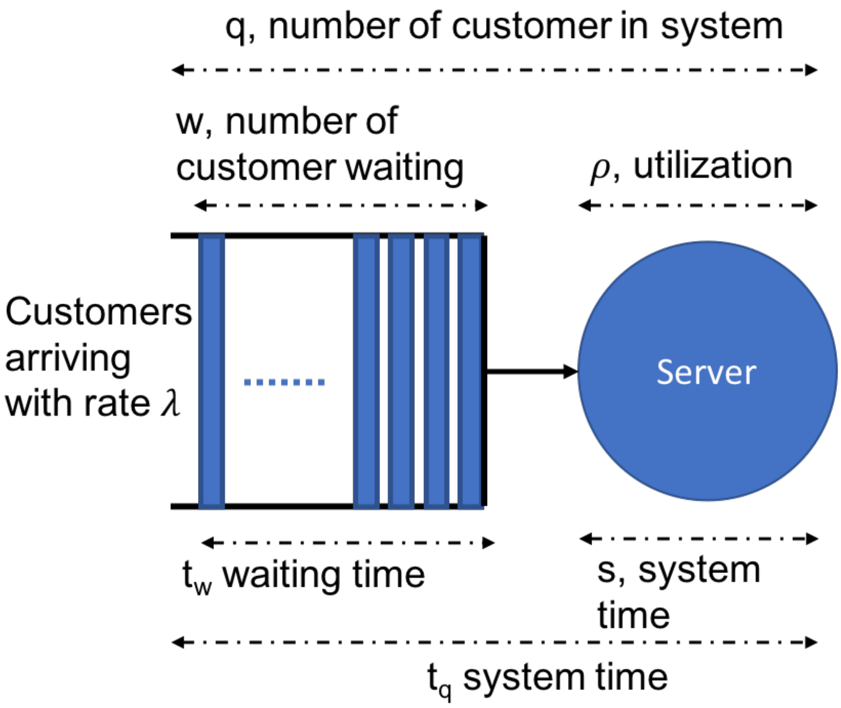

Figure 5.1: Queuing System [PS01]

behaviour of queuing is essential to improve the performance of networks. There are different types of queues and Kendall's notation is used to describe them. A queuing system can be described as $A/B/X/Y/Z$ and the meaning of parameters are shown in Table 5-A. There are four types of distribution can be take and they are listed in

A	Distribution of inter-arrival times of customers
B	Distribution of service time
X	Number of servers
Y	Capacity of the queue
Z	Queue Discipline (e.g., FIFO)

Table 5-A: Parameters in Kendall's Notation

Table 5-B. Developed by John Kingman, the Kingman formula (also known as Kingman

M	Markovian (Memoryless)
D	Deterministic Distribution
E_k	Erlang Distribution (k: shape parameter)
G	General Distribution

Table 5-B: Distribution Types

approximation) describes the waiting time of a packet in the G/G/1 queue shown in Figure 5.2. Kingman formula monitors the utilization and variance of the queue and the equation is as shown in Eq 5.1.

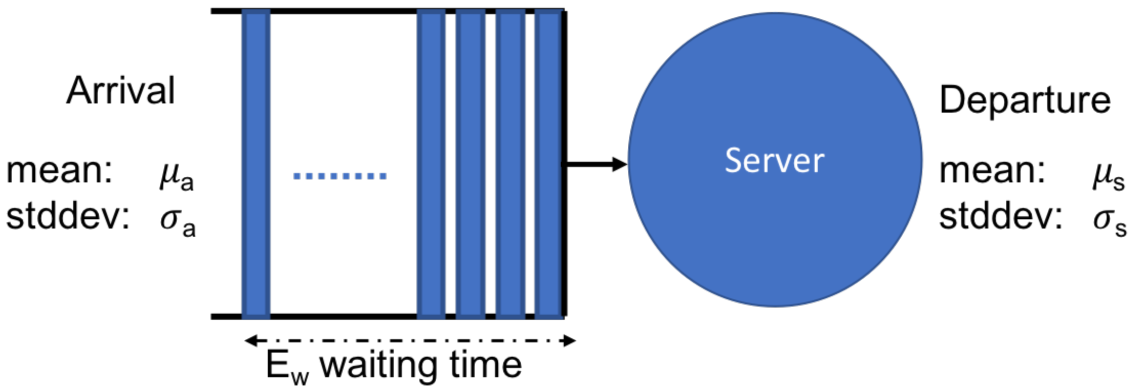

Figure 5.2: G/G/1 Queue

$$E(W_q) \approx \frac{\rho}{1-\rho} \frac{C_a^2 + C_s^2}{2} \tau \tag{5.1}$$

The parameters are explained in Table 5-C and ρ is calculated by Eq 5.2.

$$\rho = \frac{\mu_a}{\mu_s} \tag{5.2}$$

E(W_q)	The estimated waiting time
μ_a	The inter-arrival time of packets
σ_a	The standard deviation of the μ_a
μ_s	The service time of each packet
σ_s	The standard deviation of the μ_s
ρ	Utilization
C_a	The coefficient of variation for the arrival
C_s	The coefficient of variation for the service

Table 5-C: Parameter in G/G/1 Queue and Kingman Formula

5.2.2 IP Multimedia Subsystem

The IP Multimedia Subsystem (IMS) is used to deliver multimedia content over IP networks. IMS is established based on the Session Initiation Protocol (SIP) [RSC+10] which is a communication protocol for signaling and control Internet telephony. IMS is originally designed by 3GPP aiming to provide the access of multimedia and voice application from wireless and wired terminals. IMS applies to UMTS, WiMAX, WLAN, LTE and 5G networks as well as fixed networks. The IMS structure is shown in Figure 5.3. The application servers are applications themselves such as VoIP. The Call Session Control Function (CSCF) acts as SIP proxy. According to the functions, there are three types of CSCF, the Proxy CSCF (P-CSCF), the Interrogating CSCF (I-CSCF) and Serving CSCF (S-CSCF). The P-CSCF will decided the I-CSCF according to the domain name provided by the IMS terminal, i.e., the UE and forward the request of SIP registration from UE to the I-CSCF. The I-CSCF will query the Home Subscriber Server (HSS) to get the address of the user and forward the message from P-CSCF to S-CSCF. S-CSCF accepts the request forward by P-CSCF and cooperate with HSS to authorize UEs and provide the multimedia service. According to information provided by P-CSCF, the control plane will create new IP-bearers for that application. An Evolved Packet System (EPS) Bearer is a connection-oriented network which looks like a virtual tunnel. The structure of an eps bearer is shown in Figure 5.4. The end-to-end bearer is composed

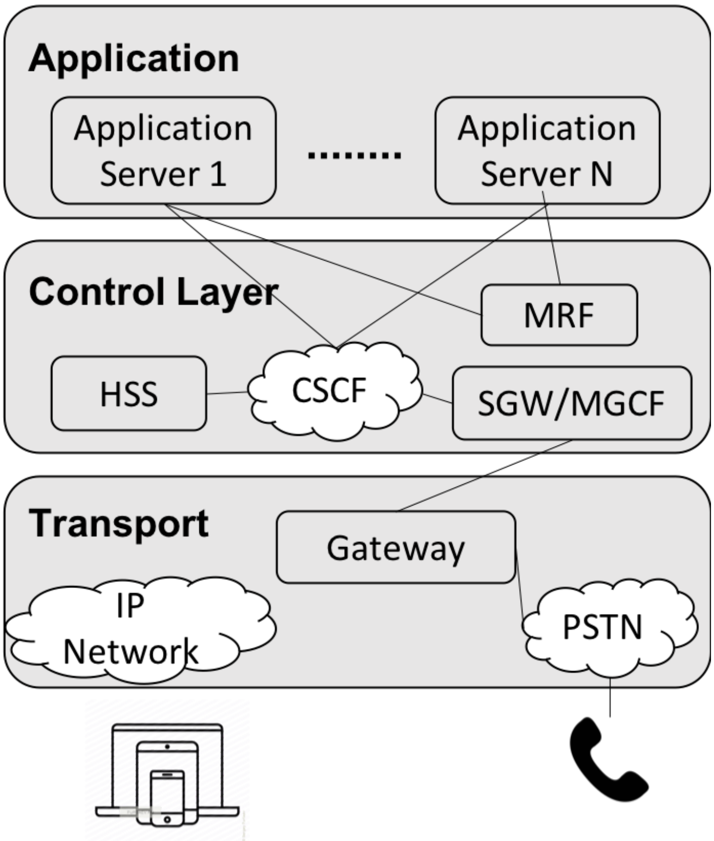

Figure 5.3: IP Multimedia Subsystem

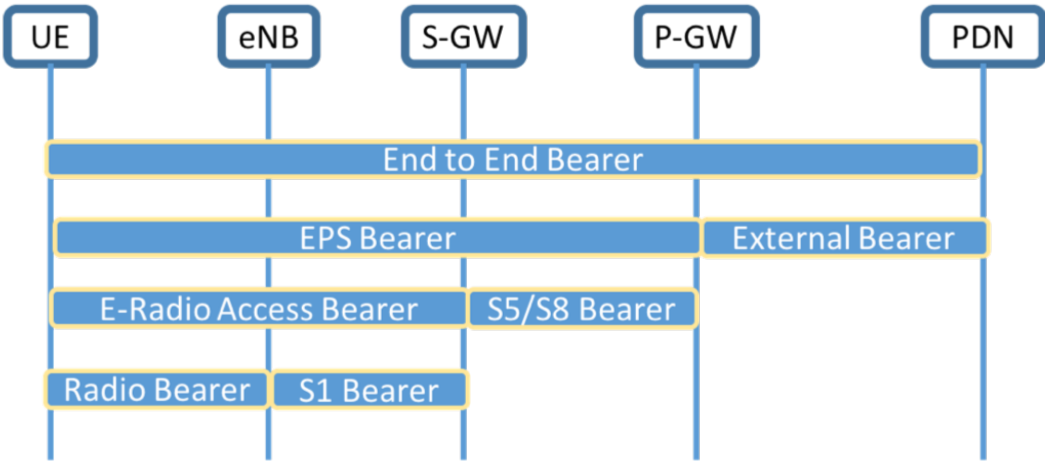

Figure 5.4: Structure of EPS Bearer

of several different bearers. The external bearer is responsible for the data transmission between the Packet Gateway (PGW) to the Packet Data Network (PDN). The EPS Bearer is responsible for the data transmission from PGW to the UE. The EPS Bearer has 2 parts, E-Radio Access Bearer (E-RAB) and S5/S8 Bearer. The S5/S8 Bearer forwards packets between Service Gateway (SGW) and the PGW. The E-RAB forwards packets between the SGW and UEs. The E-RAB is composed of the Radio Bearer and S1 Bearer. The Radio Bearer is responsible for the data transmission of the access networks (between the UE and eNB) and S1 Bearer transmits date between eNBs and SGWs.

5.2.3 QoE for VoIP

The QoE metric is reflected by the MOS score which is recommend by ITU. Normally, the MOS score is the average opinion of quality given by people who are evaluating the content. The QoE level is given by excellent, good, fair, poor and bad. However, obtaining the QoE level by asking people is time-consuming and cannot help to improve the quality of VoIP service in real-time. Hence, the E-model [CR01] is proposed to assess the QoE of VoIP directly. In the E-model, ration factor, R, for G.711 is calculated by Eq 5.3, where d is the end-to-end delay and e is the loss rate of packets. R ranges from 100 (desirable) to 0 (unacceptable). The relationship between the R factor and the MOS score is shown in Eq 5.4

$$R = 94.2 - 0.024d - 0.11(d - 177.3)H(d - 177.3) - 30ln(1 + 15e) \tag{5.3}$$

$$MOS = 1 + 0.035R + 7 * 10^6 R(R - 60)(100 - R) \tag{5.4}$$

5.3 Design of QoE Based Active Queue Management

5.3.1 Estimating the Queuing Delay

The proposed algorithm keeps track of the inter-arrival rate (λ) and service rate (μ) over a window of 10 packets, which means that it updates the monitored values every 10 packets. The average queuing delay is given by Eq 5.1. The load and mean service time are given by Eq 5.5 and Eq 5.6, where $\bar{\lambda}$ and $\bar{\mu}$ are the average value over 10 packets window. The square of coefficient of variation of the inter-arrival rate and service time are given by Eq 5.7 and Eq 5.8 respectively.

$$\rho = \frac{\bar{\lambda}}{\bar{\mu}} \tag{5.5}$$

$$\tau = \frac{1}{\bar{\mu}} \tag{5.6}$$

$$C_a^2 = Var(\frac{1}{\bar{\lambda}})\bar{\lambda}^2 \tag{5.7}$$

$$C_s^2 = Var(\frac{1}{\bar{\mu}})\bar{\mu}^2 \tag{5.8}$$

5.3.2 Dropping Policy

The proposed algorithm tracks the drops at RLC layer. Estimated queuing delay and packet loss probability are applied into Eq 5.3 in order to obtain the R factor, while the MOS score is given by Eq 5.4. According to [CR01] , the MOS score and the rating of service for VoIP traffic is as shown in Table 5-D. According to the required QoE level,

Quality of Voice Rating	**MOS**
Best	4.34 - 4.50
High	4.03 - 4.34
Medium	3.60 - 4.03
Low	3.10 - 3.60
Poor	2.58 - 3.10

Table 5-D: Quality Ratings and Associated MOS Score

the decision of whether to drop a packet is made. We assume that when there is no congestion, users will get at least a “high” level of service. When congestion happens, the QOE level starts to degrade due to increasing delay and drop of packets. When it drops below upper bound (4.03), the proposed algorithm checks the MOS score on the arrival of each packets. If the MOS score is still below the upper bound, it will drop 1 packet from the head of queue. If the MOS score drops below the lower bound, the algorithm stops dropping packets on realization that the congestion cannot be solved by actively dropping packets. The degradation of performance may be due to other reasons such as overloading of the network. To guarantee the connection, packets should be kept in the buffer instead of being discarded. Details of the algorithm are shown in Algorithm 4.

Algorithm 4 QoE Aware AQM for VoIP

On the arrival of every 10 packet:
    Update $\rho = \frac{\bar{\lambda}}{\bar{\mu}}$, $\tau = \frac{1}{\bar{\mu}}$
Calculate the square of coefficient of variation:
    $C_a^2 = Var(\frac{1}{\bar{\lambda}})\bar{\lambda}^2$
    $C_s^2 = Var(\frac{1}{\bar{\mu}})\bar{\mu}^2$
Calculate the MOS value by:
    $R = 94.2 - 0.024d - 0.11(d - 177.3)H(d - 177.3) - 30ln(1 + 15e)$ [CR01]
    $MOS = 1 + 0.035R + 7 * 10^6 R(R - 60)(100 - R)$ [CR01]
if 3.60 < MOS < 4.03 **then**
        Drop 1 packet from the head of queue
end if

5.4 Simulation Results and Discussions

5.4.1 Simulation Setup

The proposed algorithm is implemented in NS3. To keep consistent with the previous work in Chapter 4, the same topology is used and it is shown in Figure 5.5. The UEs are randomly distributed within 500 meters to 5000 meters by the “ns3::UniformRandomVariable” model. The number of UEs varies from 42 to 50 which is a typical value seen in one cell

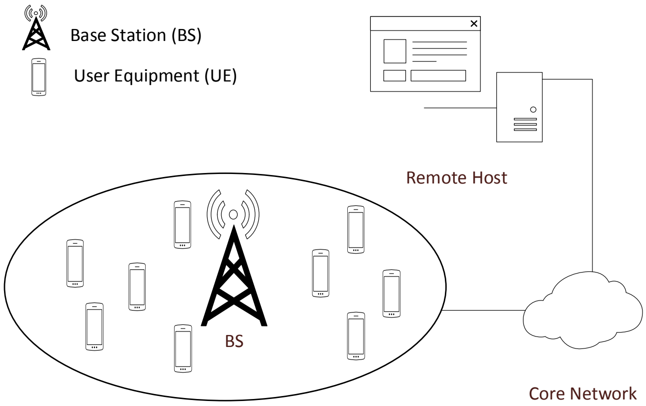

Figure 5.5: Simulation Topology.

in practice. The buffer at the RLC layer is set to 50 packets. The propagation delay is set to 50 ms and link rate is 100Mbps between the server and the core network. VoIP traffic is generated from the server to UE using ON-OFF traffic generator, with on time 0.96 seconds and off time 1.69 seconds. Parameters are listed in Table 5-E

Parameters	**Value**
Random Distribution Model	UniformRandomVariable
Server OnOffApplication	Mean $T_{on} = 0.96, T_{off} = 1.69$
Client PacketSink	UdpSocketFactory
Path Loss Model	FriisPropagationLossModel
Scheduling Algorithm	PfFfMacScheduler
Number of UE	42, 44, 46, 48, 50
Distance to Base Station	500 to 5000 meters
Server Bandwidth	10 Mbps

Table 5-E: Parameters in Simulation

5.4.2 Simulation Results

The performance of the proposed algorithm is evaluated in terms of the MOS score. As the MOS score is related to delay and loss, these two metrics are also evaluated. As shown in Figure 5.6, compared with CoDel, the average end-to-end delay is decreased by around 80%. However, this does come at the expense of increasing drops by around

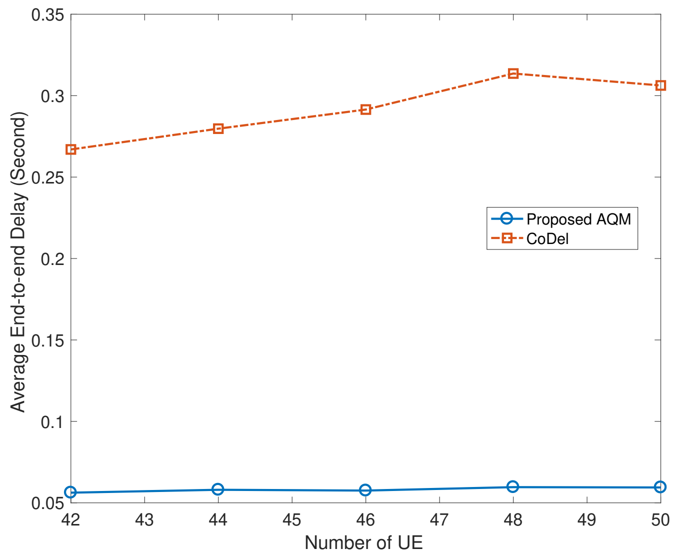

Figure 5.6: Average End-to-end Delay with Increasing Number of UEs.

70% as shown in Figure 5.7. There is a trade off between delay and loss; the MOS factor

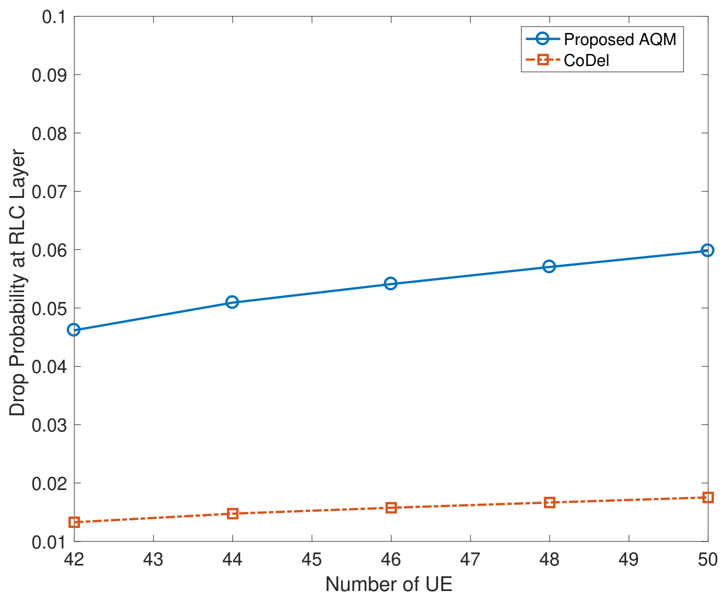

Figure 5.7: Drop Probability

plays an important role in balancing these two metrics. As shown in Figure 5.8, it can be seen that with the increasing number of UEs, the system becomes more congested as delay and loss both increase. And MOS value decreases for both the propsed algorithm and CoDel. The horizontal line is the lower bound of the medium level service. When there are more than 44 users, CoDel fails to keep the service level. However, the proposed algorithm successfully guarantees the service quality. Jain's fairness index is used to rate

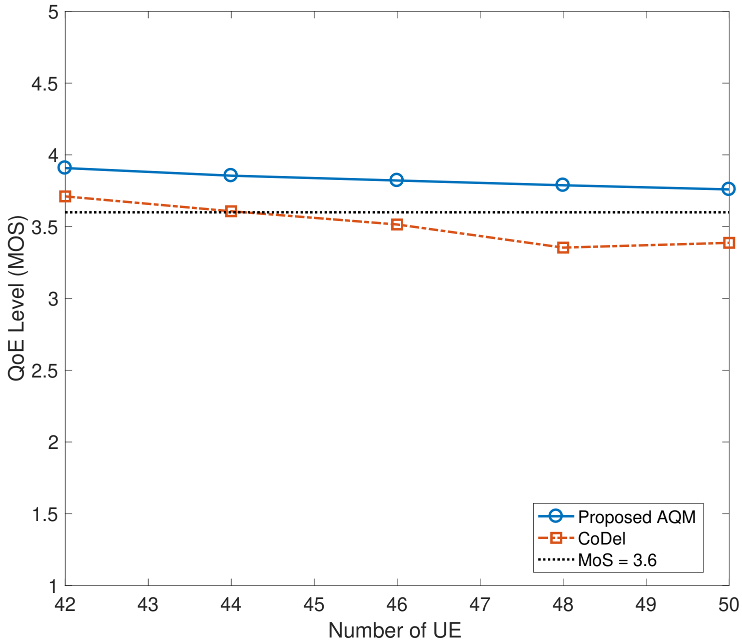

Figure 5.8: QoE Level

the fairness in a network when there are multiple uses [JCH84]. The Jain's fairness index results are shown in Figure 4.8. It can be seen that the proposed algorithm maintains similar fairness to that of CoDel. The strength of the proposed algorithm is summarized below.

- It monitors the real time inter-arrival and service time hence it is adaptive and can fit the fast-changing environment in cellular networks.
- The estimation of the queuing delay is solidly based on the classical queuing theory, i.e., G/G/1 queue.
- It has only one parameter to be set, i.e., the expected QoE level. With help of

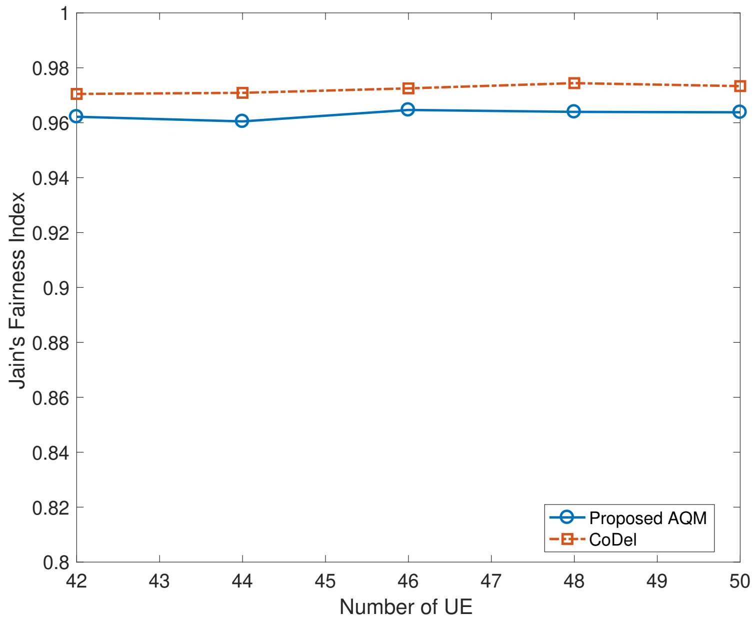

Figure 5.9: Jain's Fairness Index

QoE metric, delay and loss are automatically balanced. No complicated parameter settings are required.

- The proposed algorithm protects the connection when the system is heavily congested. Real time traffic flows are mostly based on UDP traffic which will not respond to drop of packets. Using other metrics will cause unnecessary drops which has no contribution to improve the QoE

5.5 Conclusions

In this chapter, a novel user experience based AQM is proposed aiming to improve the QoE of VoIP traffic. The proposed algorithm monitors QoE level instead of queuing delay or average queue length compared with traditional queue length. For VoIP traffic, the QoE is decided by both end-to-end delay and packet loss ratio. Compared with queuing delay, processing, proportionate and transmission delay can be ignored, hence reducing queuing delay can improve the QoE. Traditional AQMs reduce queuing delay by actively dropping packets which will increase the packet loss ratio and the overall QoE are not improved and can be even worse. As shown in Figure 5.8, although CoDel provides lower drop probability, but it cannot provide a better QoE compared with the proposed algorithm. The advantage of using the QoE metrics is that it can make a balance between the drop probability and the queuing delay. From the simulation results, it can be seen the QoE providing by CoDel decreases with the increasing number of users and it cannot maintain above the baseline. The QoE provides by the proposed algorithms also decreases a bit with the increasing number of users as the network is becoming more congested, but it still maintains above the baseline. However, the proposed algorithm is not as fair as CoDel as Jain's Fairness Index is based on the goodput. Dropping more packets can achieve higher QoE sometimes. Different users drop different number of packets to achieve better QoE and it is why the proposed algorithm is not as fair as CoDel. However, the Jain's Fairness Index still remains over 95% which is also good from the fairness perspective. Hence the proposed algorithm outperforms CoDel when the networks is congested by VoIP traffic.

Chapter 6

Active Queue Management for Dynamic Adaptive Video Streaming over HTTP

According to the forecast by Cisco [cic17], Video traffic through mobile networks will increase at a Compound Annual Growth Rate (CAGR) of 55% from 3660 to 33173 Petabytes (PB) per month. Video traffic, with no doubt, is already the dominated traffic in the Internet. As discussed in previous chapters, cellular access networks have high link rate with significant variations. To fit into the network scenario and assure user experience, Dynamic Adaptive video Streaming over HTTP (DASH) has been widely adopted by video service providers such as YouTube and NetFlix.

6.1 Introduction to DASH

Before turning to DASH, traditional real time video streaming uses Real-Time Streaming Protocol (RTSP) [SRL98]. RTSP is a stateful protocol and once the connection between the server and the client is established, the server will keep track of the state of the client until the end of the connection. The video content can be sent over TCP or UDP. If the video is not real time, another choice is progressive download which creates a copy of the video in the local host. Compared with stateful protocol, servers can give a quick response to HTTP requests, hence servers respond to DASH request quicker. Progressive download has several weaknesses. First it wastes bandwidth as the users may lose interest after watching a few frames. Secondly, it is not rate adaptive hence cannot fit into today's heterogeneous networks.

DASH [Sto11], is proposed by Moving Picture Experts Group (MPEG), also called MPEG DASH. It is a technology used to transmit video streaming with dynamic bit rate. An architecture is shown in Figure 6.1. A video content is splited into several

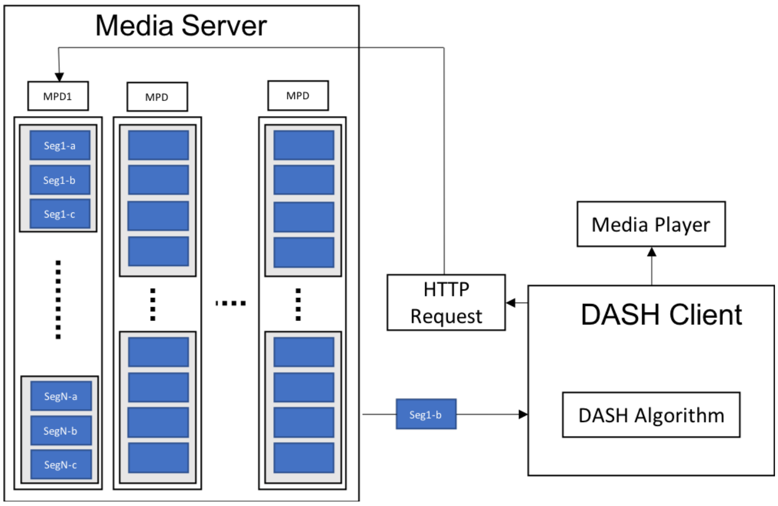

Figure 6.1: DASH Architecture

segments and each segment contains part of the video content. A video segment has several copies of different sizes in the sever representing different resolutions.

Media Presentation is defined in [ETS11] which is a structured collection of the media content. A media presentation contains Periods, Representations and Segments. The model of media presentation is shown in Figure 6.2.

- Period is the top level of media presentation. It describes a part of media content with the start time.

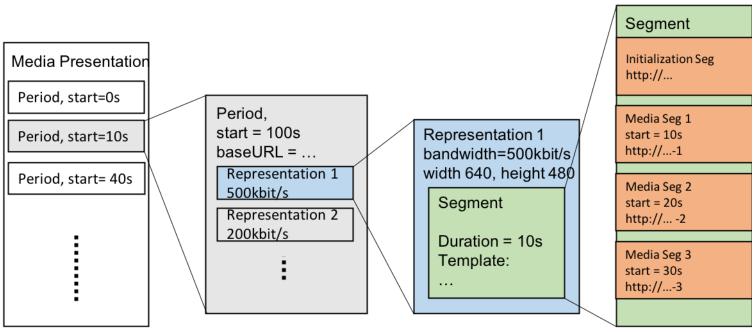

Figure 6.2: Media Presentation Model

- Representation contains same content in different codecs. Different representation contains same media content but in different resolutions. It allows users to request content with different quality that they can play without wasting bandwidth on extra pixels, e.g., a 1080p TV doesn't need 4k video.

- Segment is the media segments that actually being played by users. Segments locations and start time is described inside, as shown in Figure 6.2.

A media presentation is described in Media Presentation Description (MPD) which is stored in the sever together with the video segments. The MPD is the description file of all the video segments including timing, URL, bit rates and video resolution. DASH clients send requirements for next data segment according to the networks conditions. DASH is agnostic to application layer protocols and it is able to cooperate with any protocols.

6.2 Design and Implementation of AQM for DASH

According to the principle of DASH and state-of-the-art work, the requirements of designing an AQM for DASH are summarized here.

- Low latency. Reduce the latency of packet will reduce the initial delay which is a Key Performance Indicator (KPI) in DASH.

- Low Loss. Loss of packets will trigger the transition of resolution from high to low.

Previous work suggests using flow queuing to provide good isolation for DASH from other flows. In cellular networks, flow queuing is not necessary as flow isolation is guaranteed by the EPS bearer, mentioned in Chapter 5. Hence, the design of the AQM is focused on how to reduce latency and loss of packets. Previous AQMs normally monitor the queue at the network layer which is not suitable for cellular due to different network structures. [HKT+17] suggests deploying AQMs at MAC layer in Wi-Fi networks and use airtime fairness instead of Jain's Fairness Index. The airtime fairness will give each user same time slot to transmit data which seems fair. However, users with poor channel conditions will suffer a very low throughput. Inspired by [HKT+17], Bufferbloat issues can be controlled at other layers. For DASH, the latency can be controlled at the application layers by deploying an AQM. The benefits are listed below.

- Enough information. DASH clients communicate with the player directly hence it knows how many contents are waiting in the player's buffer.

- Avoid dropping. Existing AQM algorithms normally drop packets to reduce latency as routers and base stations are only responsible for forwarding packets. Deploying an AQM at application layer can control the requests from clients to servers.

- Work independently. DASH is based on TCP and there are works trying to control RWND [MVSA13] or block requests back to server [IJLB16]. Making a modification to TCP will affect other applications based on TCP and face a fairness issue competing with other TCP variant.

The architecture of DASH on the client node is shown in Figure 6.3. The received media content will be buffered and after all the segments are received, DASH client will requests new media content from the server. The duration that received contents can play is monitored by the dash client. The main idea of the proposed algorithm is to monitor the play out buffer by tracking the length of content received and played. Then a user can set a threshold to the play out buffer according to users' experience. The operation

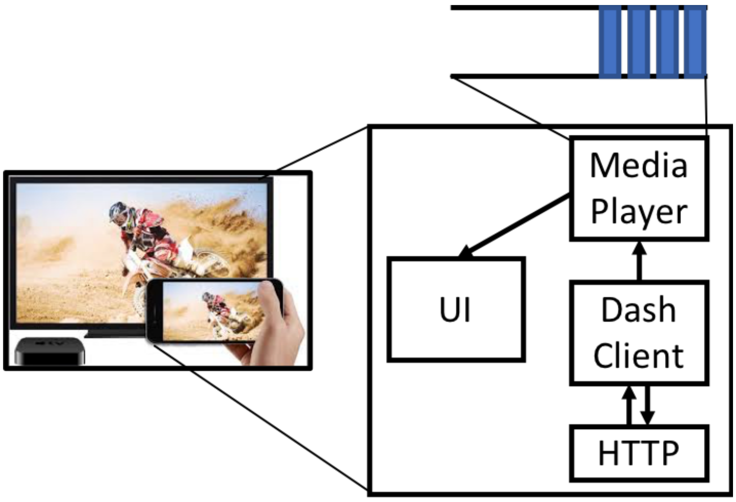

Figure 6.3: Dash Architecture on a Node

is the same one as a user can choose the quality of video when watching YouTube. If the buffer level reaches the threshold, the DASH client will back off a random time before sending the request to the server. The algorithm is shown in Algorithm 5. Today's

Algorithm 5 Random Back Off AQM for DASH Video

On the arrival of each packet:
Track the length of content in the arriving packet: $t_{buffered}$
Track the length of content in the packet forwarded to the player: t_{played}
$bufferDelay+ = t_{buffered} - t_{played}$
if $bufferDelay > Threshold$ **then**
    $t_{backOff} = U(0, 1)$
    $t_{request} = t_{request} + t_{backOff}$
end if

cellular networks have a large bandwidth but if all the users request data together, the buffer in the base station will be filled quickly and thus bring the Bufferbloat issue. The content buffered in the base station will be downloaded by the clients gradually, hence, waiting a short period of time will not incur stalling.

6.3 Simulations and Results

The proposed algorithm is implemented and tested in NS3 using a single cell topology. To keep consistent with previous work, the topology used here is the same as that in Chapter 4, as shown in Figure 6.4. To evaluate the performance of the proposed algorithm, the

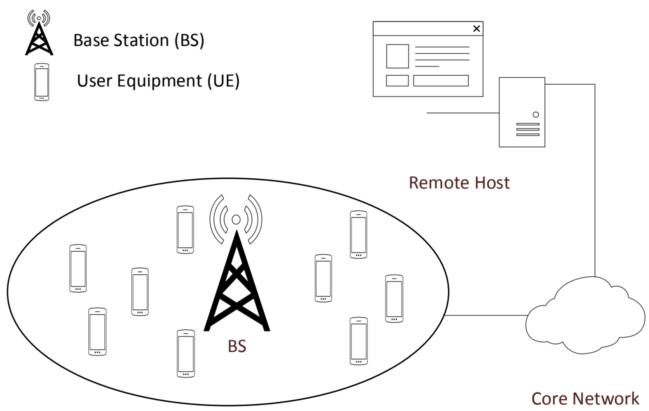

Figure 6.4: Simulation Topology

algorithm is tested in different scenarios. Common parameters used in all scenarios are shown in Table6-A.

Parameters	Value
Random Distribute Model	UniformRandomVariable
Server Application Model	DASH Server [VMS+16]
Client Application Model	DASH Client [VMS+16]
Path Loss Model	FriisPropagationLossModel
Scheduling Algorithm	PfFfMacScheduler
Server Bandwidth	10 Mbps

Table 6-A: Parameters in all Scenarios

6.3.1 Scenario I

In scenario I, UEs have same CQI value. Two sets of results are shown here. The CQI value of theses two sets are 15 and 8 respectively. A UE with CQI 15 means that the channel condition is very good and CQI 8 means the channel condition is medium. CQI is controlled by adjusting the distance between UEs and the base station. Detailed parameters are shown in Table 6-B and Table 6-C. For CQI 15, UEs are randomly allocated

Parameters	**Value**
Distance to Base Station	500 to 1000 meters
Number of UE	10, 20, 30, 40, 50

Table 6-B: Scenario I Parameters (CQI 15)

Parameters	**Value**
Distance to Base Station	9000 to 9500 meters
Number of UE	10, 20, 30, 40, 50

Table 6-C: Scenario I Parameters (CQI 8)

from 500 to 1000 meters. For CQI 8, UEs are randomly allocated from 9000 to 9500m, according to [1].

Figure 6.6 - 6.7 show the simulation results when all UEs have CQI 15. Figure 6.5 shows the average goodput of each flow. CoDel has the highest goodput when the number of

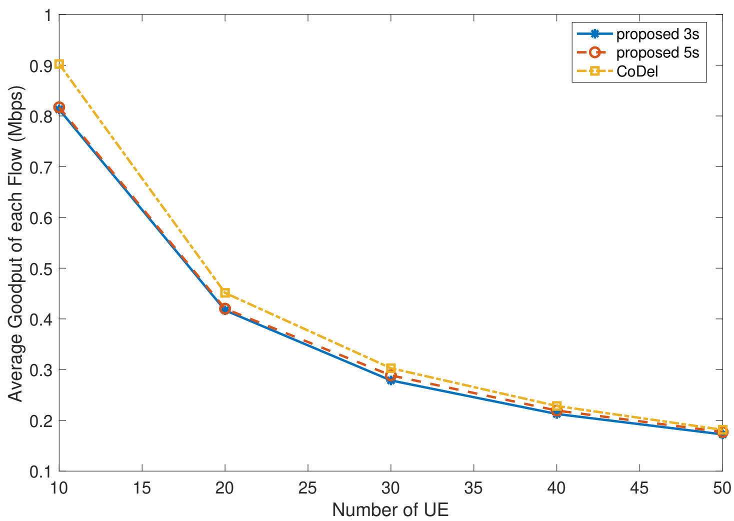

Figure 6.5: Average Goodput of each Flow with Increasing Number of UE (CQI 15)

UE is small. The proposed algorithm has similar goodput with threshold 3 seconds and 5 seconds. With the increasing number of UE, the goodput of each flow reduces as each

[1] The LTE signals can cover up to 100km [TIM+13].

UE needs to compete for the bandwidth.

Figure 6.6 shows the average end-to-end delay of each packet. From Figure 6.6, when

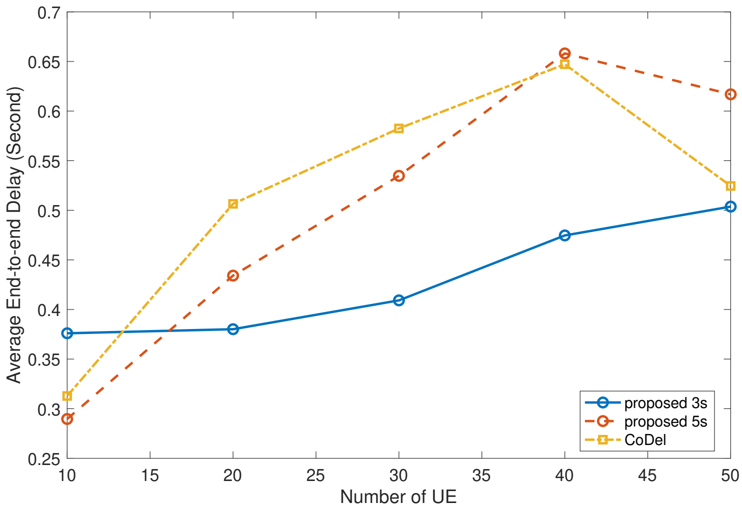

Figure 6.6: Average End-to-end Delay with Increasing Number of UE (CQI 15)

the threshold is set to different values, the proposed algorithm provides different performance. When there are 10 UEs, the average end-to-end delay is the lowest when the threshold of the play out buffer is set to 5 seconds. When there are 20 UEs, the average end-to-end delay with a threshold of 5 seconds becomes higher than that with the threshold of 3 seconds. With the increasing number of UEs, the average end-to-end delay keeps increasing until there are 40 UEs. When the number UEs increases to 50, the average end-to-end delay decreases compared with that when there are 40 UEs, if the threshold of the play out buffer is set to 5 seconds. When the threshold is set to 3 seconds, the average end-to-end delay increases compared with that when there are 40 UEs. CoDel gives similar performance to the proposed algorithm with 5 seconds buffer threshold regarding to average end-to-end delay. When there are more than 40 UEs, CoDel maintains a lower delay. When there are 10 UEs, CoDel maintains a lower delay compared to the proposed algorithm with 3 seconds target. From 40 UEs to 50 UEs, the average end-to-end delay decreases when CoDel and 5 seconds threshold is chosen. While the average end-to-end delay increases when 3 seconds threshold is chosen. The abnormal behaviours will be discussed later.

Figure 6.7 shows Jain's Fairness Index. When there are 10 and 20 UEs, both the pro-

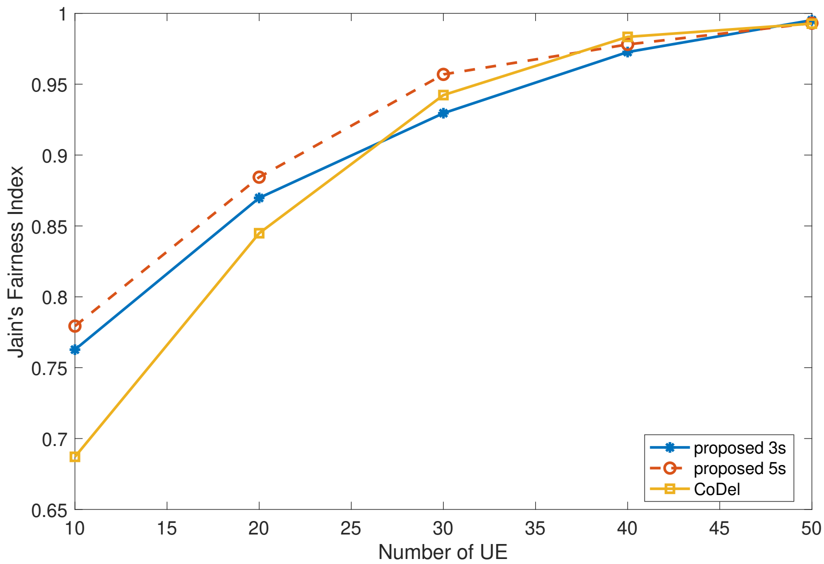

Figure 6.7: Jain's Fairness Index with Increasing Number of UE (CQI 15)

posed algorithm and CoDel are not fair. With the increasing number of UE, the proposed algorithm and CoDel becomes more and more fair. The reason of unfairness will be discussed later.

Figure 6.9 - Figure 6.10 show the simulation results when all UEs have CQI 8. Figure 6.8 shows the average goodput of each flow. CoDel has the highest goodput but the difference is very small when the number of UE is less 30. When the number of UE is over 30, the proposed algorithm with threshold of 3 seconds is a bit lower compared to that with threshold of 5 seconds and CoDel. The goodput of each flow reduces with the increasing number of UE as the bandwidth is shared by all UEs. Figure 6.9 shows the average end-to-end delay of each packet. When there are 10 UEs, the average end-to-end delay is similar. When there are 20 and 30 UEs, the proposed algorithm with 3 seconds delay gives much lower delay compared to that with 5 seconds and CoDel. Compared with 20 UEs, if the proposed algorithm with threshold of 3 seconds, the average end-to-end delay keeps increasing when there are 30 and 40 UEs. The results of average end-to-end

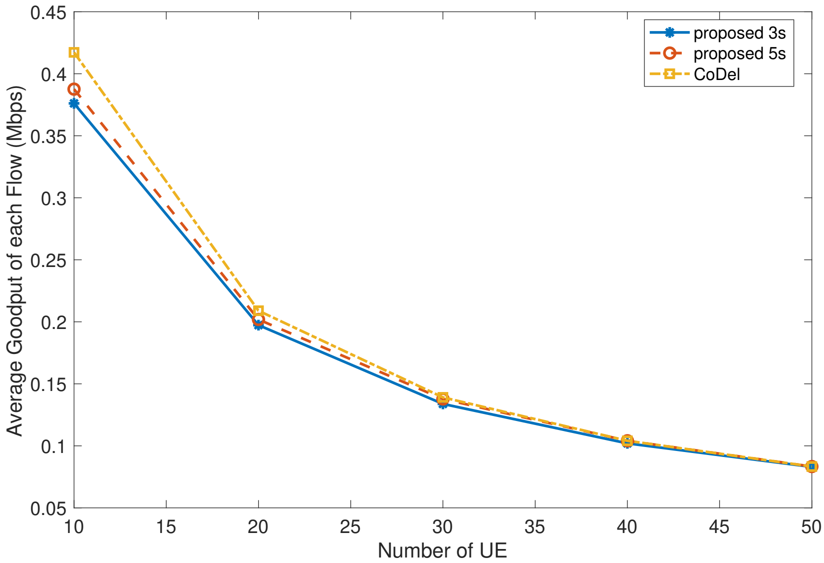

Figure 6.8: Average Goodput of each Flow with Increasing Number of UE (CQI 8)

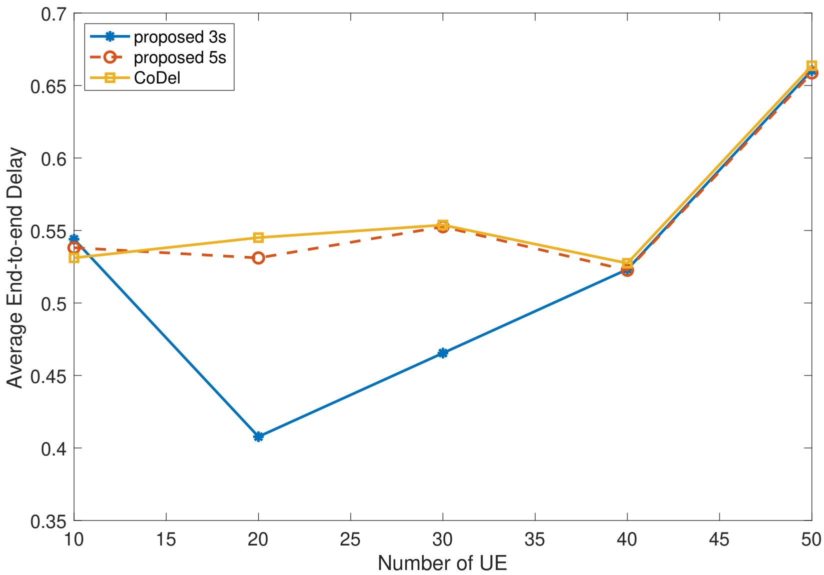

Figure 6.9: Average End-to-end Delay with Increasing Number of UE (CQI 8)

delay are quite similar when the number of UE is larger than 40. When there are enough RBs for all the UEs, the queuing delay of packets in the eNB is only because the link is saturated. When the RBs are not enough, the UEs in the system need wait for the

RBs to be scheduled, which adds additional queuing delay. When the number of UEs is increased to 20 from 10, the goodput of each UE decreases. And as the channel quality is not good, the server will not receive as many ACK as that in Figure 6.5. The packets in-flight are actually decreased so the average end-to-end delay is decreased. When the number of UEs continue to increase, although the packets in-flight are reduced, the queuing delay mainly comes from the time waiting to be scheduled for the RBs. It is why the average end-to-end delay increase again when the number of UEs is over 20. Insight discussion will be given later. Figure 6.10 shows Jain's Fairness Index. Different

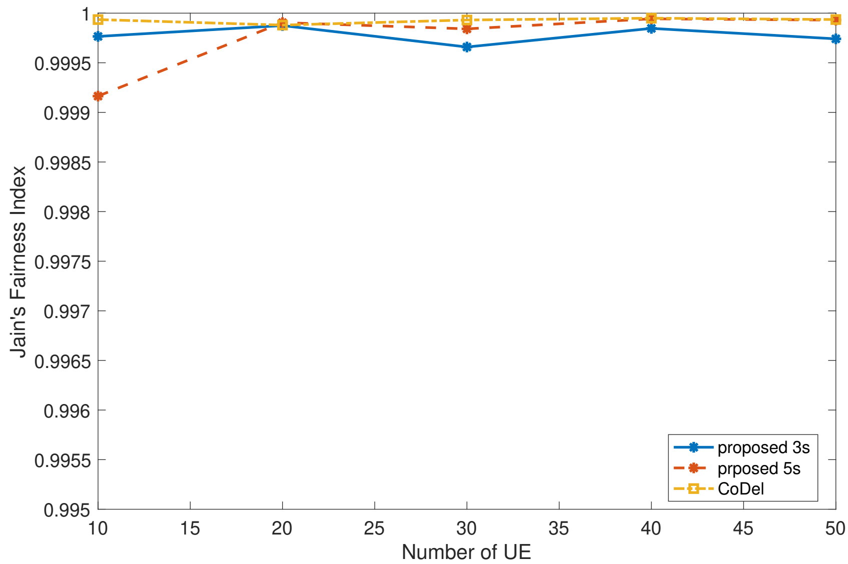

Figure 6.10: Jain's Fairness Index with Increasing Number of UE (CQI 8)

from the results of CQI 15, all the algorithms are very fair to each flow.

6.3.2 Scenario II

In Scenario I, all UEs in the simulation have same CQI. In the real world, cellular network users experience variable CQI. In this scenario, a realistic scenario is considered. The parameters are listed in Table 6-D. The number of UEs in this scenario is chosen from 42 to 50 and the step is 2 UEs. The UEs in the simulation have different CQI values and

Parameters	Value
Number of UE	42-50
Distance to Base Station	500 to 5000 meters
Server Bandwidth	10 Mbps

Table 6-D: Scenario II parameters (variant CQI)

the CQI values are controlled by changing the distance between the UEs and the base stations. Figure 6.11 shows the results of average end-to-end delay. The performance of

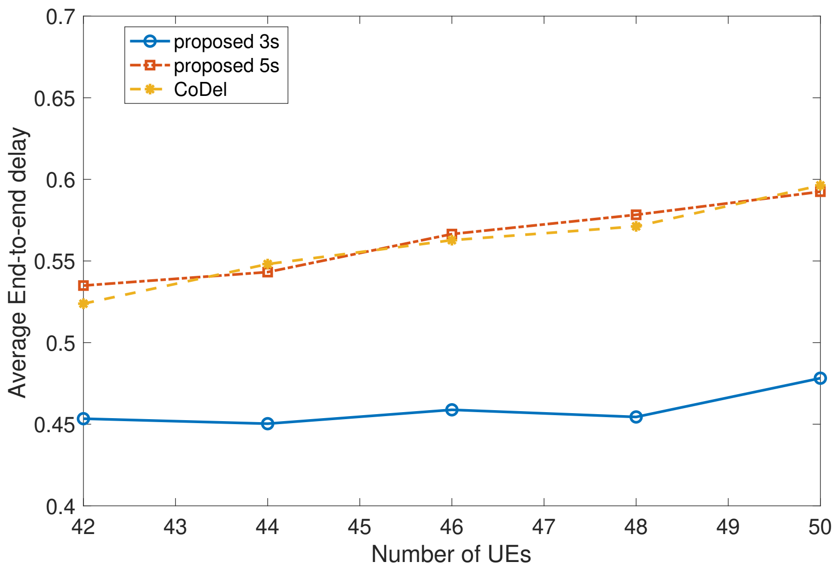

Figure 6.11: Average End-to-end Delay with Increasing Number of UE

the proposed algorithm with 5 seconds threshold is very similar compared with CoDel. With the increasing number of UEs, the average end-to-end delay gradually increases. When the threshold of the proposed algorithm is set to 3 seconds, the average end-to-end delay is reduced by around 100 milliseconds. Figure 6.12 shows the average goodput of each flow. The goodput reduces when the number of UEs increases. The average goodput of each flows is similar but CoDel provides the highest one. Figure 6.13 shows the results of Jain's Fairness Index. Although the UEs has different CQI values, but the goodput among them are fair.

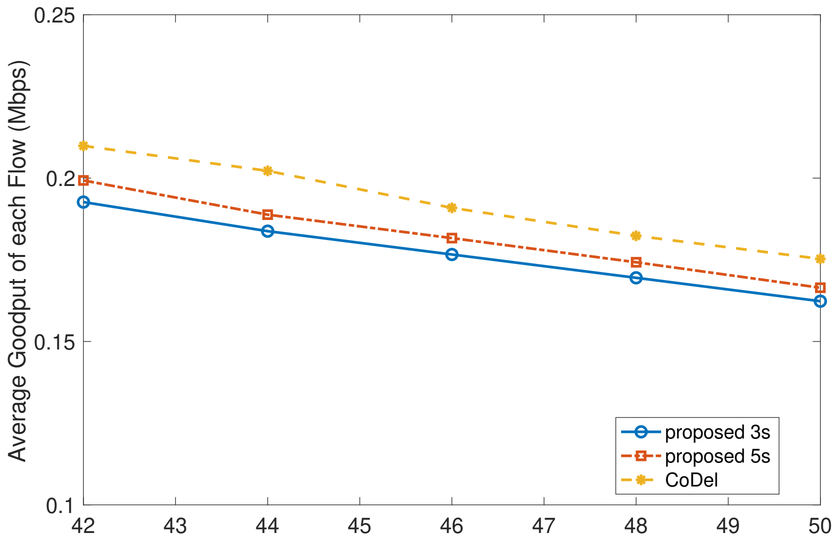

Figure 6.12: Average Goodput

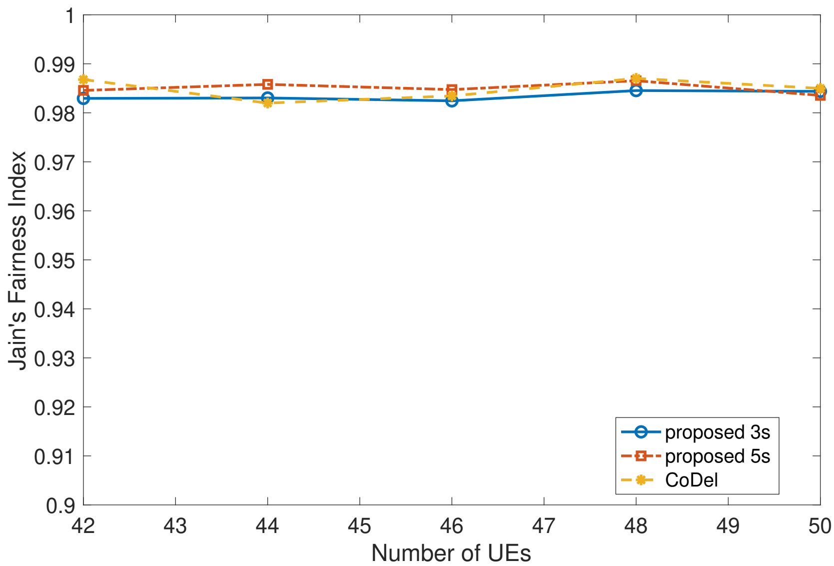

Figure 6.13: Jain's Fairness Index

6.4 Discussion

In this section, insight discussions focus on the proposed algorithm are given to illustrate the simulation results.

When there are 10 UEs, the average end-to-end delay in Figure 6.6 is different from that in Figure 6.9. The differences are more clear in Figure 6.14. The average end-to-end

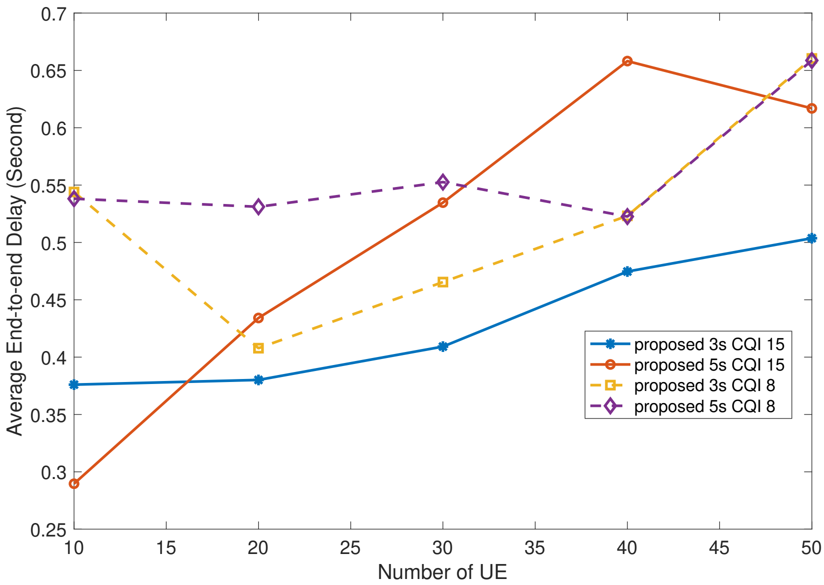

Figure 6.14: Average End-to-end Delay of the Proposed Algorithm with different Thresholds

delay with CQI 15 is lower compared to that with CQI 8. Figure 6.7 and Figure 6.10 show Jain's Fairness Index. When CQI is 15, the fairness index is getting higher with the increasing number of UEs. While when CQI is 8, the fairness index is high regardless of the number of UEs. From Figure 6.5 and Figure 6.8, it can be seen that the average goodput of each flow with CQI 15 is twice as much as that with CQI 8 and the goodput of each flow is negative related to the number of UEs in the system. When the UEs have higher CQI, the Jain's Fairness Index is worse compared to UEs with lower CQI. All the flows are transmitted using TCP. When a flow is punished by AQM, it will stop increasing the sending rate. In a single server topology with multiple TCP flows, the other flows have advantages in competing the bandwidth of the server. As a result, there are fairness issues. When the server have enough bandwidth, the fairness issue will be gone. Figure 6.15 shows Jain's Fairness Index when the bandwidth of the server is set to 100 Mbps. The other parameters in the simulation maintains the same value. It can be seen from Figure 6.15 that the goodput of each flow is fair regardless of the number of UEs. Figure 6.16 shows the average goodput of each flow when the server bandwidth

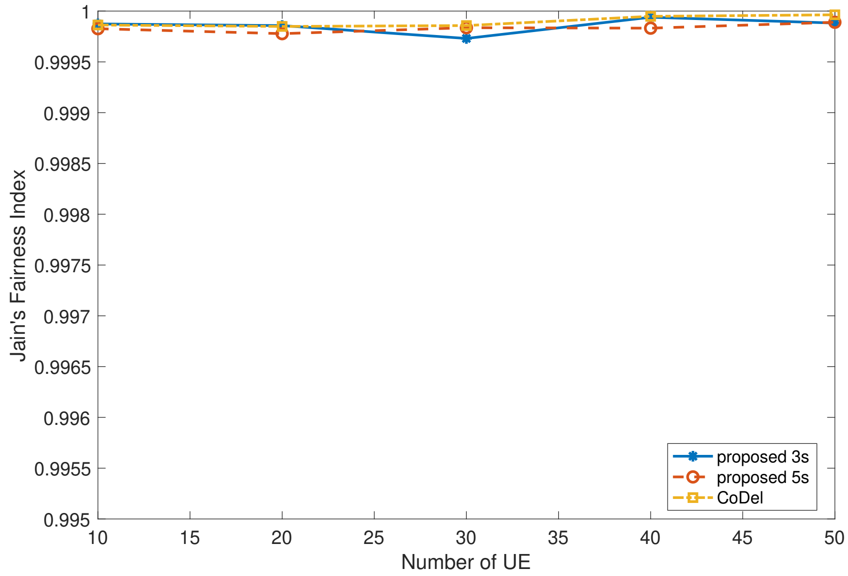

Figure 6.15: Jain's Fairness Index (server with 100 Mbps bandwidth)

increase to 100 Mbps. The average goodput of each flow is quite similar with previous results when the server bandwidth is set to 10 Mbps. It indicates that when there are 10 UEs in the system the maximum goodput is around 0.8 Mbps with the proposed algorithm. Figure 6.17 shows the average end-to-end delay of each packet when the

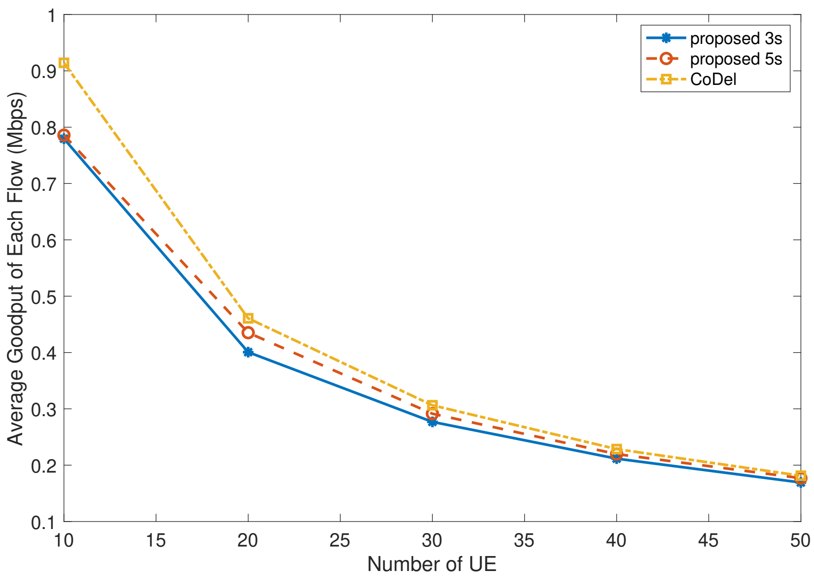

Figure 6.16: Average Goodput of each flow (server with 100 Mbps bandwidth)

server bandwidth is set to 100 Mbps. Compared with Figure 6.6, it can be seen the the average end-to-end delay is higher when the bandwidth of the server is 100 Mbps. It is because each flow has similar share of the bandwidth at the server side, as mentioned before. Each UE has similar amount of traffic waiting to be transmitted at the access network and the resource blocks in the base station need to be scheduled to server each UE.

When the number of UEs in the system increases to 20, the average end-to-end delay

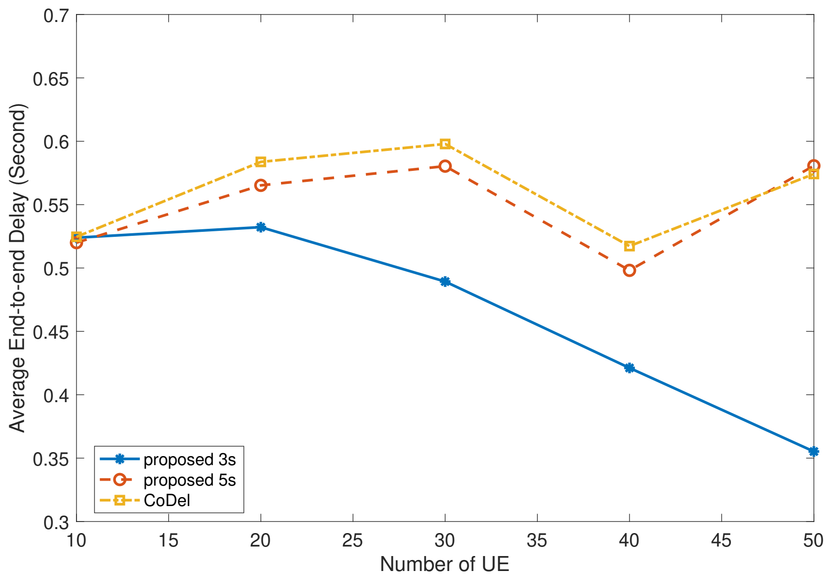

Figure 6.17: Average End-to-end Delay (server with 100 Mbps bandwidth)

decreases when the threshold is set to 3 seconds and the CQI is 8. Compared with the purple line in Figure 6.14, the average end-to-end delay is lower when the threshold is set to 3 seconds. It is because the proposed algorithm reacts quickly with a lower threshold. The proposed algorithm gives similar goodput with different threshold, as shown in Figure 6.8. When the CQI is 15, it is a bit of complex as fairness issues are seen, shown in Figure 6.7. Although the average goodput of each flow is the same with different threshold, the goodput of each flow varies a lot.

Figure 6.18 shows the average end-to-end delay when the bandwidth of the server is different. When there are small numbers of UEs in the system and the server has 100

Mbps bandwidth, there are more packets in flight which results in higher average end-to-end delay. However, with the increasing number of UEs, the in-flight packets continues to trigger the proposed algorithm. As a results, the average end-to-end delay decreases with the increasing number of UEs. While when the threshold is set to 5 seconds, the number of in-flight packet is not enough to trigger the proposed algorithm, which leads to the increase of average end-to-end delay if the UEs in the systems are over 40, as seen in Figure 6.17. When the CQI is 8 and there are more than 40 UEs in the system, the proposed algorithm is not frequently triggered, shown in Figure 6.14. That is why the yellow line and the purple line are overlapped

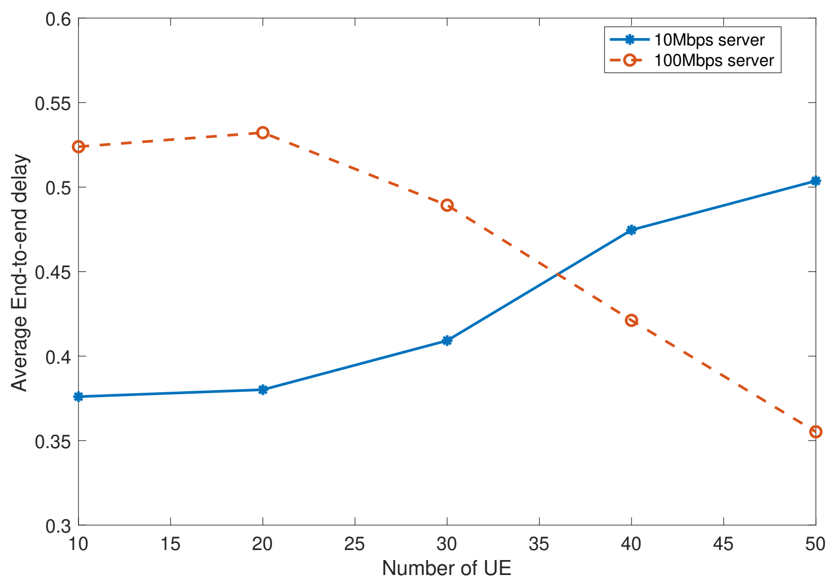

Figure 6.18: Average End-to-end Delay (server with different bandwidth)

6.5 Conclusions

This chapter introduces the character of DASH algorithm and the design principle to design an AQM for DASH in cellular networks. An AQM for DASH traffic is proposed and tested using NS3 simulator. It is worth mentioning that the proposed algorithm also works in wired scenarios. The reason to focus in cellular networks is because there are many users with in a single cell watching DASH contents and hence results in experiencing large queuing delay. In wired networks scenarios, there are fewer users and large bandwidth for each user.

There are actually 3 scenarios tested. In scenario I, the effect of CQI is analysed. In scenario II, UEs are with different CQI values and the effect of the bandwidth at the server side is also tested. Simulation results show that the proposed algorithm has better performance in controlling the average end-to-end delay. It is true that streaming videos are not sensitive to delay as long as no interruptions are seen. However, lower average end-to-end delay means lower initial delay when watching a video and the initial delay is related to the QoE of users. And if there are other users in the system browsing web page or playing games, the average end-to-end delay is also reduced. The proposed algorithm reduced the average end-to-end delay by controls the request to the server instead of dropping packets, it avoids retransmission. Traditional AQMs monitors the buffer at routers or base stations. Neither queue length or queuing delay is meaningful to a video user. The proposed algorithm monitors the play out buffer. If a user don't want to buffer extra contents and wants to speed up the response of the network, the user can set the threshold easily.

Chapter 7

Conclusions and future work

7.1 Summary

Bufferbloat issues have been proved to be existing in all kinds networks. Large queuing delay incurred by excessive buffering results in degraded performance for real time applications such as (VoIP and DASH videos). New type of entertainment, such as live show broadcast by phone, and new online games, such as PlayerUnknown's Battlegrounds, keep emerging. Buffering is necessary prevent the loss of packet and to absorb burst traffic. However, without carefully management, buffering will incur unwanted delay.

This work mitigates the bufferbloat issue in cellular networks using AQM algorithms. Tradition AQMs using either length of the queue or waiting time of packets in the queue as the metric to making dropping decisions. This work considers CQI which are unique in mobile broadband (MBB) networks. Involving CQI can differentiate UEs with different channel quality. It protect the UEs with worse channel.

Even through CQI is involved and it gives better performance from the perspective of average end-to-end delay, end-to-end delay is obscure to a user. AQMs reduces the queuing delay by actively dropping packets. To balance the drop and the delay, the QoE metric is involved in the AQM for VoIP traffic. This AQM assumes that a user has the

QoE above Medium when there is no congestion. When the QoE drops below High level, the algorithm start dropping packets in order to reduce the delay.

Different applications have different requirement of QoE. DASH video is widely used in today's Internet however evaluating the QoE of dash video is complex. To reduce the delay brought by DASH packets, a new AQM using random back off algorithm is proposed. It monitors the play out buffer and when the content buffered reaches threshold, the AQM will hold the request for next packets for a random time. It avoids dropping packets and the retransmission due to drop of packets. The proposed algorithm outperforms CoDel in reducing the average end-to-end delay and meanwhile maintains similar goodput.

7.2 Key contributions

The major contributions in this thesis are detailed below.

- Cross layer design is involved. The Channel quality aware AQM use CQI. Using CQI is to protect UEs with poor channel quality and meanwhile, the proposed algorithm reduced latency in the network.

- QoE metric is involved. The AQM for VoIP involves QoE in making the dropping decisions. Compared with existing AQMs, it automatically makes the balance between drop and delay.

- Indirect control of latency. For specific type of traffic such as DASH, it is hard to control the delay directly where it happens due to lack of information. In such occasions, the AQM can be deployed at other layer to indirectly reduce the latency.

Deploying AQMs can control the queuing delay. Different AQMs have different advantages and disadvantages. An AQM controlled by queue length is not sensitive to queuing delay. An AQM controlled by queuing delay gives good performance from the respective of delay but not all the packets requires low queuing delay. For specific user scenarios, clear recommendations can be given.

7.3 Future work

Bufferbloat is vast topic involving different type of traffic, different traffic control technologies and protocols. Next Generation Networks (NGN) has larger bandwidth and interactive application keeps increasing. Mitigating bufferbloat need carefully manage the buffer which requires correct and precise modelling of traffic going through the network. Machine learning provides a new aspect of modelling traffic and mapping between network performances with QoE.

Cellular networks have been an important port in our daily life and it is more complex compared with wired and Wi-Fi networks. The number of users in one cell is changing and the wireless channel is time variant. It will be more challenging when users in multiple cells are considered. A general model to further reduce latency in cellular networks still needs to be revealed. The cooperation of different layers in dealing with bufferbloat need to be further investigated.

Appendix A

Author's publications

Conference papers

1. **Y. Dai**, V. Wijeratne, Y. Chen and J. Schormans, "Channel quality aware active queue management in cellular networks", *2017 9th Computer Science and Electronic Engineering (CEEC)*, Colchester, 2017, pp. 183-188.

2. **Y. Dai**, V. Wijeratne, Y. Chen and J. Schormans, "User Experience Aware Active Queue Management in Cellular Networks" accepted by Journal of Communication.

3. **Y. Dai**, V. Wijeratne, Y. Chen and J. Schormans, "Active Queue Management for Dynamic Adaptive Video Streaming over HTTP" submited to IET Journal of Engineering.

References

[AB14] Ekta Agrawal and Kanojia Sindhuben Babula. Evaluation of voice codecs of voip applications for manet. *International Journal of Current Engineering and Technology*, 4(4), 2014.

[AC17] G. Armitage and R. Collom. Benefits of flowqueue-based active queue management for interactive online games. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9, July 2017.

[ACA96] A. Arulambalam, Xiaoqiang Chen, and N. Ansari. Allocating fair rates for available bit rate service in atm networks. *IEEE Communications Magazine*, 34(11):92–100, Nov 1996.

[AEN16] M. A. Ali, A. Esmailpour, and N. Nasser. Performance evaluation for lte applications with buffer awareness consideration. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–7, April 2016.

[AG16] M. Ansari and M. Ghaderi. Parallel http for video streaming in wireless networks. In *2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 337–342, Sept 2016.

[AGG+13] S. Alfredsson, G. Del Giudice, J. Garcia, A. Brunstrom, L. De Cicco, and S. Mascolo. Impact of tcp congestion control on bufferbloat in cellular networks. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, pages 1–7, June 2013.

[AL18] Dhulfiqar A. Alwahab and Sándor Laki. A simulation-based survey of active queue management algorithms. In *Proceedings of the 6th International Conference on Communications and Broadband Networking*, ICCBN 2018, pages 71–77, New York, NY, USA, 2018. ACM.

[ALLY01] S. Athuraliya, S. H. Low, V. H. Li, and Qinghe Yin. Rem: active queue management. *IEEE Network*, 15(3):48–53, May 2001.

[AOAL17] A. M. Alkharasani, M. Othman, A. Abdullah, and K. Y. Lun. An improved quality-of-service performance using red x2019;s active queue management flow control in classifying networks. *IEEE Access*, 5:24467–24478, 2017.

[AR04] C. R. Anderson and T. S. Rappaport. In-building wideband partition loss measurements at 2.5 and 60 ghz. *IEEE Transactions on Wireless Communications*, 3(3):922–928, May 2004.

[ASAB17] R. Al-Saadi, G. Armitage, and J. But. Characterising ledbat performance through bottlenecks using pie, fq-codel and fq-pie active queue management. In *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, pages 278–285, Oct 2017.

[ASC08] A. V. Alejos, M. G. Sanchez, and I. Cuinas. Measurement and analysis of propagation mechanisms at 40 ghz: Viability of site shielding forced by obstacles. *IEEE Transactions on Vehicular Technology*, 57(6):3369–3380, Nov 2008.

[BHL+14] H. Baligh, M. Hong, W. C. Liao, Z. Q. Luo, M. Razaviyayn, M. Sanjabi, and R. Sun. Cross-layer provision of future cellular networks: A wmmse-based approach. *IEEE Signal Processing Magazine*, 31(6):56–68, Nov 2014.

[BNPF17] J. D. Beshay, A. T. Nasrabadi, R. Prakash, and A. Francini. On active queue management in cellular networks. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 384–389, May 2017.

[BOP94] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. Tcp vegas: New techniques for congestion detection and avoidance. *SIGCOMM Comput. Commun. Rev.*, 24(4):24–35, October 1994.

[BP95] 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*, 13(8):1465–1480, Oct 1995.

[CBHB16] Q. D. Coninck, M. Baerts, B. Hesmans, and O. Bonaventure. Observing real smartphone applications over multipath tcp. *IEEE Communications Magazine*, 54(3):88–93, March 2016.

[Cer14] V. G. Cerf. Bufferbloat and other internet challenges. *IEEE Internet Computing*, 18(5):80–80, Sept 2014.

[CGM+02] Claudio Casetti, Mario Gerla, Saverio Mascolo, M. Y. Sanadidi, and Ren Wang. Tcp westwood: End-to-end congestion control for wired/wireless networks. *Wirel. Netw.*, 8(5):467–479, September 2002.

[Cha17] T. J. Chaudhery. Performance evaluation of codel queue mechanism and tfrc transport protocol when using voip flows. In *2017 International Conference on Frontiers of Information Technology (FIT)*, pages 1–6, Dec 2017.

[cic17] Cisco visual networking index: Global mobile data traffic forecast update, 2016 - 2021. White Paper, June 2017.

[CJWG11] Vint Cerf, Van Jacobson, Nick Weaver, and Jim Gettys. . *Queue*, 9(12):10:10–10:20, December 2011.

[CL13] J. Chen and V. K. N. Lau. Large deviation delay analysis of queue-aware multi-user mimo systems with two timescale mobile-driven feedback. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5036–5040, May 2013.

[CLG+13] Yung-Chih Chen, Yeon-sup Lim, Richard J. Gibbens, Erich M. Nahum, Ramin Khalili, and Don Towsley. A measurement-based study of multipath tcp performance over wireless networks. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 455–468, New York, NY, USA, 2013. ACM.

[CPG+13] 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*, 15(2):678–700, Second 2013.

[CR01] R. G. Cole and J. H. Rosenbluth. Voice over ip performance monitoring. *SIGCOMM Comput. Commun. Rev.*, 31(2):9–24, April 2001.

[CR15] S. Chavan and G. Raina. Performance of tcp with a proportional integral enhanced (pie) queue management policy. In *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pages 1013–1018, May 2015.

[CT14] Y. C. Chen and D. Towsley. On bufferbloat and delay analysis of multipath tcp in wireless networks. In *2014 IFIP Networking Conference*, pages 1–9, June 2014.

[DK13] M. K. Dadhania and K. V. Kumar. Modified red algorithm to improve the performance of web traffic. In *Advanced Computing and Communication Technologies (ACCT), 2013 Third International Conference on*, pages 187–194, April 2013.

[DKS89] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *SIGCOMM Comput. Commun. Rev.*, 19(4):1–12, August 1989.

[DYYL14] Yi Deng, Xiaobo Yu, Qinmin Yang, and Jiangang Lu. On qos performance of different voice codecs over ad hoc networks in underground mines. In *Control Conference (CCC), 2014 33rd Chinese*, pages 5521–5526. IEEE, 2014.

[EBBEBB13] Said El Brak, Mohammed Bouhorma, Mohamed El Brak, and Anouar Bohdhir. Speech quality evaluation based codec for voip over 802.11 p. *International Journal of Wireless & Mobile Networks*, 5(2):59, 2013.

[ETS] TS ETSI. 136 213 v11. 3.0 (3gpp ts 36.213 version 11.3. 0 release 11). *LTE.,(Jul. 2013)*, 178.

[ETS11] TS ETSI. 26.234 v10. 2.0 (rel. 10). *UMTS-Transparent End-to-End Packet-Switched Streaming Service (PSS)*, 2011.

[FGS+01] Sally Floyd, Ramakrishna Gummadi, Scott Shenker, et al. Adaptive red: An algorithm for increasing the robustness of red's active queue management, 2001.

[FHXC15] C. W. Feng, L. F. Huang, C. Xu, and Y. C. Chang. Congestion control scheme performance analysis based on nonlinear red. *IEEE Systems Journal*, PP(99):1–8, 2015.

[FJ93] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, Aug 1993.

[FKSS01] Wu-Chang Feng, D. D. Kandlur, D. Saha, and K. G. Shin. Stochastic fair blue: a queue management algorithm for enforcing fairness. In *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, volume 3, pages 1520–1529 vol.3, 2001.

[FODA14] S. Ferlin-Oliveira, T. Dreibholz, and Ö. Alay. Tackling the challenge of bufferbloat in multi-path transport over heterogeneous wireless networks. In *2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)*, pages 123–128, May 2014.

[FRHB13] Alan Ford, Costin Raiciu, Mark Handley, and Olivier Bonaventure. Tcp extensions for multipath operation with multiple addresses. Technical report, 2013.

[FSKN17] B. Felix, A. Santos, B. Kantarci, and M. Nogueira. Cd-asm: A new queuing paradigm to overcome bufferbloat effects in hetnets. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6, Oct 2017.

[FSKS02] Wu-chang Feng, Kang G. Shin, Dilip D. Kandlur, and Debanjan Saha. The blue active queue management algorithms. *IEEE/ACM Trans. Netw.*, 10(4):513–528, August 2002.

[Get11] J. Gettys. Bufferbloat: Dark buffers in the internet. *IEEE Internet Computing*, 15(3):96–96, May 2011.

[GPKC17] C. A. Grazia, N. Patriciello, M. Klapez, and M. Casoni. Mitigating congestion and bufferbloat on satellite networks through a rate-based aqm. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2017.

[GQC+16] Yihua Guo, Feng Qian, Qi Alfred Chen, Zhuoqing Morley Mao, and Subhabrata Sen. Understanding on-device bufferbloat for cellular upload. In

Proceedings of the 2016 Internet Measurement Conference, IMC '16, pages 303–317, New York, NY, USA, 2016. ACM.

[HHS17] L. A. Haibeh, N. Hakem, and O. A. Safia. Performance evaluation of voip calls over manet for different voice codecs. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1–6, Jan 2017.

[HJHB15] Toke H iland-J rgensen, Per Hurtig, and Anna Brunstrom. The good, the bad and the wifi: Modern {AQMs} in a residential setting. *Computer Networks*, 89:90 – 106, 2015.

[HJKT+17] Toke Høiland-Jørgensen, Michał Kazior, Dave Täht, Per Hurtig, and Anna Brunstrom. Ending the anomaly: Achieving low latency and airtime fairness in wifi. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*, pages 139–151, Santa Clara, CA, 2017. USENIX Association.

[HJMT+14] T Hoeiland-Joergensen, P McKenney, D Taht, J Ghettys, and E Dumazet. Flowqueue-codel: draft-hoeiland-joergensen-aqm-fq-codel-00. Technical report, Internet-draft, IETF, 2014.

[HK04] Liujia Hu and A. D. Kshemkalyani. Hred: a simple and efficient active queue management algorithm. In *Computer Communications and Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference on*, pages 387–393, Oct 2004.

[HKT+17] Toke Høiland-Jørgensen, Michal Kazior, Dave Täht, Per Hurtig, and Anna Brunstrom. Ending the anomaly: Achieving low latency and airtime fairness in wifi. *CoRR*, abs/1703.00064, 2017.

[HP92] Peter G. Harrison and Naresh M. Patel. *Performance Modelling of Communication Networks and Computer Architectures (International Computer S*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1992.

[HRX08] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: A new tcp-friendly high-speed tcp variant. *SIGOPS Oper. Syst. Rev.*, 42(5):64–74, July 2008.

[HSBP13] Tobias Hoßfeld, Raimund Schatz, Ernst Biersack, and Louis Plissonneau. Internet video delivery in youtube: from traffic measurements to quality of experience. In *Data Traffic Monitoring and Analysis*, pages 264–301. Springer, 2013.

[HSVT12] T. Hobfeld, R. Schatz, M. Varela, and C. Timmerer. Challenges of qoe management for cloud applications. *IEEE Communications Magazine*, 50(4):28–36, April 2012.

[HXT+10] Junxian Huang, Qiang Xu, Birjodh Tiwana, Z. Morley Mao, Ming Zhang, and Paramvir Bahl. Anatomizing application performance differences on smartphones. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 165–178, New York, NY, USA, 2010. ACM.

[IJLB16] H. Im, C. Joo, T. Lee, and S. Bahk. Receiver-side tcp countermeasure to bufferbloat in wireless access networks. *IEEE Transactions on Mobile Computing*, 15(8):2080–2093, Aug 2016.

[IKVF15] N. Iya, N. Kuhn, F. Verdicchio, and G. Fairhurst. Analyzing the impact of bufferbloat on latency-sensitive applications. In *2015 IEEE International Conference on Communications (ICC)*, pages 6098–6103, June 2015.

[Jac88] V. Jacobson. Congestion avoidance and control. *SIGCOMM Comput. Commun. Rev.*, 18(4):314–329, August 1988.

[JBT14] T. Jain, A. B., and M. P. Tahiliani. Performance evaluation of codel for active queue management in wired-cum-wireless networks. In *2014 Fourth International Conference on Advanced Computing Communication Technologies*, pages 381–385, Feb 2014.

[JCH84] Raj Jain, Dah-Ming Chiu, and William R Hawe. *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*, volume 38. Eastern Research Laboratory, Digital Equipment Corporation Hudson, MA, 1984.

[JDSL14] J. M. Jiménez, J. R. Diaz, S. Sendra, and J. Lloret. Choosing the best video compression codec depending on the recorded environment. In *2014*

IEEE Global Communications Conference, pages 1347–1352, Dec 2014.

[JK14] I. Järvinen and M. Kojo. Evaluating codel, pie, and hred aqm techniques with load transients. In *39th Annual IEEE Conference on Local Computer Networks*, pages 159–167, Sept 2014.

[JK17] I. Järvinen and M. Kojo. Gazing beyond horizon: The predict active queue management for controlling load transients. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 126–135, March 2017.

[JLW+12] Haiqing Jiang, Zeyu Liu, Yaogong Wang, Kyunghan Lee, and Injong Rhee. Understanding bufferbloat in cellular networks. In *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, CellNet '12, pages 1–6, New York, NY, USA, 2012. ACM.

[JS17] Ahmed Jameel and Maryam Shafei. Qos performance evaluation of voice over lte network. 06, 01 2017.

[JWLR12] Haiqing Jiang, Yaogong Wang, Kyunghan Lee, and Injong Rhee. Tackling bufferbloat in 3g/4g networks. In *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, pages 329–342, New York, NY, USA, 2012. ACM.

[JWLR16] H. Jiang, Y. Wang, K. Lee, and I. Rhee. Drwa: A receiver-centric solution to bufferbloat in cellular networks. *IEEE Transactions on Mobile Computing*, 15(11):2719–2734, Nov 2016.

[KA17] J. Kua and G. Armitage. Optimising dash over aqm-enabled gateways using intra-chunk parallel retrieval (chunklets). In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9, July 2017.

[KAB16] J. Kua, G. Armitage, and P. Branch. The impact of active queue management on dash-based content delivery. In *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, pages 121–128, Nov 2016.

[KHH+12] Mohammad T Kawser, Nafiz Imtiaz Bin Hamid, Md Nayeemul Hasan, M Shah Alam, and M Musfiqur Rahman. Downlink snr to cqi mapping for different multipleantenna techniques in lte. *International Journal of Information and Electronics Engineering*, 2(5):757, 2012.

[KKFR15] C. Kulatunga, N. Kuhn, G. Fairhurst, and D. Ros. Tackling bufferbloat in capacity-limited networks. In *2015 European Conference on Networks and Communications (EuCNC)*, pages 381–385, June 2015.

[Kli17] R. Klimek. Context-aware and pro-active queue management systems in intelligent environments. In *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 1077–1084, Sept 2017.

[KLM14] Nicolas Kuhn, Emmanuel Lochin, and Olivier Mehani. Revisiting old friends: Is codel really achieving what red cannot? In *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, CSWS '14, pages 3–8, New York, NY, USA, 2014. ACM.

[KNAB17] J. Kua, S. H. Nguyen, G. Armitage, and P. Branch. Using active queue management to assist iot application flows in home broadband networks. *IEEE Internet of Things Journal*, 4(5):1399–1407, Oct 2017.

[KRB+17] Nicolas Kuhn, David Ros, Amadou Baba Bagayoko, Chamil Kulatunga, Gorry Fairhurst, and Naeem Khademi. Operating ranges, tunability and performance of codel and pie. *Computer Communications*, 103:74 – 82, 2017.

[KSC91] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis. Weighted round-robin cell multiplexing in a general-purpose atm switch chip. *IEEE Journal on Selected Areas in Communications*, 9(8):1265–1279, Oct 1991.

[KWL15] M. L. Ku, L. C. Wang, and Y. L. Liu. Joint antenna beamforming, multiuser scheduling, and power allocation for hierarchical cellular systems. *IEEE Journal on Selected Areas in Communications*, 33(5):896–909, May 2015.

[LCTA12] Jaime Lloret, Alejandro Canovas, Jesus Tomas, and Marcelo Atenas. A network management algorithm and protocol for improving qoe in mobile iptv. *Computer Communications*, 35(15):1855 – 1870, 2012. Smart and

Interactive Ubiquitous Multimedia Services.

[LHG15] Y. Liu, M. Huynh, and D. Ghosal. Efficient downlink scheduling with discontinuous reception (drx) mechanism in lte network. In *2015 International Conference on Computing, Networking and Communications (ICNC)*, pages 1019–1023, Feb 2015.

[LL15] Jin Seong Lee and Jaiyong Lee. Multipath tcp performance improvement in mobile network. In *2015 Seventh International Conference on Ubiquitous and Future Networks*, pages 710–714, July 2015.

[LM97] Dong Lin and Robert Morris. Dynamics of random early detection. In *Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '97, pages 127–137, New York, NY, USA, 1997. ACM.

[LRK+99] Reiner Ludwig, Bela Rathonyi, Almudena Konrad, Kimberly Oden, and Anthony Joseph. Multi-layer tracing of tcp over a reliable wireless link. *SIGMETRICS Perform. Eval. Rev.*, 27(1):144–154, May 1999.

[LSCP12] Jonathan S Lu, Daniel Steinbach, Patrick Cabrol, and Philip Pietraski. Modeling human blockers in millimeter wave radio links. *ZTE Communications*, 10(4):23–28, 2012.

[LZGS84] Edward D. Lazowska, John Zahorjan, G. Scott Graham, and Kenneth C. Sevcik. *Quantitative System Performance: Computer System Analysis Using Queueing Network Models.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1984.

[MCLC11] Ricky K.P. Mok, Edmond W.W. Chan, Xiapu Luo, and Rocky K.C. Chang. Inferring the qoe of http video streaming from user-viewing activities. In *Proceedings of the First ACM SIGCOMM Workshop on Measurements Up the Stack*, W-MUST '11, pages 31–36, New York, NY, USA, 2011. ACM.

[ML97] H. Michiel and K. Laevens. Teletraffic engineering in a broad-band era. *Proceedings of the IEEE*, 85(12):2007–2033, Dec 1997.

[MVSA13] Ahmed Mansy, Bill Ver Steeg, and Mostafa Ammar. Sabre: A client based technique for mitigating the buffer bloat effect of adaptive video flows. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys '13, pages 214–225, New York, NY, USA, 2013. ACM.

[Nag84] John Nagle. Congestion control in ip/tcp internetworks. *SIGCOMM Comput. Commun. Rev.*, 14(4):11–17, October 1984.

[NJ12] Kathleen Nichols and Van Jacobson. Controlling queue delay. *Queue*, 10(5):20:20–20:34, May 2012.

[PA14] Carlos Pereira and Ana Aguiar. Towards efficient mobile m2m communications: Survey and open challenges. *Sensors*, 14(10):19582–19608, 2014.

[PFAB14] Christoph Paasch, Simone Ferlin, Ozgu Alay, and Olivier Bonaventure. Experimental evaluation of multipath tcp schedulers. In *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, CSWS '14, pages 27–32, New York, NY, USA, 2014. ACM.

[PK11] Z. Pi and F. Khan. An introduction to millimeter-wave mobile broadband systems. *IEEE Communications Magazine*, 49(6):101–107, June 2011.

[PKTH16] A. K. Paul, H. Kawakami, A. Tachibana, and T. Hasegawa. An aqm based congestion control for enb rlc in 4g/lte network. In *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5, May 2016.

[PNP+13] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg. Pie: A lightweight control scheme to address the bufferbloat problem. In *High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference on*, pages 148–155, July 2013.

[Pos03] Jon Postel. Rfc 793: Transmission control protocol, september 1981. *Status: Standard*, 88, 2003.

[PS01] J. M. Pitts and J. A. Schormans. *Introduction to IP and ATM Design and Performance: With Applications Analysis Software*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 2001.

[PSL+16] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang. Recent advances in cloud radio access networks: System architectures, key techniques, and open issues. *IEEE Communications Surveys Tutorials*, 18(3):2282–2308, thirdquarter 2016.

[PZR02] Erich Plasser, Thomas Ziegler, and Peter Reichl. On the non-linearity of the red drop function. In *Proc. of Inter. Conference on Computer Communication (ICCC)*. Citeseer, 2002.

[RAT13] D. M. Raghuvanshi, B. Annappa, and M. P. Tahiliani. On the effectiveness of codel for active queue management. In *2013 Third International Conference on Advanced Computing and Communication Technologies (ACCT)*, pages 107–114, April 2013.

[RSC+10] J Rosenberg, H Schulzrinne, G Camarillo, A Johnston, J Peterson, R Sparks, M Handley, and E Schooler. Rfc 3261-sip: Session initiation protocol, 2010.

[SCHW01] Hung-Kai Shiu, Yi-Hao Chang, Ting-Chao Hou, and Cheng-Shong Wu. Performance analysis of tcp over wireless link with dedicated buffers and link level error control. In *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No.01CH37240)*, volume 10, pages 3211–3216 vol.10, 2001.

[SCW+16] M. Seufert, P. Casas, F. Wamser, N. Wehner, R. Schatz, and P. Tran-Gia. Application-layer monitoring of qoe parameters for mobile youtube video streaming in the field. In *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pages 411–416, July 2016.

[SHIK12] Sea Shalunov, Greg Hazel, Janardhan Iyengar, and Mirja Kuehlewind. Low extra delay background transport (ledbat). Technical report, 2012.

[SJS14] A. Showail, K. Jamshaid, and B. Shihada. An empirical evaluation of bufferbloat in ieee 802.11n wireless networks. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 3088–3093, April 2014.

[SJS16] A. Showail, K. Jamshaid, and B. Shihada. Buffer sizing in wireless networks: challenges, solutions, and opportunities. *IEEE Communications Magazine*, 54(4):130–137, April 2016.

[SK] M Scharf and S Kiesel. Head-of-line blocking in tcp and sctp: Analysis and measurements. published in global telecommunications conference, globecom'06.

[SLMP03] M. Sagfors, R. Ludwig, M. Meyer, and J. Peisa. Queue management for tcp traffic over 3g links. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, volume 3, pages 1663–1668 vol.3, March 2003.

[SRL98] Henning Schulzrinne, A Rao, and R Lanphier. Rfc 2326: real time streaming protocol. *Available on http://www. ietf. org/rfc/rfc2326. txt*, 1998.

[Sto11] Thomas Stockhammer. Dynamic adaptive streaming over http –: Standards and design principles. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 133–144, New York, NY, USA, 2011. ACM.

[SV95] M. Shreedhar and George Varghese. Efficient fair queueing using deficit round robin. *SIGCOMM Comput. Commun. Rev.*, 25(4):231–242, October 1995.

[SV15] P. Szil gyi and C. Vulk n. Efficient lte pdcp buffer management. In *2015 IEEE International Conference on Communications (ICC)*, pages 5928–5934, June 2015.

[SVM16] F. Schwarzkopf, S. Veith, and M. Menth. Performance analysis of codel and pie for saturated tcp sources. In *2016 28th International Teletraffic Congress (ITC 28)*, volume 01, pages 175–183, Sept 2016.

[SVS14] Janne Seppänen, Martín Varela, and Aggeliki Sgora. An autonomous qoe-driven network management framework. *J. Vis. Comun. Image Represent.*, 25(3):565–577, April 2014.

[Tan09] Yifeng Tan. *Active Queue Management for LTE uplink in eNodeB*. PhD thesis, Helsinki University of Technology, 2009.

[TGWTG11] U. Toseef, C. Goerg, T. Weerawardane, and A. Timm-Giel. Performance comparison of pdcp buffer management schemes in lte system. In *2011 IFIP Wireless Days (WD)*, pages 1–3, Oct 2011.

[TIM+13] ZH Talukder, SS Islam, D Mahjabeen, A Ahmed, S Rafique, and MA Rashid. Cell coverage evaluation for lte and wimax in wireless communication system. *World Applied Sciences Journal*, 22(10):1486–1491, 2013.

[TSH+18] Rohit P. Tahiliani, Sagar Sachdeva, Sachin Hadke, Shane Sheehan, and Eamonn O' Nuallain. Evaluation of random early detection and adaptive random early detection in benchmark scenarios. *Journal of Information Sciences and Computing Technologies*, 7(1):667–672, 2018.

[TSZS06] K. Tan, J. Song, Q. Zhang, and M. Sridharan. A compound tcp approach for high-speed and long distance networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–12, April 2006.

[TT17] R. P. Tahiliani and H. Tewari. Implementation of pi2 queuing discipline for classic tcp traffic in ns-3. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–6, June 2017.

[TWTGG11] U. Toseef, T. Weerawardane, A. Timm-Giel, and C. G rg. Lte system performance optimization by discard timer based pdcp buffer management. In *8th International Conference on High-capacity Optical Networks and Emerging Technologies*, pages 116–121, Dec 2011.

[VMS+16] D. J. Vergados, A. Michalas, A. Sgora, D. D. Vergados, and P. Chatzimisios. Fdash: A fuzzy-based mpeg/dash adaptation algorithm. *IEEE Systems Journal*, 10(2):859–868, June 2016.

[VS94] Curtis Villamizar and Cheng Song. High performance tcp in ansnet. *SIGCOMM Comput. Commun. Rev.*, 24(5):45–60, October 1994.

[Whi15] G. White. Active queue management in docsis 3.1 networks. *IEEE Communications Magazine*, 53(3):126–132, March 2015.

[WWZH11] J. Wang, J. Wen, J. Zhang, and Y. Han. Tcp-fit: An improved tcp congestion control algorithm and its performance. In *INFOCOM, 2011 Proceedings IEEE*, pages 2894–2902, April 2011.

[YLLL18] Jiancheng Ye, Ka-Cheong Leung, Victor OK Li, and Steven H Low. Combating bufferbloat in multi-bottleneck networks: Equilibrium, stability, and algorithms. Technical report, Technical Report TR-2018-001, Dept. of EEE, HKU, 2018.

[ZDZ+17] F. Zhou, T. Dreibholz, X. Zhou, F. Fu, Y. Tan, and Q. Gan. The performance impact of buffer sizes for multi-path tcp in internet setups. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 9–16, March 2017.

[ZMS+13] H. Zhao, R. Mayzus, S. Sun, M. Samimi, J. K. Schulz, Y. Azar, K. Wang, G. N. Wong, F. Gutierrez, and T. S. Rappaport. 28 ghz millimeter wave cellular communication measurements for reflection and penetration loss in and around buildings in new york city. In *2013 IEEE International Conference on Communications (ICC)*, pages 5163–5167, June 2013.

[ZMZ+16] Menglei Zhang, Marco Mezzavilla, Jing Zhu, Sundeep Rangan, and Shivendra S. Panwar. The bufferbloat problem over intermittent multi-gbps mmwave links. *CoRR*, abs/1611.02117, 2016.

[ZMZ+17] M. Zhang, M. Mezzavilla, J. Zhu, S. Rangan, and S. Panwar. Tcp dynamics over mmwave links. In *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–6, July 2017.

[ZNSH12] C. C. Zarakovitis, Q. Ni, D. E. Skordoulis, and M. G. Hadjinicolaou. Power-efficient cross-layer design for ofdma systems with heterogeneous qos, imperfect csi, and outage considerations. *IEEE Transactions on Vehicular Technology*, 61(2):781–798, Feb 2012.