DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2020

L4S in 5G networks

DAVIDE BRUNELLO

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

L4S in 5G networks

DAVIDE BRUNELLO

Master in Computer Science
Date: July 11, 2020
Supervisor: Mustafa Özger
Industrial Supervisor:Ingemar Johansson S
Examiner: Cicek Cavdar
School of Electrical Engineering and Computer Science
Host company: Ericsson AB
Swedish title: L4S i 5G-nätverk

Abstract

Low Latency Low Loss Scalable Throughput (L4S) is a technology which aims to provide high throughput and low latency for the IP traffic, lowering also the probability of packet loss. To reach this goal, it relies on Explicit Congestion Notification (ECN), a mechanism to signal congestion in the network avoiding packets drop. The congestion signals are then managed at sender and receiver side thanks to scalable congestion control algorithms. Initially, in this work the challenges to implement L4S in a 5G network have been analyzed. Using a proprietary state-of-the-art network simulator, L4S have been implemented at the Packed Data Convergence Protocol layer in a 5G network. The 5G network scenario represents a context where the physical layer has a carrier frequency of 600 MHz, a transmission bandwidth of 9 MHz, and the protocol stack follows the New Radio (NR) specifications. L4S has been adopted to support Augmented Reality (AR) video gaming traffic, using the IETF experimental standard Self-Clocked Rate Adaptation for Multimedia (SCReAM) for congestion control. The results showed that when supported by L4S, the video gaming traffic experiences lower delay than without L4S support. The improvement on latency comes with an intrinsic trade-off between throughput and latency. In all the cases analyzed, L4S yields to average application layer throughput above the minimum requirements of high-rate latency-critical application, even at high system load. Furthermore, the packet loss rate has been significantly reduced thanks to the introduction of L4S, and if used in combination with a Delay Based Scheduler (DBS), a packet loss rate very close to zero has been reached.

Keywords

Low Latency Low Loss Scalable Throughput, Self-Clocked Rate Adaptation for Multimedia, 5G, Latency-critical application

Sammanfattning

Low Latency Low Loss Scalable Throughput (L4S) är en teknik som syftar till att ge hög bittakt och låg fördröjning för IP-trafik, vilket också minskar sannolikheten för paketförluster. För att nå detta mål förlitar det sig på Explicit Congestion Notification (ECN), en mekanism för att signalera "congestion", det vill säga köuppbyggnad i nätverket för att undvika att paketet kastas. Congestionsignalerna hanteras sedan vid avsändare och mottagarsida där skalbar anpassning justerar bittakten efter rådande omständigheter. I detta arbete har utmaningarna att implementera L4S i ett 5G-nätverk analyserats. Sedan har L4S implementerats på PDCP lagret i ett 5G-nätverkssammanhang genom att använda en proprietär nätverkssimulator. För att utvärdera fördelarna med implementeringen har L4S-funktionerna använts för att stödja Augmented Reality (AR) videospelstrafik, med IETF-experimentella standard Self-Clocked Rate Adaptation for Multimedia (SCReAM) för bitrate-kontroll. Resultaten visade att med stöd av L4S upplever videospelstrafiken lägre latens än utan stöd av L4S. Förbättringen av latens kommer med nackdelen av en minskning av bittakt som dikteras av den inneboende avvägningen mellan bittakt och latens. I vilket fall som helst är kapacitetsminskningen med L4S rimlig, eftersom goda kapacitetsprestanda har uppnåtts även vid hög systembelastning. Vidare har paketförlustfrekvensen reducerats avsevärt tack vare införandet av L4S, och om den används i kombination med en Delay baserad schemaläggare (DBS) har en paketförluster mycket nära noll uppnåtts.

Sommario

L4S (Low Latency, Low Loss, Scalable Throughput) è una tecnologia che mira a fornire allo stesso tempo throughput elevato e bassa latenza per il traffico IP, riducendo inoltre la probabilità di perdita di pacchetti. Per raggiungere questo obiettivo, essa si basa sull'Explicit Congestion Notification (ECN), un meccanismo per segnalare la congestione nella rete evitando la perdita dei pacchetti. I segnali di congestione vengono quindi gestiti dal mittente e dal destinatario grazie agli algoritmi di controllo della congestione scalabili. Inizialmente, in questo lavoro sono state analizzate i problemi implementativi di L4S in una rete 5G. Utilizzando un simulatore di rete allo stato dell'arte, L4S è stato implementato a livello di protocollo Packed Data Convergence Protocol in una rete 5G. Lo scenario di rete 5G rappresenta un contesto in cui lo strato fisico ha una frequenza portante di 600 MHz, una larghezza di banda di trasmissione di 9 MHz e lo stack di protocollo segue le specifiche New Radio (NR). L4S è stato quindi adottato per supportare il traffico di videogiochi in realtà aumentata (AR), utilizzando lo standard sperimentale IETF Self-Clocked Rate Adaptation for Multimedia (SCReAM) per il controllo della congestione. I risultati hanno evidenziato che quando supportato da L4S, il traffico di videogiochi ha sempre avuto una latenza inferiore rispetto al caso senza L4S. Il miglioramento della latenza tuttavia comporta un compromesso intrinseco tra throughput e latenza. In ogni caso, utilizzando L4S sono state ottenute buone prestazioni di throughput anche a carico di sistema elevato in tutti i casi analizzati. Inoltre, il tasso di perdita di pacchetti è stato significativamente ridotto grazie all'introduzione di L4S e, se utilizzato in combinazione con un Delay Based Scheduler (DBS), è stato raggiunto un tasso di perdita di pacchetti molto vicino allo zero.

Acknowledgments

I would like to thank Ericsson Research for the opportunity to work on a very interesting and relevant thesis project. In particular I am very grateful to Mats Nordberg and Ingemar Johansson S, that helped and guided me with important feedback, suggestions and also supported me not only from a working point of view.

I'm very grateful also to Mustafa Özger, and Cicek Cavdar, my supervisor and examiner at KTH, that provided relevant feedback also for the writing of the report.

I would also like to thank EIT Digital that gave me the opportunity to study in two different countries, to discover new cultures, to meet new students and grow as a person.

I thank all my friends that have been close to me during these years, also through video-calls in the corona times.

Finally, I want to thank my family and Giulia, certainties in my life, which although distant, have always managed to support me along this journey.

Stockholm, July 11, 2020

Contents

Acronyms

5G-EIR 5G-Equipment Identity Register.

5GCN 5G Core Network.

AF Application Function.

AMF Access and Mobility Management Function.

AR Augmented Reality.

AUSF Authentication Server Function.

CE Congestion Experienced.

CN Core Network.

CP Control Plane.

CU Central Unit.

DCTCP Data Centre TCP.

DN Data Network.

DRB Data Radio Bearer.

DU Distributed Unit.

e2e end-to-end.

ECN Explicit Congestion Notification.

eMMB enhanced Mobile Broadband.

EPC Evolved Packet Core.

F1-U F1 interface for User plane data.

FC Flow Control.

gNB next-generation NodeB.

GTP GPRS Tunneling Protocol.

KPI Key Performance Indicators.

L4S Low Latency Low Loss Scalable Throughput.

LMF Location Management Function.

MAC Medium-Access Control.

MIMO Multiple-Input Multiple-Output.

mMTC massive Machine Type Communications.

N3IWF Non-3GPP InterWorking Function.

NEF Network Exposure Function.

NIC Network Interface Controller.

NR New Radio.

NRF Network Repository Function.

NSSF Network Slice Selection Function.

NWDAF Network Data Analytics Function.

OTT Over-The-Top.

PCF Policy Control Function.

PDCP Packet Data Convergence Protocol.

PDU Protocol Data Unit.

PGW PDN Gateway.

PHY Physical Layer.

PTO Probe Time Out.

QoS Quality of Service.

RACK Recent ACKnowledgment.

RAN Radio Access Network.

RAT Radio Access Technology.

RED Random Early Discard.

RLC Radio-Link Control.

RTCP Real-time Control Protocol.

RTO Retransmission timeout.

RTP Real-time Transport Protocol.

RTT Round Trip Time.

SDAP Service Data Application Protocol.

SDU Service Data Unit.

SEPP Security Edge Protection Proxy.

SMF Session Management Function.

SMSF Short Message Service Function.

TCP Transmission Control Protocol.

ToS Type of Service.

TTI Transmission Time Interval.

UDM Unified Data Management.

UDR Unified Data Repository.

UDSF Unstructured Data Storage Function.

UE User Equipment.

UP User Plane.

UPF User Plane Functions.

URLLC Ultra Reliable Low Latency Communications.

VR Virtual Reality.

Chapter 1

Introduction

Nowadays, interactive high-rate latency-critical applications as online gaming, Virtual Reality (VR), Augmented Reality (AR) and remote control are becoming more and more widespread and used. These applications require low latency and high throughput to bring better usage experience. Although achieving high throughput is possible, it is challenging to guarantee high throughput and low latency at the same time. This because latency can be adversely affected by events such as congestion and packet loss. These events occur when it is injected into the network more traffic than the current capacity of the network itself. To avoid that these unwanted events happen, congestion control algorithms have been developed. The latter have the purpose of preventing and limiting congestion by reducing the transmission rate when necessary. Classic congestion control mechanisms, like TCP New Reno [1], TCP Cubic [2] and the TCP Reno friendly[1] congestion control use the packet loss as the only signal of congestion. Then, when retransmission occurs, the congestion window is reduced (by 50% for Reno, by 30% for Cubic), significantly decreasing the throughput.

To reduce the number of retransmissions and to reach higher throughput, the network nodes implement relatively large buffers, where packets are queued instead of being dropped. However, the use of large buffers introduces another problem, known as queue delay. It is an important component of the total delay of a flow, especially when a network node begins to be congested. Queue delay occurs when in a network node the input rate of packets is higher than the output rate, and therefore the incoming packets are "queued" in the buffer waiting to be processed and sent.

[1] Congestion control algorithms are TCP Reno friendly if they can coexist with TCP Reno without affecting negatively the Reno traffic rate.

To prevent and reduce the effects of congestion, Active Queue Management (AQM) [3] techniques have been introduced and Explicit Congestion Notification (ECN) [4] has been standardized. The AQM techniques consist of queue management mechanisms to prevent the node from becoming completely congested and therefore discarding all new incoming packets. ECN was proposed in 2001, and relies in the use of 2 bits in the IP header to report a congestion experienced. AQM and ECN can be used together in a network node to improve performance. For example, when the queue starts growing in the network node, packets in the queue are Congestion Experienced (CE) marked instead of being discarded. In this way the packet loss, and thus the retransmission of the packet are avoided. ECN and AQM are discussed in more detail in Sections 2.1 and 2.2.

Although ECN and AQM are useful to avoid congestion and retransmission, they are not enough to provide low latency, low packet loss and good throughput at the same time. For this reason, Low Latency Low Loss Scalable Throughput (L4S) [5] was proposed in 2015. L4S is a technology intended to ensure low delay and low packet loss rate for Internet flows, without affecting throughput performance.

5G is the new cellular communication standard. We can distinguish 3 main different 5G use-cases: mobile broadband (eMBB), massive machine-type communications (MTC) and Ultra-Reliable Low Latency Communication (URLLC). Since L4S is a general technology that works with IP traffic, all the 5G use-cases can benefit from L4S assistance to reach lower latency.

This thesis focuses on high-rate latency-critical applications such as real-time cloud based AR online gaming in a 5G network with L4S support. The goal is to evaluate the impact of L4S applied to a 5G network context, and the Key Performance Indicators (KPIs) considered are throughput, end-to-end latency[2], network queue delay[3] and packet loss rate.

1.1 Problem

Ensuring low latency is not easy, especially in a loaded system. The factors that make difficult to keep low latency are congestion, retransmission and queue delay. Congestion controls, AQM techniques, ECN have been implemented during the years to mitigate all the factors that can affect latency but there is a trade-off that must be taken into consideration. Keeping the latency and the

[2]The definition of the latency metrics is provided in Section 3.3.

[3]Network queue delay means the sum of the time spent in the queue of each network node.

queue delay very low means limiting the throughput performance. The purpose of L4S is to mitigate this trade-off by minimizing the queue delay component, to provide lower and more stable end-to-end delay without significantly compromising throughput performance. The strategy to achieve that results is to signal a congestion by changing the ECN bits in the IP header (marking) and more details about L4S mechanism are given in Section 2.4.

L4S is a technology designed to be implemented in different types of networks, both wired and wireless. Since L4S is not designed for a specific access technology, the L4S implementation in a specific context can involve some challenges. In this section, the L4S implementation challenges in the 5G network are presented.

The 5G network infrastructure is designed so that the bottleneck is usually the radio access part. The radio channel may be subject to variations in quality that can be compared to congestion. To avoid packet losses in case of channel condition variations, queues are implemented in the Radio Access Network (RAN), where packets are stored waiting to be transmitted. The queue can be implemented in two places: either at the Packet Data Convergence Protocol (PDCP) layer or the Radio-Link Control (RLC) layer. The use of a queue, however, causes an increase in the queue delay component, and therefore the increase in the end-to-end latency.

Usually the queue is located at the RLC layer, and this implies an implementation challenge for L4S marking. At RLC layer packets are encrypted and therefore it is impossible to inspect, manipulate and mark packets to signal a congestion. Another implementation challenge for the deployment of L4S in a 5G network is the propagation of L4S congestion, since it must be guaranteed also when packets are tunneled, e.g. by the GPRS Tunneling Protocol (GTP) .

Regarding coexistence of different traffic types on the same network, only traffic that supports L4S should be L4S marked. This means that a mechanism to filter the L4S capable traffic should be present in the network nodes. In addition, a related problem is to understand whether it is necessary to use different priorities for L4S capable and not-L4S capable[4] traffic.

1.2 Purpose

The purpose of this work is to investigate whether implementing L4S features in a 5G network context can bring benefits to high-rate latency-critical appli-

[4]With L4S capable traffic it is intended the traffic that supports L4S, that is also indicated by the ECN codepoint ECT(1). The traffic that does not support L4S, indicated both through the Not-ECT or ECT(0) codepoints, is also denoted as not-L4S capable traffic.

cations such as cloud based AR video gaming.

1.3 Goals

The main goal of this thesis work is to analyze and evaluate L4S when supporting video gaming traffic in the 5G access network. To achieve the main goal, four sub-goals need to be fulfilled.

- The first sub-goal is to comprehend where there are possible obstacles for the L4S deployment in the 5G network.
- The second goal regards having a working L4S implementation in the 5G network.
- The third goal concerns gaining knowledge about how varying the L4S implementation parameters can impact the traffic KPIs.
- The last sub-goal consists in the understanding of the results, to analyze and assess the KPIs when the traffic is supported by L4S.

The description of the methodology needed to reach the goals is provided in Section 1.5.

1.4 Research Question

In order to solve the problems listed in Section 1.1 and to reach the thesis goals, the research questions that guided this thesis work are here reported:

- The first question concerns tunnels used in a cellular network such as GPRS Tunnelling Protocol for user plane (GTP-U) and IPSec tunnels. In particular, the question is: can they obstruct the propagation of the L4S congestion signals?
- As for the queue at RLC layer, the question is the following: is it possible to move the queue to the PDCP layer, so that congestion can be easily reported?
- If the queue can not be moved to the PDCP layer, what other mechanisms can be implemented to report congestion?
- Another question affects how to handle classic and L4S traffic, is it necessary to use different schedulers with different priorities?

- Furthermore, given that L4S marks packets early when the queue starts to grow (based on a threshold), what is the impact of changing the marking thresholds?

Finally, the question that describes the most general problem, and that this thesis aims to answer is: **Can L4S improve the KPIs of a high-rate latency-critical application traffic in the 5G context?**

1.5 Research Methodology

The research methodology and the approach to reach the goals of this thesis work are here presented. The first task consists in a literature review regarding the standards and specifications with the aim of acquiring knowledge about the L4S and 5G concepts. Through this task it is possible to answer the first three research questions, reaching the first sub-goal of the work. The second step regards the implementation of the L4S features. The Ericsson proprietary state-of-the-art network simulator is used to emulate a 5G network context, then the simulator is modified through the addition of the L4S marking strategy. The third goal is achieved by carrying out several simulations with a different L4S parameters configuration. The results analysis provides the understanding to answer the second-last research question. In the last phase, simulation with and without L4S support are performed. Mill's method of difference is used in this phase, since all the background variables and the scenario are maintained constant, except for the L4S support that is the feature of interest. Finally, through a results comparison, the main goal is achieved and the last research question is answered.

1.6 Novelty and contribution of the thesis

L4S is a general technology designed to fit different network architectures. Thus, implementing L4S in a specific network context (as a cellular networks) can involve challenges. At the time of this thesis is written, no work that discuss L4S implementation in a 5G (and cellular) networks is publicly available.

The novelty of this thesis work consist of the implementation and assessment of L4S technology in a 5G network scenario. The contribution regards the understanding about the L4S implementation challenges in a cellular context. The contribution includes also the proposed L4S marking strategy at the gNB, the understanding of how the implementation parameters changes

the KPIs and lastly the assessment of L4S when supporting high-rate latency-critical application in a 5G network context.

1.7 Delimitations

This thesis focuses on the implementation of L4S in the stand-alone 5G network configuration (a.k.a. Option 2 [6]), with a unified RAN[5]. Considering the limited time period, to speed up the simulations it has been chosen to simplify some aspects of the 5G architecture as explained in Section 4.3. Furthermore, since L4S aims to improve latency performance on for the end-user traffic, only the user plane traffic is considered in this work. Moreover, the focus of the work is in the dowlink case, and thus the implementation of L4S in the uplink case is left for future works.

1.8 Structure of the thesis

This thesis is structured as follow. In Chapter 2 the background is provided to better understand the work. Chapter 3 explains the research methods and the methodology used. Chapter 4 discuss the implementation strategy and the scenario created. In chapter 5 the results of the work are reported and then discussed in Chapter 6. Finally, Chapter 7 provides the conclusion, reflections about the project and possible future works to be done in the field.

[5]Unified RAN means that all protocols and processing regarding the radio access part of the network are executed by the same physical hardware.

Chapter 2

Background

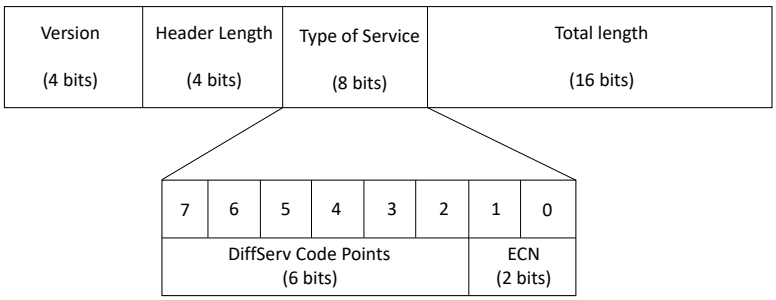
An important though not constant component of end-to-end latency is the queue delay. Although reducing the queue delay implies lower end-to-end latency, it also considerably reduces the throughput performances of the TCP Reno friendly traffic. The reason for this behavior is the intrinsic throughput-delay trade-off. L4S, explained in section 2.4, is based on the idea of decreasing end-to-end latency by reducing queue delay, without significantly impact the throughput performance.

This Chapter explains concepts necessary for the understanding of the entire project, such as ECN in Section 2.1, AQM in Section 2.2, and scalable congestion control mechanisms in Section 2.3. Finally, the architecture of the 5G network is introduced and explained in Section 2.5.

2.1 Explicit Congestion Notification

Explicit Congestion Notification (ECN) is a mechanism for the management and notification of congestion in the network avoiding the drop of packets. This mechanism is intended to reduce end-to-end latency by giving a faster and explicit signal of congestion. ECN was standardized in [4] and updated by [7].

The operating principle is simple and based on the use of two bits in the IP header to notify the presence of congestion. In particular, the bits used by ECN are the least significant bits of the Type of Service (ToS) octet of the IPv4 header, and the least significant bits (10 and 11) of the Traffic Class field in the IPv6 header. Fig. 2.1 shows the structure, highlighting the position of the ECN bits in the IPv4 and IPv6 headers.



(a) IPv4 header, first 4 bytes focusing on ToS.

(b) IPv6 header, first 4 bytes.

Figure 2.1: ECN bits in the IP headers.

Through these two bits, it is possible to map 4 different codepoints, respectively reported and explained in Table 2.1. The acronym ECT stands for ECN Capable Transport, and the ECT codepoint is used to inform the network nodes that the packet supports the ECN marking. It can be noted that codepoints ECT(0) and ETC(1) have the same meaning.

Binary Codepoint	**Codepoint Name**	**Meaning**
00	Not-ECT	Not ECN-capable transport
01	ECT(1)	ECN-capable transport
10	ECT(0)	ECN-capable transport
11	CE	Congestion Experienced

Table 2.1: ECN codepoints and meaning.

The CE codepoint means that a congestion has been detected, and a CE marked packet must be considered at the endpoints (sender and receiver) as a dropped packet. The primary requirement for ECN operation is that the network nodes are ECN capable, that is, they can inspect the ECN codepoint and change it by setting the CE codepoint in case of congestion.

The three most important benefits that can be achieved by using ECN marking are throughput improvement, reduced head-of-line blocking and also a reduced probability of Retransmission Timeout (RTO) expiry. Throughput,

head-of-line blocking and RTO are affected from packet drop, and since ECN reduces the packet drop probability, all the three aspects are improved. A detailed explanation for these benefits is provided in [8].

2.2 Active Queue Management

AQM is a policy for managing buffers in various nodes of a network, such as routers and switches. It consists in the drop (or mark) of the packets in the queue before the queue is approaching the node's memory limit. The purpose of the AQM is to reduce network congestion and consequently to increase performance by signaling the congestion to congestion-controlled transport protocol (e.g. TCP).

Nodes that do not implement AQM, buffer packets as long as they have available memory. This implies that at time of congestion, when the buffer becomes full, a new incoming packet has to be discarded. This is also commonly known as tail drop. The packet drop involves a considerable increase in latency due to the retransmission, further aggravated by the fact that the network is congested.

To address this problem, AQM includes a series of techniques and algorithms such as [9], [10] and [11], trying to avoid (or limit) packet losses by notifying in advance a possible future state of congestion. This notification can be done in two possible ways: *Dropping* or *Marking* packets. *Dropping* means that the packet is discarded by the node. Then, the receiver notifies the sender that a packet has been lost, and in this way the sender understands that a congestion happened.

Marking packets exploits the two ECN bits in the IP header that can be changed to signal congestion. At receiver side, a marked packed has the same meaning of a missing packet, it indicates a congestion in the network. By using ECN, the receiver can notify the sender immediately about congestion, without waiting for the sender to understand on his own that congestion happened, through mechanism like RTO, Probe Time Out (PTO) and Recent ACKnowledgment (RACK).

It is clear that if possible, it would be preferable to mark packets instead of packets drop, but not all the traffics support the marking features.

An example of AQM based on Random Early Discard (RED) involves the setting of a *min threshold* and a *max threshold*. The value of the threshold is not predefined, it is adjustable and can vary in different nodes. When a new packet reach the node that is using RED, it is always queued if the queue length is below the *min threshold*. If the queue length is in between the two

thresholds then the packet is dropped accordingly to a probability that depends on the queue length. In the case the queue length is above the *max threshold*, the packet is always dropped. Through this process, dropping packets before filling the buffer, the sender is notified to reduce the sending rate in time to avoid a congestion.

Going more into detail, among the AQM algorithms, there is a distinction between *queue management* and *scheduling* algorithms. The queue management algorithms control the queue size by deciding whether to mark or discard packets. On the other hand, scheduling algorithms determine the order in which the packets are sent out of the node, intending to distribute the available bandwidth between the flows. Although confusion may arise between these types of algorithms, it should be remarked that they are designed to solve different problems, and are normally used in combination.

In addition, AQM methods can exploit ECN [4] in order to avoid packets drop. As reported in [12], it is recommended to implement AQM technique in each node of the network, and AQM algorithms should support ECN.

2.3 Scalable Congestion Control

Since it was developed, it has been known that TCP New Reno would not be able to scale in high bandwidth-delay products, because after a startup phase (called slow start) it enters in the Additive Increase Multiplicative Decrease mode (AIMD). In the AIMD phase, when a congestion happen, the sending rate is reduced by the 50%, but since every RTT the sending rate is increased additively, it takes long time to recover after a congestion event. This implies that when the channel bandwidth increase, TCP New Reno takes longer time to recover.

To overcome the problem, several variants of congestion control have been proposed and implemented over the years such as *Cubic*, that reduces the congestion window by 30% (instead of 50%) and have a function to recover faster after a congestion. However, also Cubic is approaching its scalability limit. An example reported in [5] illustrates that in a scenario with a flow rate of 800 Mb/s and 36 ms RTT, TCP Cubic takes 340 round trips to recover, that corresponds to 12 seconds (on the same scenario, New Reno would take almost a minute to recover).

To solve the scalability problem, new solutions with scalable congestion control have been developed based on ECN as DCTCP [13], TCP Prague [14], BBRv2 [15] and SCReAM [16]. The scalable congestion control algorithms adapt the sending rate proportionally to the number of congestion signal re-

ceived. In this way, if no congestion signal are received, the rate is increased. On the contrary if congestion happens, the rate is decreased, but not decreased as much as it happen in Reno or Cubic: it is decreased proportionally to the fraction of the CE marked packets. Scalable congestion controls introduce on average 2 control signals per RTT [17] regardless of the flow rate, and in this way, it is possible to scale with high throughput while keeping accurate congestion control dynamic.

Thus far, the implementation of protocols with a scalable congestion control has been limited to the closed systems as data centers. Scalable congestion controls require a change in the sender, receiver and network nodes [18], and it is possible to do this update easily at once only in a controlled environment.

2.4 L4S

Low Latency Low Loss Scalable Throughput (L4S) [5] is a technology intended to ensure very low delay for the Internet traffic. It is based on the idea of signaling early on a CE when the number of packets queued in a network node starts growing above a defined threshold. In this way, it is possible to keep a low queue delay and, therefore, a lower end-to-end latency maintaining at the same time a good link utilization. At the time this report is written, L4S is still in the standardization process at the Internet Engineering Task Force (IETF).

L4S is based on ECN, and the process of signal congestion and rate adaptation is the following:

- The sender through the use of a specific ECN codepoint (explained in Table 2.2) signals that the packet supports L4S.
- Then, the network nodes recognize the packet as L4S packet, and when a congestion happens it is signaled by changing the ECN bits to indicate CE.
- The packet reaches the receiver, and if the ECN bits show that a congestion happened during the path, the receiver notifies the sender about the congestion.
- The sender, notified about the congestion, reduces the sending rate.

Some conditions are required to allow the process described to happen. The first requirement concerns the endpoints. The receiver must be able to

notice any CE codepoint and to notify the sender about the congestion. Therefore, the latter must modify the transmission rate, thanks to the scalable congestion control, to prevent the network from congesting.

The last requirement touches the various network devices, such as routers, switches, Access Points (AP) and base stations. This refers in particular to those nodes in the network where congestion could occur. To effectively manage congestion, each node in the network should be able to distinguish an L4S packet from a classic packet, reserving different treatment. The classic traffic[1] should be treated normally, while L4S traffic should have a reserved queue where packets are CE marked as soon as queue starts growing. For this reason, two different queues (one for the L4S capable traffic, and one for the not-L4S capable) should be present in the system. The suggested solution is the use of a Dual Queue Coupled AQM explained in [5]. It is a mechanism where the L4S queue and the classic queue are separated, they have different AQM, and there is a scheduler that regulates the output of the network node to give rate fairness between L4S and classic traffic. More details about the implementation are provided in [19].

In L4S implementation, the codepoints have a different meaning from the one presented in Table 2.1, and in Table 2.2 is reported the L4S codepoints meaning.

Binary Codepoint	**Codepoint Name**	**Meaning**
00	Not-ECT	Not ECN-capable transport
01	ECT(1)	L4S-capable transport
10	ECT(0)	Not L4S-capable transport
11	CE	Congestion Experienced

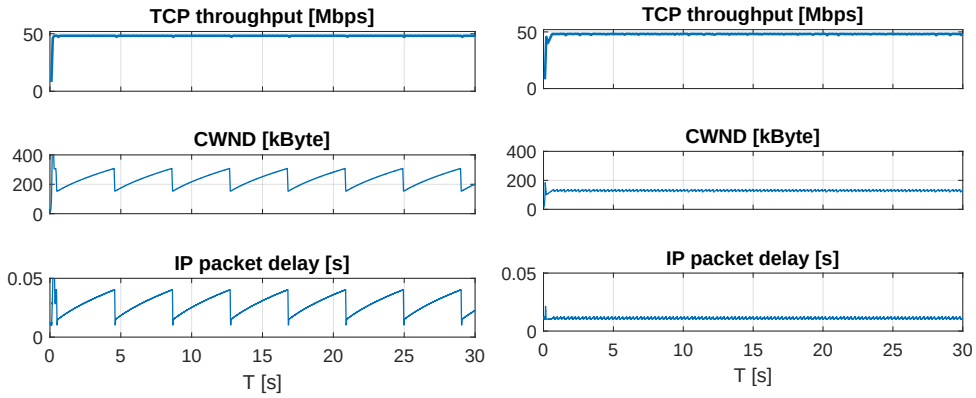
Table 2.2: L4S codepoints and meaning.

As one can see, Table 2.1 and 2.2 have different meaning, in particular for the codepoints ECT(0) and ECT(1). When talking about ECN (Table 2.1), the codepoints ECT(0) and ECT(1) mean that the traffic supports ECN marking, and so network nodes can signal a CE by marking with the CE codepoint. ECT(0) and ECT(1) have the same meaning because at the time ECN was proposed, only three states needed to be signaled: not ECN capable, ECN capable e CE. With the introduction of L4S, another state must be signaled: a code-

[1] With classic it is intended the traffic that is not L4S-capable, also known as not-L4S capable traffic.

point to represent the traffic that is L4S capable. The codepoints then should be interpreted as reported in Table 2.2: ECT(1) means that the traffic should be treated as L4S capable traffic, ECT(0) means that the traffic is ECN capable but not L4S capable (so it has to be sent to the *classic* queue) and the Not-ECT means that the traffic is not ECN capable (and thus not L4S capable). To summarize, network nodes should treat as L4S capable traffic the packet with the ECT(1) and CE codepoints, while ECT(0) and Not ECT-capable should be sent to the classic queue.

L4S requires scalable congestion control algorithms at the endpoint, such as DCTCP, TCP Prague SCReAM or BBRv2, which are able to adapt and respond quickly to congestion signals. Fig. 2.2 shows the comparison between a classic congestion control (TCP New Reno) with the standard ECN (a) and a scalable congestion control (DCTCP) with L4S support (b). The scenario represent a wired network with 50 Mbps bandwidth, 10 ms one way latency (20 ms RTT) and only one client that is downloading one large file via FTP. From Fig. 2.2 (b) one can see how the IP packet delay, that is the delay from when the IP packet leaves the server to the moment he reaches the client, is lowered with respect to (a). The congestion window in (b) shows a more stable behavior than (a) while the transport layer throughput is around 50 Mbps in both cases.



(a) TCP Reno and ECN

(b) DCTCP and L4S

Figure 2.2: Comparison between TCP New Reno with ECN and DCTCP with L4S.

2.5 5G Architecture

5G is the next cellular communication standard. From a technical point of view, the 5G network can be divided into 2 main parts, New Radio (NR) and 5G Core Network (5GCN). NR is the new radio access network technology, accountable to provide network access to the end-user. 5GCN is instead responsible for guarantee services, control the NR and deliver user data to external Data Network (DN).

In the following subsections a deeper overview of 5GCN, NR and their architecture is provided. It should be noted that in this section (and in this thesis in general) the architecture of the 5G network is analyzed, explaining the protocols used focusing on the user-plane. Therefore, specific topics related for example to modulations, configurations of antennas, beam-forming, MIMO and frequencies are omitted, while the services offered by the control-plane are only briefly discussed and not investigated in details.

2.5.1 New Radio

To guarantee the performance suitable to satisfy the three main use cases of the 5G network (eMMB, mMTC, URLLC), the introduction of new technology for radio-access was necessary. This new technology is known as NR. NR can coexist and even collaborate with LTE RAN to provide Dual Connectivity (DC) to the User Equipment (UE). Furthermore, in addition to interacting with 5GCN, NR can be connected to the 4G core network (EPC). This is necessary to operate in non-standalone mode with dual connectivity.

The radio access network, as was the case in 4G, is composed by two nodes. The first one is the UE. Usually it is a smartphone, a tablet, a PC but it can also be a sensor or an IoT device. The second node is the base station, that in NR is called gNB. It has the task of connecting and allowing communication between the UE and the core network.

To effectively guarantee and manage the user plane communication between the gNB and the UE, a protocol stack similar to that adopted in LTE is used, but still with some changes explained in Section 2.5.1.1. Moreover, since the 5GCN implements new functions, new interfaces have been developed for the communication between the NR and the 5GCN.

2.5.1.1 New Radio Protocols and Interfaces

NR implements the protocol stack with some modification if compared to LTE. Fig. 2.3 illustrates how the protocol stack in the gNB in the case of downlink

traffic. An explanation of the tasks of the protocols is also reported.

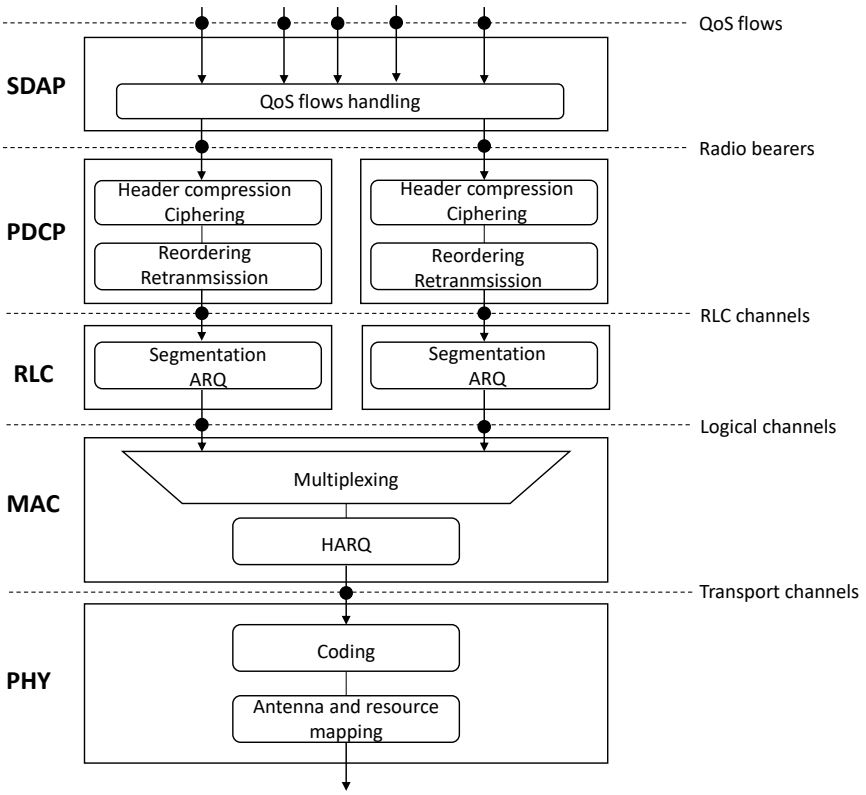


Figure 2.3: User-plane protocol stack [20].

Service Data Application Protocol (SDAP) is the protocol responsible for Quality of Service (QoS) management. SDAP map the flows in the different radio bearers according to the QoS required. This protocol is not present in LTE, and in the scenario where NR is connected to EPC, SDAP is not used.

Packet Data Convergence Protocol (PDCP) performs IP header compression through RObust Header Compression (ROHC) [21], ciphering and integrity protection to defend against eavesdropping and IP spoofing. Moreover, it is responsible for removing duplicated packets, needed in the case of handover, and optionally can provide in-sequence delivery.

Radio-Link Control (RLC) has the task of managing segmentation and retransmission. The service offered to the PDCP layer is called RLC channel. Each RLC channel corresponds to a single RLC entity and a single radio bearer. Unlike the LTE RLC layer, it does not provide in-sequence delivery to higher layers [20].

Medium-Access Control (MAC) multiplexes the logical channels offered

by the RLC layer, manages the Hybrid Automatic Repeat Request (H-ARQ) retransmission . With the goal of faster processing, the MAC header placement has been changed from LTE: instead of having a MAC header at the begin of the MAC PDU, the header information are placed before every MAC SDU. The MAC layer also includes the scheduler functionalities.

The **Physical Layer (PHY)** offers transport channels as a service to the MAC layer. PHY manages all aspects related to physical transmissions, such as the management of antennas, modulations and encoding.

As explained, a protocol layer offer services to the higher protocol layer, and make use of these services offered from the lower layer. In the downlink case, when a packet is received by the upper layer protocol, the current protocol adds a header and after carrying out the necessary processing it forwards it to the lower layer. When talking about data related to the protocol stack, two different nomenclatures can be used: *SDU* (Service Data Unit) which indicates the block of data from/to the upper layer and *PDU* (Protocol Data Unit) which indicates data from/to the lower layer. To explain the process with an example, the PDCP protocol receives an SDAP PDU (a.k.a. PDCP SDU) from the SDAP layer. After processing the data and adding the PDCP header it sends the PDCP PDU (or RLC SDU) to the RLC layer. Fig. 2.4 illustrates the process just described.

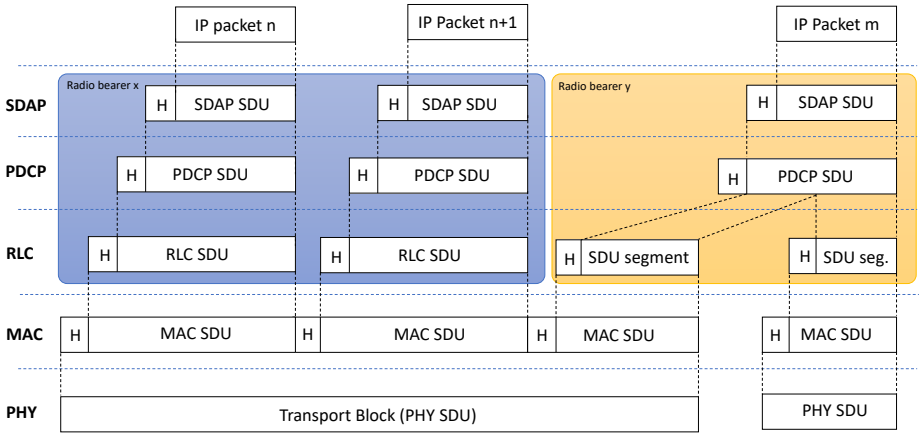


Figure 2.4: User-plane data flow [20].

Fig. 2.4 shows that a MAC PDU can contain RLC PDU of different RCL entities. Moreover, if needed, the RLC layer can segment the PDCP PDU in

many RLC SDUs.

In the 5G network the name of the interface varies from the LTE nomenclature. The interfaces that connects the gNB with the 5GCN are called NG-c (control plane) and NG-u (user plane) [22]. User Plane Function (UPF) and Access and Mobility Management Function (AMF) are the two core network entities that are interfaced with the NR. UPF is responsible for the user plane, while AMF is a control plane function. UPF and AMF are discussed in Section 2.5.2. The interface that connects different gNBs is the Xn interface [23], that is in turn split in Xn-c for control and Xn-u for user plane. The interface between the gNB and the UE is known as Uu. There is also the possibility to split the gNB in two parts, Central Unit (CU) and Distributed Unit (DU). The interface that connects the CU to the DU is called F1 interface [24]. Moreover, it is possible to split the CU user plane from the CU control plane, and the interface that connect CU-u and CU-c is the E1 interface [25]. Fig. 2.5 gives an overview of the interfaces and the connections for the 5G NR.

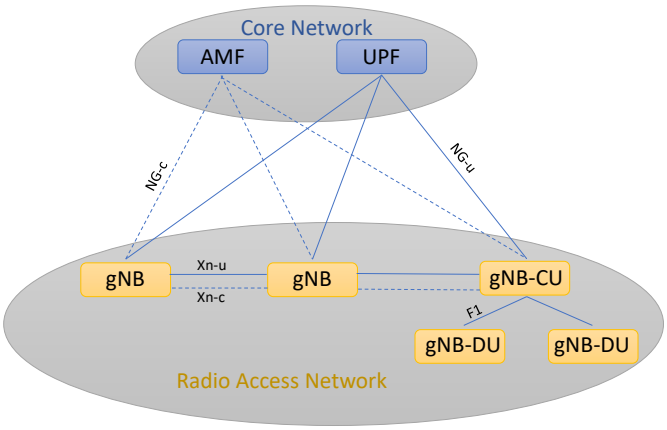


Figure 2.5: NR interfaces [20].

2.5.1.2 Scheduler

The scheduler is one of the most important components of the RAN. It is a part of the MAC layer, but can be viewed as a separate entity. The role of the scheduler is to assign the physical resources dynamically for the transmission both in downlink and uplink. Depending on the transmission scheme (FDD or TDD), the resources consist of *resource blocks* (frequency-domain) or *OFDM symbol* and *slots* if a time-domain transmission is used.

Typically once per slot, the scheduler takes a decision about the transmission, and notifies the selected UEs to be ready to receive the data. This process

is called dynamic scheduling.

In NR, the downlink and uplink scheduling are separated: this means that the uplink and downlink decisions can be taken independently. The scheduling strategy is not specified in the standards, and thus every operator can implement his own strategy. Usually this strategy is created in a way to exploit the physical resources at the best, favoring transmission to devices in good channel conditions and hoping that devices with bad channel conditions will be in a more favorable position in the following slots. However, to avoid that devices in unfavorable conditions are never served, a priority system is used: if a device is not served in a time slot, in the next slot it has a higher priority, until it is served even if the channel quality is not desirable.

In this thesis the role of the scheduler is of significant importance, because changing the scheduler settings, and using different priorities to support L4S traffic, substantial differences in the throughput and latency performance can be obtained. The schedulers used in this work are explained in Section 4.5.

2.5.2 Core Network

The 5G Core Network (5GCN) can be seen as an evolution of the Evolved Packet Core (EPC) improved mainly in 3 aspects: service-based architecture, user plane/control plane division and the support for network slicing [20]. This means that the 5G core network is designed to accommodate the services and features that the network provides. Through the division between the user plane and data plane, it is easier to scale, upgrade and manage the network. Finally, thanks to the network slicing, there is the possibility to provide different services through different logical networks on the same, shared, physical infrastructure. An example of slicing can be the coexistence of a mobile broadband application and a latency-critical application, that run on the same network infrastructure but are logically separated. Fig. 2.6 represents the 5GCN focusing the attention on the functionality provided by the CN.

In Fig. 2.6 the blue blocks are the control plane functions, while the UPF is the only user plane function. It can be seen that the UPF is connected directly to the RAN, to the external DN and it is also connected to the control plane through the Session Management Function (SMF). The RAN and the UE are connected to the control plane only via AMF. The user plane and the control plane functions are explained in the next sections.

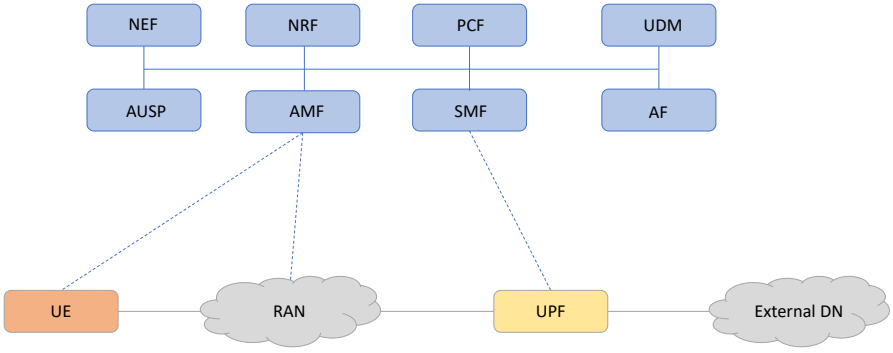


Figure 2.6: 5GCN architecture [20].

2.5.2.1 User Plane Function

The User Plane Function (UPF) is the only 5GCN function related to the user plane. The main task of this function is to connect the NR to the external DN (e.g. Internet). It also deals with the management of QoS, routing and forwarding of user data. The UPF is one endpoint of the GTP tunnel, so it takes care of encapsulating the packets arriving from the outside and send them to the gNB (vice versa it decapsulates the packets that travel the opposite path).

2.5.2.2 Control Plane Functions

Unlike the user plane, the control plane is composed of several functions, and the main ones are reported and explained in this section.

Access and Mobility Management Function (AMF) is one of the most important functions. It allows the UE to register, to be authenticated and to move between different cells.

Session Management Function (SMF) is the function responsible for setting up connectivity between UE and external DNs. It also manages the User Plane connectivity, controls the user session (establishment, modification and release) and allocates the IP for the PDU session. It communicates indirectly to the UE through the AMF.

Network Repository Function (NRF) is a repository that stores the profile

of Network Functions available in the network.

Policy Control Function(PCF) provides policy control for access and mobility features, session management and PDU session selection.

Unified Data Repository (UDR) is a database that contains multiform types of data, such as UE subscription data.

Unified Data Management (UDM) uses the data stored in the UDR to provide, for example, access authorization and registration management.

Authentication Server Function (AUSF) provides service in the subscriber home network: handles authentication and furnishes security parameters for roaming and UE update.

There are further functions provided by the 5GCN, such as Unstructured Data Storage Function (UDSF), 5G-Equipment Identity Register (5G-EIR), Network Slice Selection Function (NSSF), Network Exposure Function (NEF), Network Data Analytics Function (NWDAF), Security Edge Protection Proxy (SEPP), Non-3GPP InterWorking Function (N3IWF), Application Function (AF), Short Message Service Function (SMSF), and Location Management Function (LMF). Since these functions, and in general, the study of the control plane, is not part of the purpose of this thesis, for an in-depth study on the control plane network functions [26] is a suggested reading.

2.5.3 The PDU session

The PDU session is the entity that provides association between the UE and a specific DN in a 5G network. The PDU session can be split in two main parts: the N3 tunnel and the Data Radio Bearer (DRB).

The N3 tunnel refers to the tunnel on the N3 interface (the interface between the UPF and the gNB). In the N3 interface data are GTP-U tunneled, to allow mobility of the user.

The DRB refers to the part of the network between the gNB and the UE, and it is responsible for the radio resources management. One important aspect is that one PDU session can have many DRB with different characteristics. This feature can be exploited also to guarantee different QoS.

The PDU session is established before the real communication begins, and during the establishment the SMF usually allocates[2] the IP address for the UE. In the case that the UE moves from a cell to another, the IP address does not change and so the connection is not broken[3]. In Fig. 2.7 the PDU Session

[2] Also ethernet session can be allocated, but they are out of the scope of this work.

[3] This is true in the majority of the cases, but if the network is not able to handle the handover, the IP can change.

establishment process is explained.

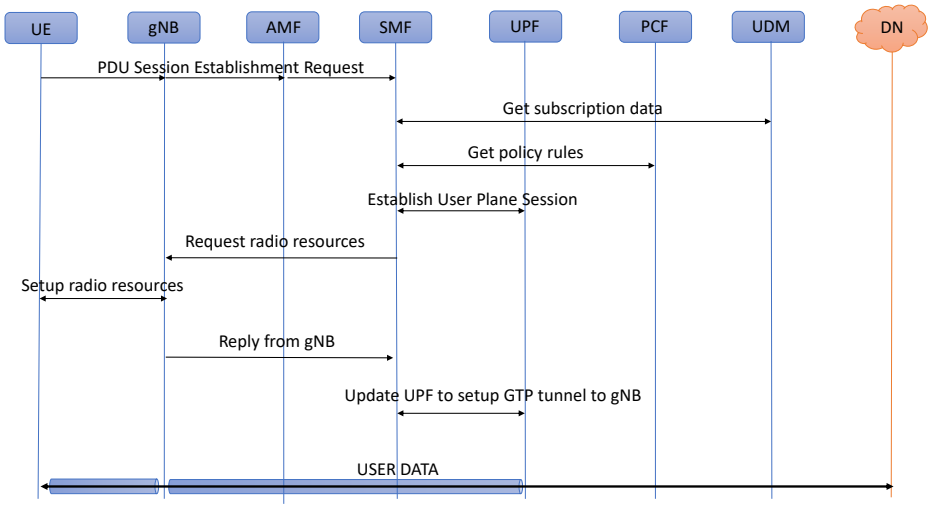
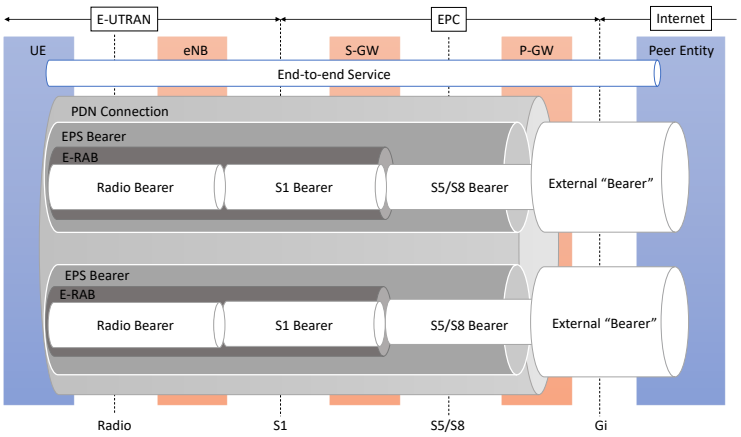


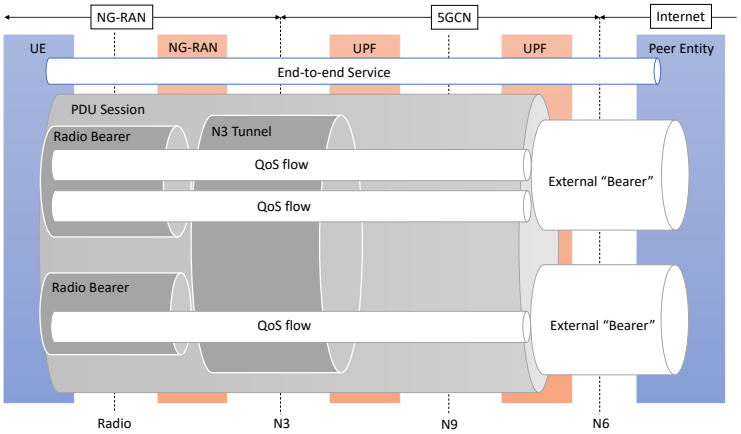
Figure 2.7: PDU Session establishment [26].

The UE can have different PDU Sessions simultaneously, that can be managed even from different UPFs. If this is the case, the UE has n different IP addresses, where n is the number of PDU Sessions established.

The PDU Session and the QoS flow can be seen as the evolution of the PDN Connection and the EPS Bearer in a 4G network. In Fig. 2.8 is reported the high level overview of a PDU Session (b) compared to the 4G PDN Connection (a). In a 5G context, the same radio bearer could carry different QoS flows enhancing the network flexibility. Moreover, it can be seen that in the UPF are condensed the functions of the Packet Gateway (P-GW) and the Serving Gateway (S-GW).



(a) PDN Connection (4G)



(b) PDU Session (5G)

Figure 2.8: PDU Session and PDN Connection comparison [26]

2.5.4 5G Connectivity options

There are different architecture implementation options for the UE connectivity, which are shown in Fig. 2.9. In the figure, the *u* indicates the user plane connection, while the *c* stands for the control plane connection.

Option 1 is not considered a 5G option since it represents the legacy 4G network, that is the baseline and the solution for UE connectivity that is currently implemented. Options from 2 to 7 are the possible alternatives for the cooperation of current technologies (LTE RAN and EPC) with those proposed by the new standards (NR and 5GCN). As happened with the switch from 3G to 4G, it is not possible to change the entire network in a short period, and also the service must be guaranteed for an adequate length of time even for all

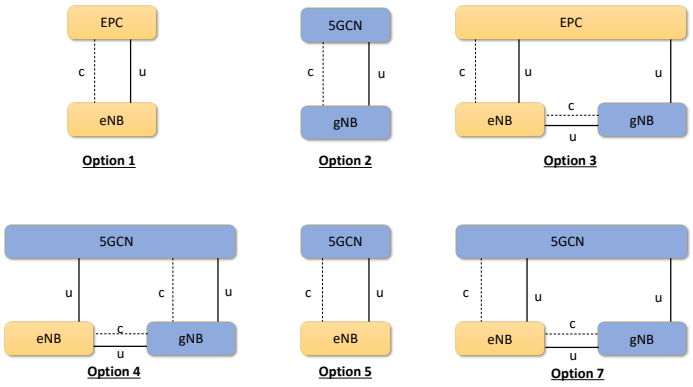


Figure 2.9: UE connectivity options [20]

the devices currently present and installed that work with the 4G network. For this reason, 4G and 5G networks have to work together. As explained in [6], concentrating the research on developing Option 3 with the aim in the future of being able to count on a fully 5G network (Option 2) seems to be the best and worthwhile choice.

2.6 Related Work and Considerations

L4S is a relatively new technology, proposed in 2015 and still in the standardization process. For this reason, the works related to L4S are still few or not publicly available, and yet none of them are related to the implementation of L4S in a cellular network environment.

2.6.1 L4S relevant works

The most relevant project and work related to L4S are the *RITE* project [27] and *Data Center To The Home* [28].

RITE stands for Reducing Internet Transport Latency, and it is a project funded by the European commission under the fp7-ICT programme, with the aim of reducing the latency over the Internet. L4S is one of the results of the RITE project.

In *Data Center to the Home* [28] the authors tested the use of DCTCP with L4S congestion control in a shared network. Through the creation of a Dual Queue Coupled AQM, the result showed a fair bandwidth division between DCTCP traffic and classic traffic. Besides to this, the results exhibited that

DCTCP traffic reaches minimal queue delays and no loss due to congestion in the numerous tests carried out. The authors of *Data Center to the Home* are part of the working group that is responsible for the L4S standardization.

2.6.2 L4S in the industry

Although a player such as Nokia stated that it has implemented L4S in some WiFi products [29], the specifications are internal to the corporation and not open to the public. Also DOCSIS standard has adopted the use of L4S to achieve low latency [30], but also in this case the companies that implement DOCSIS tend to keep the implementation details secret.

2.6.3 State-of-the-art AQM

State-of-the-art AQM such as PIE [11] and FlowQueue-CoDel [31] are designed to solve the problem of congestion and packet loss. However, it should be noted that usually network node buffers are quite large and then the AQM thresholds are too high to provide low queue delay. By lowering the thresholds in the state-of-the-art AQM it is possible to achieve very low queue delay, but classic traffic would not be able to scale and achieve good throughput performance. Furthermore, the algorithms for marking in the state-of-the-art AQM are more complex than the L4S strategy, without bringing additional benefits. These are reasons that justify the choice to investigate L4S instead of other state-of-the-art AQM to reach high throughput and low latency simultaneously.

2.6.4 AQM and ECN support in cellular networks

Currently, AQM techniques are implemented in LTE, but without ECN support. The NR standard instead is designed to support ECN, but not L4S. However, in the current implementation, the AQM marking thresholds are set to values too high to achieve a low queue delay.

There are also other technologies that assign priority to ensure lower latency to certain types of traffic, such as DiffServ [32]. However, the DiffServ mechanism works well as long as the flows that require low latency are in limited numbers. Furthermore, DiffServ traffic types that require low latency are usually characterized by a limited bitrate. For this reason, assigning the Diffserv priority to the interactive application traffic would cause the loss of the benefits that Diffserv offers. Moreover, DiffServ is not a congestion signal and

therefore it does not indicates to reduce the sending rate. This can cause high delay or even packet loss.

Chapter 3

Methods

The scope of this chapter is to describe the research method used in this thesis work. Section 3.1 describes the research process. Section 3.2 illustrates the experimental design. Section 3.3 details the data collection techniques, while Section 3.4 explains the techniques used to evaluate the reliability and validity of the data collected. Finally, Section 3.5 outlines the method used for the data analysis.

3.1 Research Process

The research process of this thesis work has been started with the review of the literature, to understand and become familiar with L4S, ECN and AQM. Also, a literature review about the state-of-the-art has been conducted to understand the 5G network architecture and the protocols involved in the stack.

Once the literature review has been completed, the L4S features has been implemented at PDCP layer in the network simulator developed by Ericsson. The simulator is a Java-based proprietary simulator, which gives the possibility to model every aspect related to a communication network: channel model, user mobility, external DNs (such as the Internet) and even client and server applications that emulate real traffic.

After having implemented the functionality in the simulator, the next step in the research process has been the definition of the simulation scenario, explained in Chapter 4. Then, simulations has been performed several times, and the results averaged, in order to give statistical value and reliability to the data obtained.

3.2 Experimental design

As explained in the previous section, this thesis work has focused on modifying a functionality within a simulator, and the consequent use of this simulator to obtain results. The simulator used is a Java-based proprietary simulator, whose characteristics cannot be revealed in detail since it is Ericsson intellectual property. The simulator implements the 3GPP specification and standards, but it also implements proprietary solutions. Like most network simulators, it provides the possibility of being able to modify many configuration parameters: number of base stations and cells, number of users, their mobility, the intensity with which new users enter the simulated system, the traffic used and many other parameters related both to the radio access and to the Core Network.

The setup scenario is illustrated in detail in Chapter 4, and at high level it consists of a system where n users are streaming AR video gaming traffic in downlink, which means that the video content is sent only by the server to the client. At the same time, $10n$ users are normally surfing the Internet (web traffic). In this thesis work, the performance of interest are those related to the video gaming users, that is the traffic that benefits from L4S support. Web traffic has the aim of stressing the network and mimic a real life traffic scenario. The background traffic and the demanding video gaming traffic content are used to make challenging for the network to keep a low latency. This complex scenario is created with the aim to give higher external validity to the results obtained.

3.3 Metrics and Data collection

When using a simulator, normally it is the simulator itself that provides the results, saving them in a specific format. Even in the case of this thesis work, all the results are provided as output from the simulator, at the end of the simulation. The format in which the files are provided is .mat, a file that can then be analyzed and post-processed through the use of Matlab©.

The simulator used also allows to configure which specific parameters to save in the result file. In this way, one can select exactly the parameters of interest, avoiding saving unnecessary data. In particular, for this work, the interesting parameters saved as a result of the simulations are described below. The parameters here reported comprehend all the KPIs of this work, and are the same parameters analyzed in the results Chapter 5.

The **video frame delay** represents the delay experienced from the moment the video frame is generated by the video encoder to the moment it is received and recreated at the client-side application. This delay can be decomposed in many smaller components.

$$VideoFrameD = RTPQD + ProcD + IPdelay \tag{3.1}$$

In equation 3.1, *VideoFrameD* indicates the video frame delay, *RTPQD* the Real-time Transport Protocol (RTP) [33] queue delay, *ProcD* is the delay due to processing at the receiver side to regenerate the video frame and the *IPdelay* is the layer 3 delay. The **IPdelay** is one important metric considered in this work. It represent the delay of the IP packet, from the moment it is sent until the moment it reaches the final destination and can be decomposed as explained in equation 3.2.

$$IPdelay = NetQD + PropD \tag{3.2}$$

In equation 3.2, *NetQD* represents the network queue delay, and *PropD* is the propagation delay.

The processing delay *ProcD* and the propagation delay *PropD* can be considered as constant delays, and are hard to reduce since they are imposed by physical laws and computational power. The **RTP queue delay** and the **Network queue delay** instead are not constant, and are two of the parameters considered in this work. The **RTP queue delay** is the delay that a video frame experiences in the RTP queue at the sender side. This delay is experienced from the packet before being sent in the network, and the explanation is given in the SCReAM description in Section 4.4. The **Network queue delay** instead is the estimated queue delay of the whole network. It can be seen as the sum of the queue delay on each node in the network and it is computed by the SCReAM congestion control algorithm.

Another interesting parameter is the **Packet loss rate**, that represents the amount of packets lost with respect to the amount of packets sent. The **transmitted rate**, the rate at which the data are transmitted, is computed as application layer (or layer 5) throughput and is one of the KPIs. The statistics related to the physical resources are also considered, for example to understand the channel utilization, denoted as **Resource statistics**. Regarding the performances of the background traffic, two metrics are considered: the **number of finished pages** and the **page delay**. The former represents the total number of web pages downloaded and read from the background users during the simulation time, while the web page delay is the amount of time needed

to download the page. Better explanation about the web traffic model is given in Section 4.4. The last parameter considered is the **Signal-to-Interference-plus-Noise Ratio (SINR)**, computed for each user as the ratio between the signal power received and the cumulative power of the interference and noise in the transmission instant. The SINR is then averaged among the simulation time, to obtain a unique SINR value representative of the whole simulation.

3.4 Assessing the reliability and validity of the data collected

All the data obtained has been provided as a result of the simulations, directly from the simulator used. To be sure of the validity of the results obtained, and to give statistical significance to them, each simulation has been performed several times, and the results of every single simulation have been averaged together to obtain the final results presented in Chapter 5.

The number of times each single simulation has been repeated depends on the number of video-users in the system. For example, with only 2 video-users present in the system, the number of simulations performed is 70, while with 50 video-users in the system, the number of simulations performed is 8. This is because a higher number of users in the system requires fewer simulations to have statistical validity[1]. It is also necessary to reduce the number of iterations with higher number of users, due to processing time. In this work, simulations have had to be rerun because of discovered discrepancies in simulation setup and also bugs in the SCReAM implementation.

3.5 Data Analysis

It has already been mentioned in the previous sections that the simulation results have been saved in a .mat file. For this reason, the post-processing was carried out entirely through the use of Matlab©.

In particular, scripts for the automatic analysis of the features of interest have been created for data analysis. All the post-processing scripts have

[1] With low number of users in the system, the probability of having users in a lucky position (e.g. alone in the cell, low interference) is higher if compared to high number of users in the system. Running the simulation with two users 70 times means having 140 different users results, thus the probability of users in a lucky position is lowered and the results analysis reflects the reality to higher extent.

been designed with re-usability in mind, to obtain a rapid and flexible post-processing.

Through the use of these scripts, the task of data analysis and post-processing has been greatly facilitated, allowing to obtain the results in the same format for each different test carried out.

Chapter 4

Setup and Implementation

In this chapter it is possible to find the details related to implementation challenges in Section 4.1, the L4S implementation choices in Section 4.2 and the simulation scenario in Section 4.3. Section 4.4 gives a description of the traffic models used also with the explanation of SCReAM and the sender side architecture. Finally, Section 4.5 explains the scheduler and the traffic priority used in this work.

4.1 Implementation challenges

4.1.1 Tunnels

The literature review is the first step of this work and answers the first research question. In particular, from [34] it is possible to conclude that IP-in-IP tunnels, such as GTP-U and IPSec are not a problem for the propagation of ECN signals and, therefore, L4S. In other words, tunnels are not a problem if network nodes and tunnel endpoints are compliant with the standards such as [7].

By analyzing what happens when a packet is encapsulated in a tunnel, GTP-U or IPSec, the original packet becomes the payload of another IP packet. The result is a sequence of external and internal IP headers, where the innermost header is the header of the original packet, while the most external one is the header of the last tunnel applied. An example of what a frame can look like is provided in Fig. 4.1.

Although the frame structure may seem complicated, the nodes of the transport network work only on the outermost header. When the propagation of the ECN signals need to be done, the network nodes have to understand if the packet is ECN capable, and set the CE codepoint in case of congestion. It

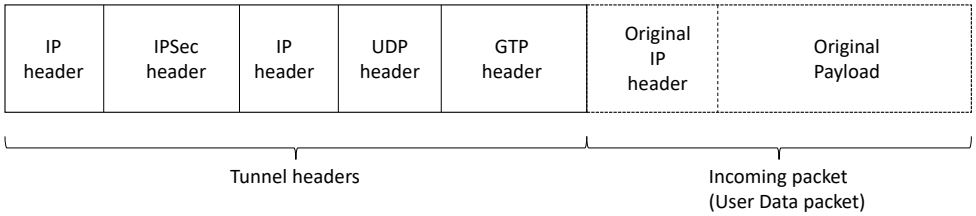


Figure 4.1: Frame structure

is therefore clear that responsibility for the correct propagation of ECN signals must be attributed to the tunnel endpoints, also called encapsulators and decapsulators. Encapsulator is responsible for creating the external header based on the internal header, while the decapsulator removes the external headers. Tables 4.1 and 4.2 show how encapsulator and decapsulator should behave, to propagate ECN signals.

Incoming Header	**Departing Outer Header**	
	Compatibility Mode	Normal Mode
Not-ECT	Not-ECT	Not-ECT
ECT(0)	Not-ECT	ECT(0)
ECT(1)	Not-ECT	ECT(1)
CE	Not-ECT	CE

Table 4.1: Encapsulator Behavior.

Arriving Inner Header	**Arriving Outer Header**			
	not-ECT	ECT(0)	ECT(1)	CE
not-ECT	not-ECT	not-ECT	not-ECT	drop
ECT(0)	ECT(0)	ECT(0)	ECT(1)	CE
ECT(1)	ECT(1)	ECT(1)	ECT(1)	CE
CE	CE	CE	CE	CE

Table 4.2: Decapsulator Behavior.

Tables 4.1 and 4.2 are the same indicated by the standards [7], a recommended reading for those who want to deepen the topic. The encapsulator has 2 different modes of operation, *normal mode* that is the one that should be applied normally, and the *compatibility mode* that should be used if it is known

that the decapsulator is not implemented following the standard requirements, and it is not able to recognize ECN capable traffic.

In conclusion, if the network nodes, encapsulator and decapsulator are implemented according to the standards, tunnels are not an obstacle for the propagation of the ECN and L4S signals.

4.1.2 Marking at RLC layer

The base station is usually the bottleneck node in a cellular network. To accommodate packets that wait to be sent over the wireless link, a queue is implemented at RLC layer. This is a problem for L4S, because the packets in the RLC queue are encrypted (the encryption is performed at PDCP layer) and thus it is not possible to mark the packets to signal congestion. To overcome this problem, two possible solutions can be adopted.

The first solution consist in moving the queue from the RLC layer to the PDCP layer. In this way when packets are queued, they are not ciphered yet, and thus it is possible to change the ECN bit to signal the increasing queue delay. Nevertheless, this solution has some limitations since it involves hardware modification, and thus, it cannot be applied to the already existing base stations. In addition, a mechanism is needed to notify the PDCP layer about the number of packets to send to lower layers, in order to exploit at maximum the physical resources available.

The other possible solution consists in keeping the queue at RLC layer and mark packets at PDCP layer. This solution has the drawback that the marking is not made from the head of the queue, and thus the signal of congestion is not immediate, rather it is delayed depending on the queue delay. This because marking at the PDCP layer, but having a queue at the RLC layer is equivalent to a tail marking, or in other words, marking the packets that enter the queue instead of marking packets that leave the queue. Also in this case a new communication mechanism must be implemented, to communicate at the PDCP layer the level of the RLC queue. An explanation of the two possible solutions is provided in Fig. 4.2.

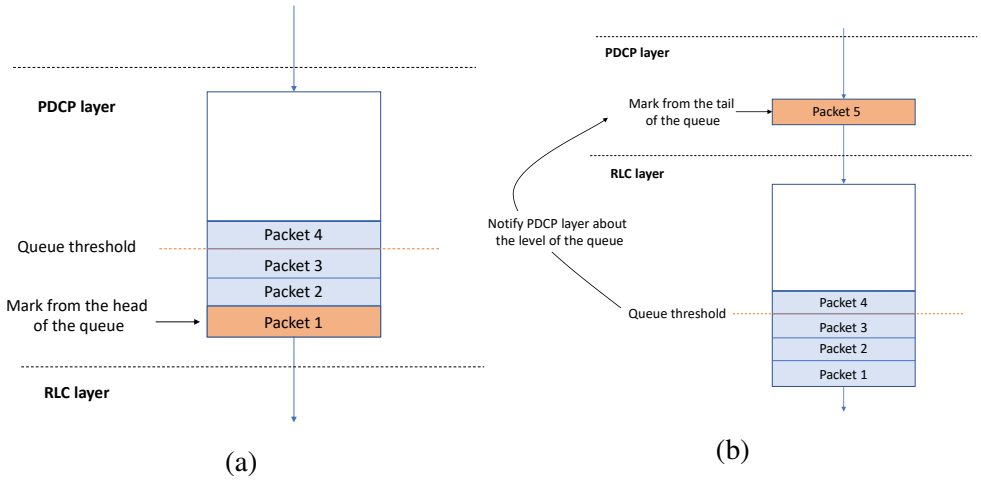


Figure 4.2: Possible solutions to enable L4S marking at gNB.

In Fig. 4.2 (a) the first solution is presented, where the queue is placed at PDCP layer. The red packet (packet 1) is the packet marked since the queue is growing above the threshold, and in this case one can see that the marking is done from the head of the queue.

Fig. 4.2 (b) instead represents the second solution: the queue is still at RLC layer, and a mechanism is present to signal the state of the queue at the PDCP layer, where new incoming packets are marked. In this case the first marked packet is packet 5 (the next packet that will enter the queue at RLC layer) that is equivalent to marking from the tail of the queue. This is not the optimal situation, since the congestion signal is delayed in the queue at RLC layer (packet 1, 2, 3 and 4 have to be sent before the packet with the congestion signal).

Both the solutions presented have pro and cons. The first solution (moving the queue to PDCP) is the one that gives best performances but it cannot be applied to the products already present in the market, while the second solution (tail marking at PDCP) is applicable to every gNB[1] but with sub-optimal performance.

[1]Unified gNB, in this work a split gNB architecture is not considered.

4.2 L4S marking strategy

As explained in Section 2.4, L4S proposes to mark packets as CE as soon as the queue delay starts to exceed a few milliseconds threshold[2]. In this way, the sender will be able to adapt its rate, keeping low queue delay and end-to-end latency, avoiding packet loss and congestion.

In this thesis work, an L4S marking strategy is proposed to fit a cellular network context, and then implemented at the PDCP layer only in the base station (gNB). The idea behind the proposed marking strategy is very simple: a packet is marked as congested accordingly to a probability called marking probability `pMark`. The marking probability is updated periodically[3] every *dT* ms, and has 4 configuration parameters. The first one is the Lower Threshold `l4sLowTh`, that represents the minimum threshold, below which a packet must never be marked. The Higher Threshold `l4sHighTh` represents the maximum acceptable threshold, above which a packet must always be marked. There is also a parameter α that aims to give a memory to the process of updating the marking probability, thus avoiding peaks and obtaining a smoothed behavior. The last parameter is `dropFromTail`, a boolean value indicating whether packets should be marked from the tail or the head of the queue. Dropping from the tail at PDCP layer is used for mimic a L4S probability updating at RLC layer (the second solution explained in Section 4.1.2). In this work, `dropFromTail` has always been set to true, to mimic the performance of RLC layer queue updating, that it is the solution that involves the sub-optimal performance. By setting `dropFromTail` to false, the performance would improve further, since the setup would represent the optimal situation.

The formula for calculating the marking probability at time *t* is composed of two steps as follows:

$$tmp(t) = \frac{qDelay(t) - l4sLowTh}{l4sHighTh - l4sLowTh} \tag{4.1}$$

$$pMark(t) = \alpha * tmp(t) + (1 - \alpha) * pMark(t - dT) \tag{4.2}$$

The variable `tmp` represents the instantaneous marking probability at time *t*, computed through a linear probability function (equation 4.1), represented also in Fig. 4.3. `tmp(t)` is then used to calculate `pMark` via the smoothing function (equation 4.2). `qDelay(t)` represents the queue delay value at

[2]The threshold can vary from implementation to implementation. Typical value is 1 ms in wired networks [35], but the thresholds have to be increased in wireless network contexts.

[3]The reason for a periodical update lies in the fact that it is computational expensive to calculate a marking probability for every packet that reaches the queue.

time t and `pMark(t-dT)` is the last computed `pMark`, where dT is the time interval between two updates. The value for dT used in this work is 5 ms.

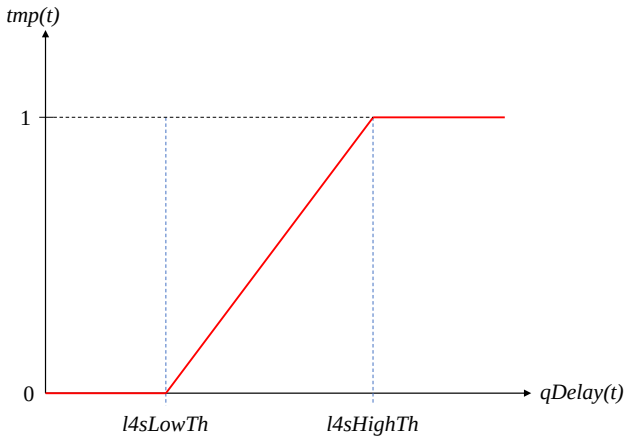


Figure 4.3: Instantaneous marking probability.

Fig. 4.3 shows how `tmp(t)` is computed. One should notice that `tmp(t)` is then averaged with the previous pMark to compute the new `pMark(t)` in equation 4.2. From equation 4.2 one can also understand that α is the parameter that weighs the contribution of the previous marking probabilities, and it belongs to range [0:1]. An α value equal to 1 implies that the marking probability updating process becomes memory-less since `pMark(t)` would be equal to the instantaneous marking probability `tmp(t)`.

α, `l4sLowTh` and `l4sHighTh` are configurable parameters and understand how changing them have an impact on the traffic KPIs is one of the goals of this thesis work.

4.3 Simulation Scenario

The simulation scenario used to test L4S is created with the idea of getting as close as possible to the real case, to have results with high external validity. To succeed in this, a very simple scenario with only one cell and one user has been initially created, to verify the correct functioning of the basic features. Then, more and more features have been added, until the final setup has been obtained.

However, the scenario built is not completely representing a 5G network. For practical reasons, since the simulation with full NR and 5GCN takes longer time, the simulation scenario is built in a way that the protocol stack is NR

except for the PHY layer that is LTE. The reason for this choice lies in the fact that simulating with the NR PHY results in simulations much more expensive in terms of time and computational resources.

It should also be specified that to reduce the simulation time, the CN is simplified. It consists of mechanisms for managing the flows that perform the same function as the UPF but in a simplified way, easier to simulate. The justification for a simplified CN to facilitate the simulation lies in the fact that the most interesting part of this study is the radio access part, and simulating a complex CN similar to the real case would not bring any benefit to assess the L4S performance.

In any case, simulation is a model of reality, and as a model, it cannot reflect every single detail of the real world. Nevertheless, simulations are performed to understand trends and hypothesize consequences applicable in the real world. The structured scenario in this thesis work has exactly this purpose: be sufficiently simple but reflect those interesting aspects that have to be analyzed to evaluate L4S in a mobile access network.

Fig. 4.4 represents a simplified version of the simulated scenario, whose implementation details are described afterwards. In particular, the simulation scenario consists of 7 base stations (gNB) with 3 sectors each, for a total of 21 cells. Each cell has 4 antennas transmitting on the base station side, while the user equipment has a single antenna. The channel model parameters are set according to the 3GPP specifications [36]. The one way latency between the server that provides the video content and the base station (gNB) is set to 25 ms. This fixed delay is chosen to mimic the adverse use-case where the video gaming server is relatively far from the end user, as can happen for server located in larger countries[4]. It is an adverse scenario because when the delay in the network is higher, the propagation of congestion signals and the reports of congestion is slower, leading also to a slower response to the congestion at the endpoints. The number of users running latency-critical interactive applications is fixed in each simulation, and simulations with different numbers of users are carried out. To further stress the network and get closer to simulating a real use case, web traffic users are also present in the system, in proportion to the number of video users. All the users are distributed randomly in the scenario and they are served from the cell that have best Reference Signals Received Power (RSRP). The users do not have movement and stay in their position during the whole simulation. The carrier frequency used is 600 MHz

[4]Considering optic fiber links between the server and the base station, 25 ms represent a distance in the order of 1500 km. Results obtained with 2.5 ms one way latency are reported in Appendix B

with a carrier bandwidth of 9 MHz. A Frequency Division Duplexing (FDD) transmission scheme is used and the protocol stack used in the RAN is compliant with the 3GPP specification and standards. The simulation time is 100 seconds. Details related to the traffic models are analyzed in the next Section.

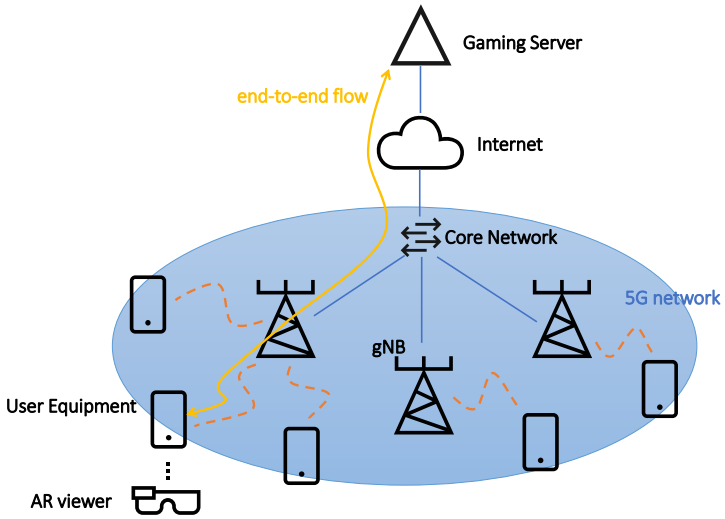


Figure 4.4: Simplified Simulated Scenario.

In Fig. 4.4, the Gaming Server represents the sender, while the receiver is the User Equipment, and they are responsible for the congestion control. The L4S marking is done at the gNB.

The number of video gaming users in the system is the parameter that varies from simulation to simulation, and consequently also the number of web users (which is proportional to the number of video users) is variable. In particular, the number of video users is 2, 5, 10, 15, 30 and 50. The proportionality factor chosen between video and web user is 10, which means that the number of web users in the system is 10 times greater than the number of video users. For instance, when there are 2 video users in the system, there are 20 web users, while with 50 video users, the web users become 500.

In the scenario described there are three aspects that can differ from simulation to simulation: the support to L4S, the scheduler used for the video gaming traffic and the presence of AQM. The support to L4S means that some simulation are run without L4S support, and the results are considered the baseline for a comparison with simulation where L4S is enabled. The different schedulers used are explained in Section 4.5. Regarding the AQM, it has to be specified that in the simulated scenario, by default the network nodes

have queues with unlimited memory and no AQM. This means that no packets are marked or dropped from a network node, and consequently the packet loss rate, by default, is zero. When the AQM is enabled packets can be dropped and so there is the possibility to have packet loss. The AQM policy consists in dropping only the oldest packet in the queue when the queue delay is above 100 ms, while if the delay is above 500 ms all the oldest packets are dropped until the queue delay return below 500 ms.

4.4 SCReAM and traffic model

The goal of L4S is to achieve low end-to-end delays, to meet the requirements[5] of interactive latency-critical applications such as AR, VR, online gaming and remote control. For this reason, the simulations are carried out with a video traffic model, using SCReAM as congestion control. SCReAM is a window-based and byte-oriented congestion control protocol intended for Real-time Transport Protocol (RTP) traffic. SCReAM is based on the self-clocking principle [39] and implements a technique similar to the one proposed in [40] to estimate the network queue delay. It is chosen as a congestion control for this thesis work since it supports L4S and it is designed to work with video traffic, that is the type of traffic typically used in the interactive latency-critical application.

[5]From [37] it is recommended a latency below 100 ms for acceptable video gaming performance. Still it is hard to define an exact hard requirement as this depends heavily on the type of game and how experienced the gamer is. It is however important to try and reduce latency as much as possible on all parts of the chain. As regarding throughput requirements, with at least 2 Mbps it is possible to stream 1080p resolution video content with a decent quality [38].

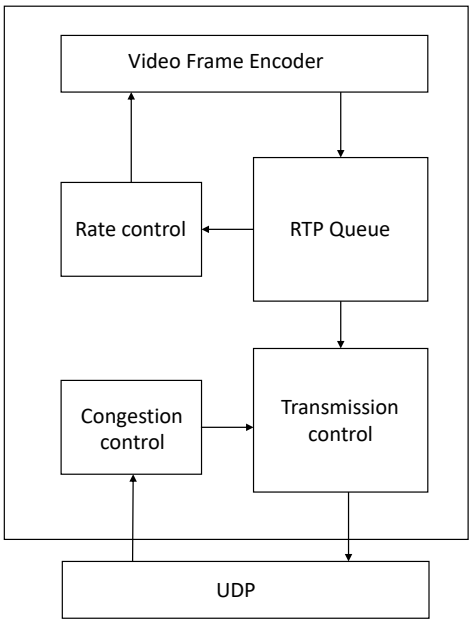


Figure 4.5: Sender Side architecture.

Fig. 4.5 shows the structure of the sender side application, that in the case of this work is the server that generates the video gaming traffic. The video frame encoder is the first element of the chain and is responsible for the generation of the video frames and encoding them to RTP packets. These packets are then kept in the RTP queue, waiting to be transmitted. The transmission control block has the task of sending packets stored in the RTP queue, based on the optimal bitrate provided by the network congestion control block. The latter is notified about congestion in the network through the Real-Time Control Protocol (RTCP) messages. The block of rate control adaptation instead has the task of adapting the rate of the video frame encoder. Based on the size of the RTP queue and adapting the video rate, it is possible to reduce the video frame delay. The video frame delay as explained in Section 3.3 is calculated from the moment the frame is generated by the video encoder on the sender side until it is recreated at the client-side application, so the time spent by the frame in the RTP queue contributes to increase the video frame delay.

Looking specifically at the traffic model, an *H.264* video encoder [41] is used in this work. The encoder outputs video frames with a bitrate range set to 2-70 Mbps. This range mimics the encoding at 4K resolution at the higher bitrates, with the ability to switch down to 1080p resolution at the lower bitrates. However, the video bitrate cannot go below 2 Mbps, as this would give a too poor video gaming experience, but at the same time it means that if the users

have a throughput lower than 2 Mpbs, then the video frame will be queued in the RTP queue at sender side, increasing the video frame delay.

The video content represents a real video-gaming model for Augmented Reality application. This video model trace file, provided internally, contains several scene changes and features of true video gaming traffic, which are not trivial to manage both from the video encoder and the network. The several scene changes require the creation of large video frame, also called I-frames, and as one can see from Fig. 4.6 the I-frames are generated every 2 seconds.

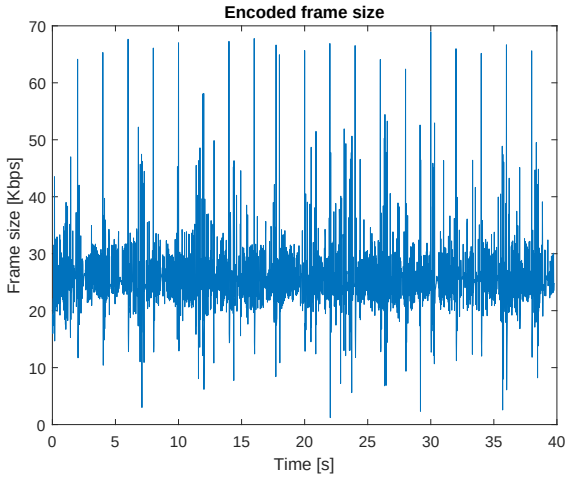


Figure 4.6: Encoded video frame size.

Fig. 4.6 reports the encoded frame size of the video gaming content used in this work. The traces are 40 seconds long, and since the simulation time is 100 seconds, when the time traces finishes, it is repeated from the beginning.

As for the background traffic, a web traffic is used to simulate real-case traffic. The model first sends an index web page. The size of the page follows a normal distribution with mean of 10710 bytes and a variance of 25032 bytes. After that, a number of web embedded objects are sent in parallel. The size of the embedded objects follows normal distribution with mean 7758 bytes and standard deviation of 126168 bytes. The number of embedded objects follows a Pareto distribution with location 15 and shape 3.1. The client reads the web page content for a random time that follows an exponential distribution with mean 5 seconds and maximum reading time of 7 seconds. After that the user requests a new web page, and this is done until the simulation stops.

The reason for using a challenging video model combined with the background traffic that constantly hammers the network, is made to create a complex scenario, in order to test the benefits of L4S in a plausible environment.

4.5 Scheduler

As explained in section 2.5.1.2, the task of the scheduler is to map traffic packets to the available resources. The scheduler strategy can differ from implementation to implementation, but the decision strategy of allocating resources always uses a priority mechanism to determine the final decision. Prioritizing one traffic over another can have a major impact in terms of performance, especially with regards to latency. For this reason, in this work, two different schedulers are used with different priority for the video gaming traffic, while web traffic is always kept with the default priority. The two scheduler used are Round Robin (RR) and Delay Based Scheduler (DBS). RR is the simplest scheduler, that allocates resources once each among the queues that have the same priority. DBS instead always schedules first the oldest packet among the queues, and it also assigns higher priority to queues with packets older than a predefined threshold. In this work the DBS threshold is set to be equal to `l4sHighTh`. The reason for this choice is to keep the latency as low as possible: if the queue delay is above the `l4sHighTh` then the packets are prioritized. To make an example, assuming a DBS with threshold 10 ms, if among the video gaming users queues one packet is older than 10 ms, the whole queue is prioritized and the oldest packet is the first packet to be scheduled. When there are no more packets older than 10 ms in the queue, the queue came back to the default priority. RR instead, schedules packets of the first queue in slot 1, while in slot 2 schedules packets from queue 2, and so on among all the queues that have the same priority.

In this work the two scheduler just described are also used in combination with absolute priority for the video gaming traffic. Absolute priority means that the video gaming traffic always has higher priority than the web traffic. The RR with absolute priority then become a RR only among the video gaming queues, and when there are free resources the web traffic is scheduled. Instead, DBS with absolute priority still mean that the oldest packet will be scheduled first, but there is no threshold to increase the priority since it is higher than the web priority by default.

To be precise, the scheduler is not as simple as it is explained in this section, and it has mechanisms to increase the priority for the traffic of users that are not served for long time. These mechanisms are needed, otherwise, when the video gaming traffic has higher priority, the web traffic would be starved out.

Chapter 5

Results

In this chapter, the most interesting results are reported and discussed. In Section 5.1 the impact of α on the results is assessed. In section 5.2 the impact of the thresholds is analyzed. Finally, in section 5.3 the main results that represent the KPIs are reported.

5.1 Finding alpha

α, that has been explained in Section 4.2, is one of the configurable parameters in the L4S implementation. It is important to understand the impact of this parameter in order to set it to a correct value that can provide better performance in terms of throughput and latency. In this Section, the behavior of the KPIs when α changes is analyzed.

Initially, to try to determine the behavior, less complex simulations have been carried out, with only 4 video gaming users in the system, 3 base stations, 9 cells, and FTP background traffic. The video gaming users are using the same video model explained in Section 4.4, while the background traffic consists in some users downloading 1 Mb file and other users downloading a 100 Kb file via FTP. The arrival process of the FTP users is on average 2 new users per second that download the 1Mb file, and 20 new users per second for the 100 Kb file. All the other parameters such as carrier frequency, carrier bandwidth are the same of the scenario explained in Chapter 4. This simplified scenario is chosen because these simulations are performed on a laptop with limited resources, so a trade-off is needed to reduce the simulation time.

The results of these simplified simulations are shown in Tables 5.1 and 5.2. The results refer to the scenario where all the parameters were kept constant except for α.

α	L4S RR	L4S DBS
1.0	16.6892	21.0215
0.5	17.0274	21.1912
0.25	16.9791	21.0817
0.125	15.9100	19.3505
0.0625	15.4454	18.4246

Table 5.1: Mean transmitted rate [Mbps].

Table 5.1 reports the values for the video transmitted rate in the case of L4S with RR scheduler and L4S with DBS, varying α.

Alpha	L4S RR	L4S DBS
1.0	3.1173	3.0237
0.5	3.2016	3.0194
0.25	3.1712	2.9241
0.125	2.7426	2.6617
0.0625	2.8118	2.5393

Table 5.2: Mean Network Queue delay [ms].

Table 5.2 reports the results for the queue delay experienced in the network, in the case of L4S with RR and with DBS, varying α.

It can be seen from Tables 5.1 and 5.2 that the differences in terms of transmitted rate and queue delay are minimal when α changes, regardless if using RR or DBS. This indicates an intrinsic stability of the performance concerning the variation of α, or in other words, changing α does not impact significantly the performances.The reduction that can be noted in the transmitted rate is to be attributed to the throughput-latency trade-off: where the transmitted rate is higher, the queue delay is higher, when the queue delay is lower also the transmitted rate is lower. These results refer to the simplified simulation described at the beginning of this section, and even if they do not have statistical value[1] they were a guideline on how to proceed with more complex simulations.

[1] Since the simulation were run only once and with a limited number of users, the results could be obtained from a lucky scenario. Statistical valid results are obtained averaging the results of several simulation with a reasonable number of users in the system.

Reasoning on equation 4.2, the formula for calculating the marking probability, one can note that a value of $\alpha = 1.0$ implies the absence of memory in the process, while $\alpha = 0.0625$ implies very little importance of the current situation of the queue. For this reason, in the most complex simulations, to reduce the number of simulations to run, only three different α values are analyzed: 0.5, 0.25 and 0.125. The results reported in Fig. 5.1 show the impact of α in simulations with statistical validity. The results refer to the scenario described in Chapter 4 where L4S traffic is supported by the DBS, `l4sLowTh` = 5 ms and `l4sHighTh` = 10 ms. The AQM is disabled in the system, so it is impossible to have packet loss.

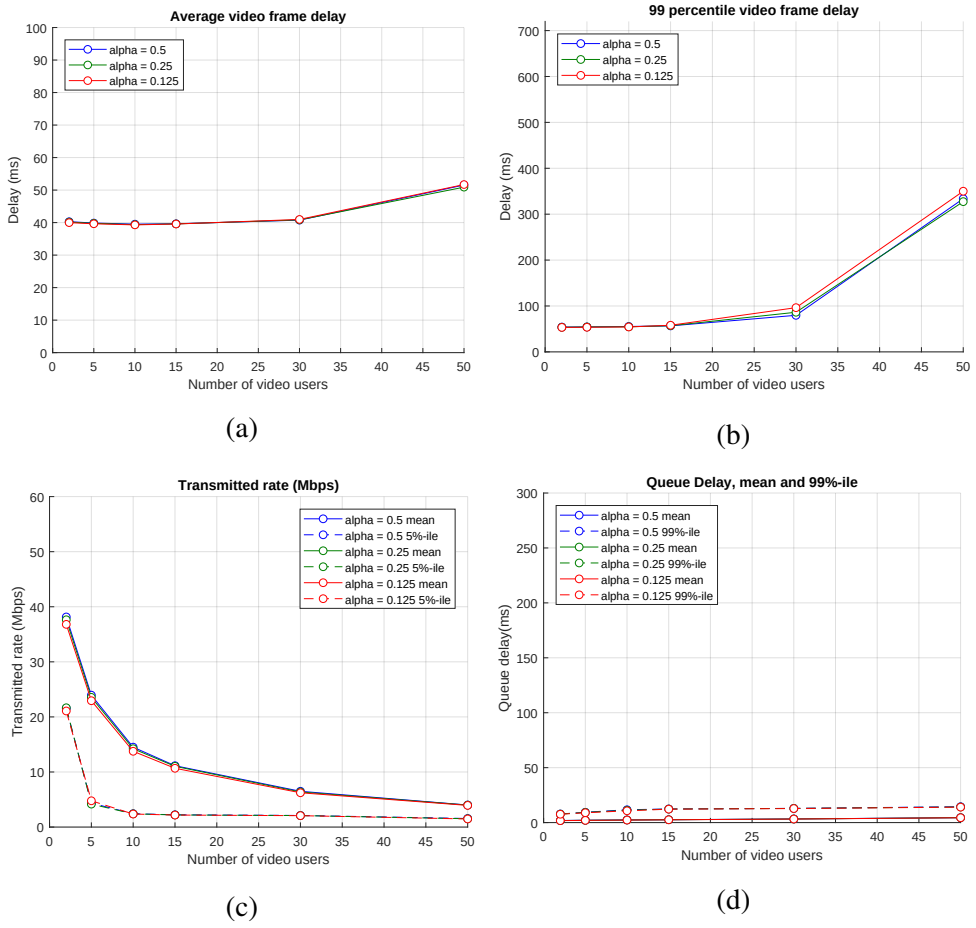


Figure 5.1: α comparison, DBS scheduler.

Fig. 5.1 shows that the variation of α does not change the metric analyzed: the lines essentially overlap. The only case in which the α variation implies

a variation in the result is the 99th percentile of the video frame delay, where alpha with $\alpha = 0.125$ give slightly higher video frame delay at high load in the system. Also, looking at the transmitted rate $\alpha = 0.125$ leads to slightly lower performances. Fig. 5.2 reports the results when RR scheduler is used.

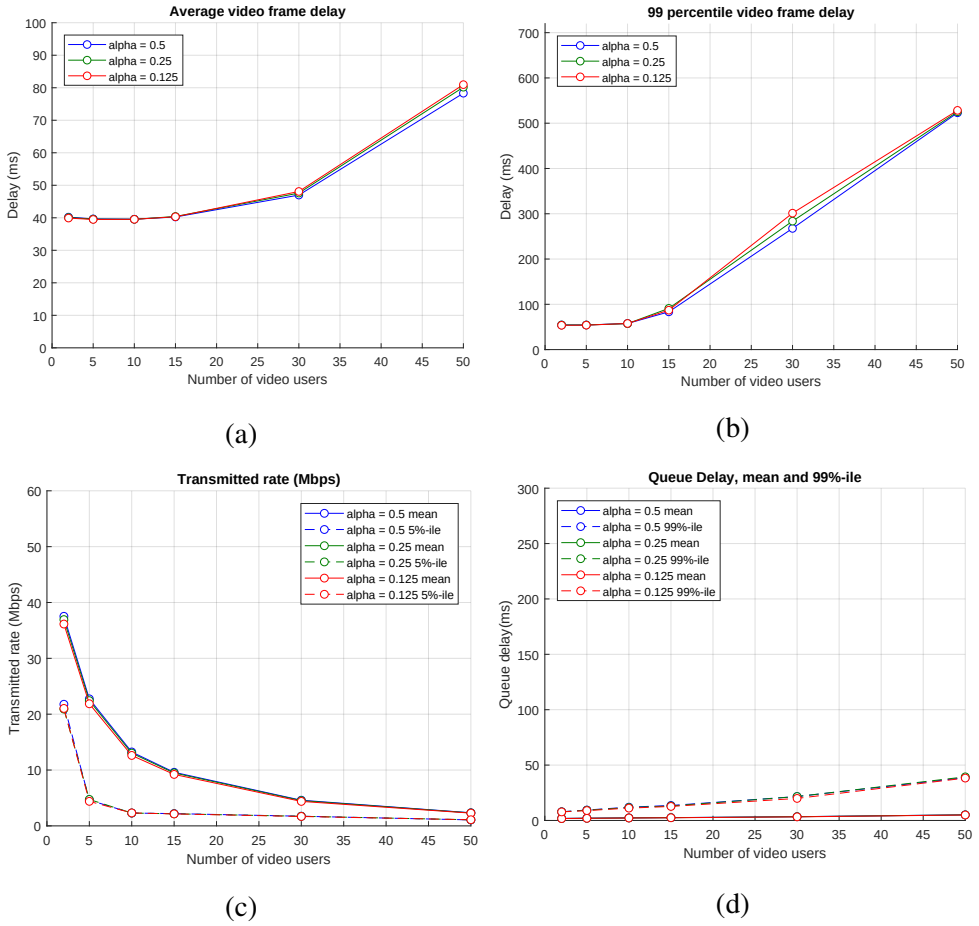


Figure 5.2: α comparison, RR scheduler.

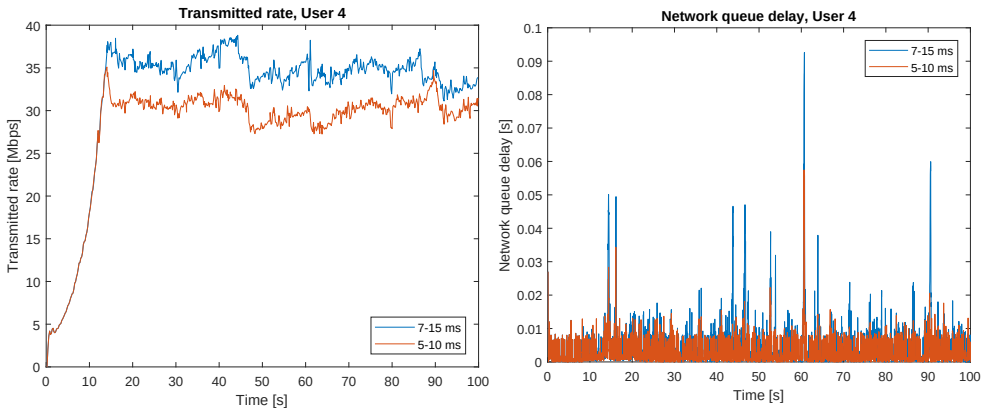
Also in Fig. 5.2 the lines overlap, with some small discrepancies in the video frame delay when the load in the system is above 30 users.

After analyzing the results reported in Fig. 5.1 and 5.2, we can assert that α does not significantly cause a change in the performance. The recommended values are $\alpha = 0.5$ or $\alpha = 0.25$, which represent a good trade-off between the instantaneous state of the queue and the history of the process.

5.2 Determine the thresholds

`l4sHighTh` and `l4sLowTh`, from now *threshold span*, are the other variable parameters that can be modified in the L4S implementation. In this thesis two threshold spans are analyzed: 5-10 ms and 7-15 ms. The rationale for the choice of these two threshold spans is that, in RAN, the scheduling can involve jitter, especially when there is other traffic in the system. Furthermore, retransmissions can frequently occur on the MAC layer. This is reflected as delay in the queue that is monitored for the L4S marking. The thresholds are chosen to give a balance between reaction to the delay jitter and actual queue delay caused by congestion. Given that 5G RAN can implement scheduling in many ways, and also use different numerology (a.k.a transmission slot lengths), this means that the marking thresholds can differ between product implementations to give a an optimal balance between queue delay, jitter and throughput.

Initially, a scenario with only 4 video gaming users in the system, 3 base stations, 9 cells, and FTP background traffic is used. DBS scheduler supports the L4S traffic. The video gaming users are using the same video model explained in Section 4.4, while the background traffic consists in some users downloading 1 Mb file and other users downloading a 100 Kb file via FTP. The arrival process of the FTP users is on average 2 new users per second that download the 1Mb file, and 20 new users per second for the 100 Kb file. All the other parameters such as carrier frequency, carrier bandwidth are the same of the scenario explained in Chapter 4. The results are shown in Fig. 5.3, and they focuses on a single user.



(a) Transmitted rate

(b) Network queue delay

Figure 5.3: Threshold span comparison, simplified scenario.

Figure 5.3 report the the time-trace of transmitted rate (a) and network queue delay (b) for a specific user (User 4). The results related to the other 3 users are omitted since they reflect the same behavior of user 4. Again, the trend is confirmed: lower threshold span (5-10 ms) implies lower delay, but also lower throughput performance.

Finally, these two threshold spans have also been used in larger simulations using the scenario explained in Chapter 4. The results shown in Fig. 5.4 are obtained with a DBS, $\alpha = 0.25$ and without AQM enabled in the system (packet loss rate is zero).

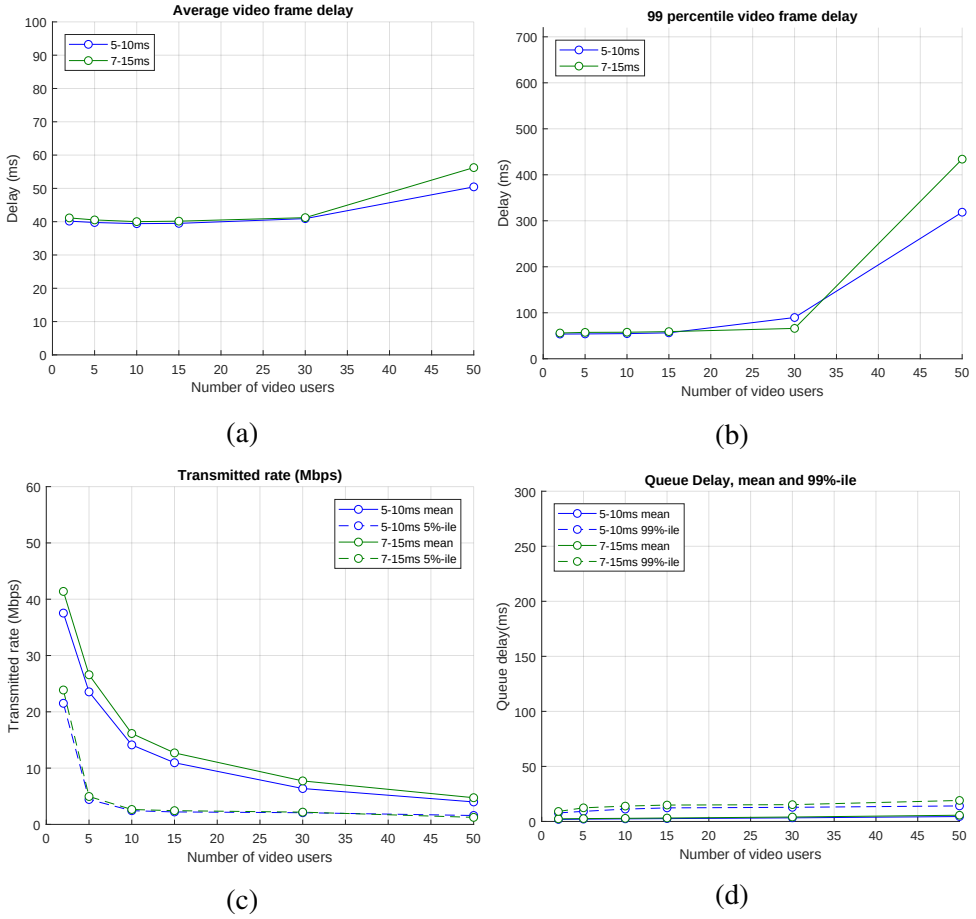
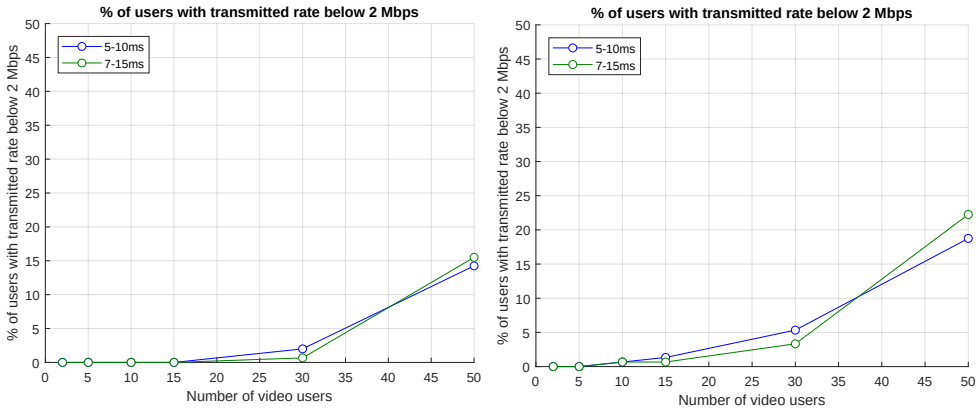


Figure 5.4: Threshold comparison, DBS scheduler.

The results in Fig. 5.4 confirm the throughput-latency trade-off: the 5-10 ms span provides lower latency, but at the same time the throughput performance are lower than using the 7-15 ms span. As expected, 5-10 ms involves lower queue delay. One strange results is the 99th percentile of the video frame

delay at 30 users: 7-15 ms is lower than 5-10 ms. This pattern is the same also changing alpha and changing the scheduler, and Fig. 5.5 explains this trend: with 7-15 ms threshold and 30 users in the system, there are fewer users with transmitted rate below 2 Mbps. 2 Mpbs is the minimum bitrate produced by the video encoder, and so having a link throughput below 2 Mbps means that packets are delayed in the RTP queue at sender side. Thus, higher RTP queue delay involves higher video frame delay.



(a) DBS

(b) RR with abs priority

Figure 5.5: % of users with transmitted rate below 2 Mbps.

As one can see in Fig. 5.6 where the RR scheduler with absolute priority is used, the trend is the same highlighted in Fig. 5.4.

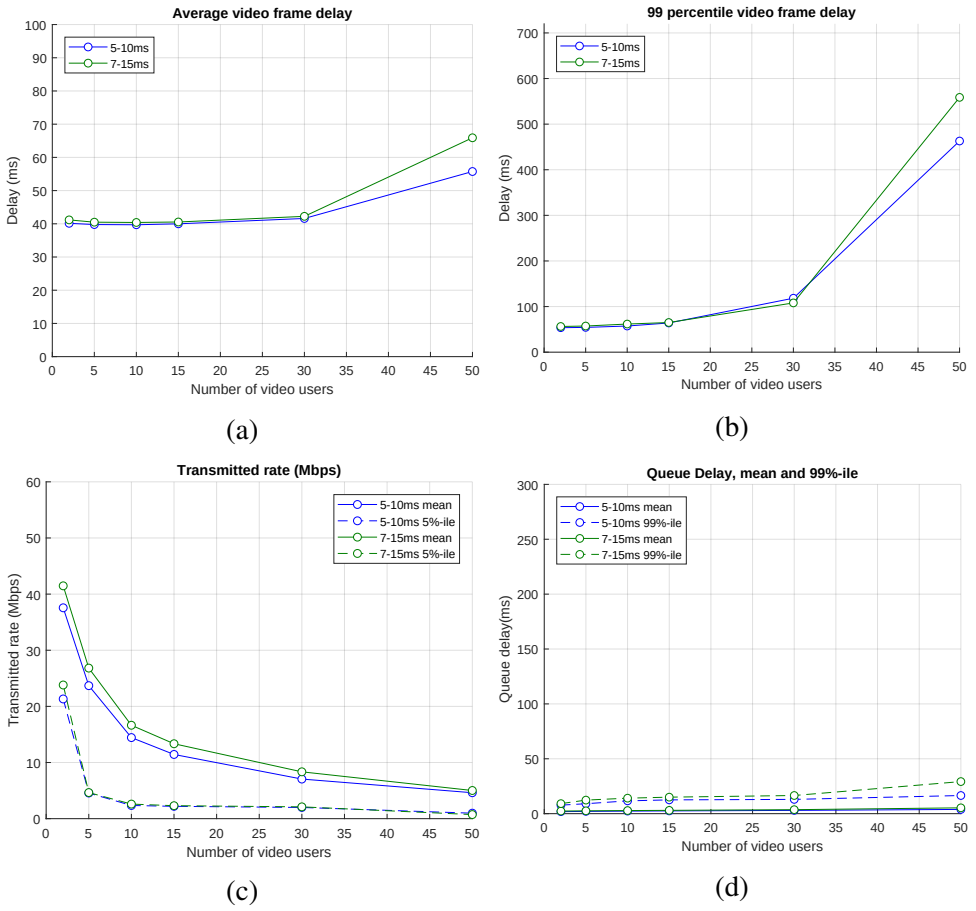


Figure 5.6: Threshold comparison, RR with absolute priority.

Fig. 5.6 highlights that also in this case the the trade-off between throughput and latency is confirmed. Another result is given from Fig. 5.6 (d), where it is shown that the 99th percentile of queue delay with the threshold span 7-15 ms starts to diverge from the 5-10 ms trajectory when the system load increase.

By comparing Fig. 5.4 (DBS) and Fig. 5.6 (RR with absolute priority) one can notice some difference in performance between the two schedulers, but analyze them is not the scope of this section, that want to investigate the impact of changing the threshold span. The analysis about performances among the schedulers is presented in Section 6.1.

In conclusion, the threshold span is a parameter that has an impact on the final performance. The two threshold span analyzed highlighted the well known throughput-latency trade-off, and it would be interesting to investigate in a future work other possible spans to see if there are combinations that make the trade-off more convenient.

5.3 Main results

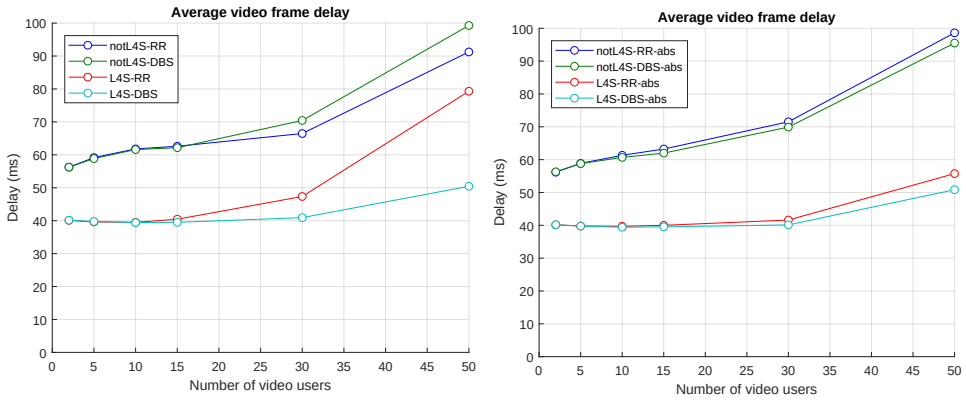
In this section the statistical valid results are presented. The results reported highlight the differences between the two cases, with and without L4S support in the network. Different schedulers were used: RR, RR with absolute priority for video gaming traffic, DBS and DBS with absolute priority for video gaming traffic. For simplicity of representation and to avoid reporting a large number of graphs, given that the changes in performance related to α and threshold span parameters were discussed in the previous sections, the following results refer to the case α = 0.25 and threshold span 5-10 ms (the threshold span that provides lower latency). The scenario used to obtain all the results reported in this section is described in Section 4.3. The AQM is not enabled and so the packet loss rate is not reported as result, since there are no packet loss. All the results do not consider the statistics of the first 20 seconds of the simulations.

5.3.1 Video Frame Delay

The video frame delay is an important metric to asses the quality of the video experienced by the end users of an application. The video frame delay is the sum of different delays, as explained in Section 3.3.

L4S is a technology proposed to reduce the video frame delay, by minimizing only the component of the video frame delay that is not constant: the queue delay. Even if not in a direct way, L4S at the same time enable the congestion control to be more precise and aware of the status of the network. In this way the congestion control algorithms can manage to adapt the transmitted rate to minimize the latency.

In this Section, the video frame delay results obtained using different scheduler and priority are reported. Fig. 5.7 and 5.8 show the average and the 99th percentile statistics.

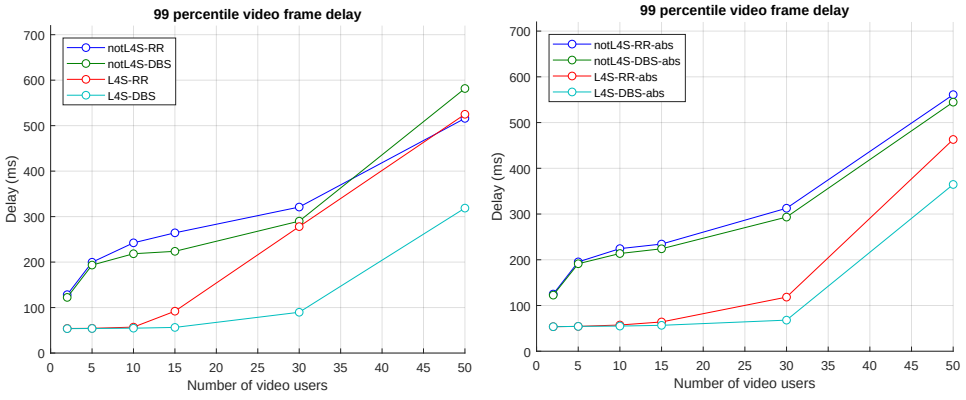


(a) (b)

Figure 5.7: Average video frame delay.

One can see from Fig. 5.7 (a) and (b) how the average video frame delay performance are always lower using L4S, even at high loads (50 video gaming users, 500 web users). In particular, from Fig. 5.7 (a) the use of DBS seems to be the best solution for L4S, while for the not-L4S case RR without priority is the suggested choice.

Looking at Fig. 5.8 (a), one can see that also in this case L4S shows lower delay, although using DBS the difference is more evident than with RR. It is also noted that using RR without L4S, lower delay is reached at high load in the system. The motivation for this trend will be discussed in the next sections.

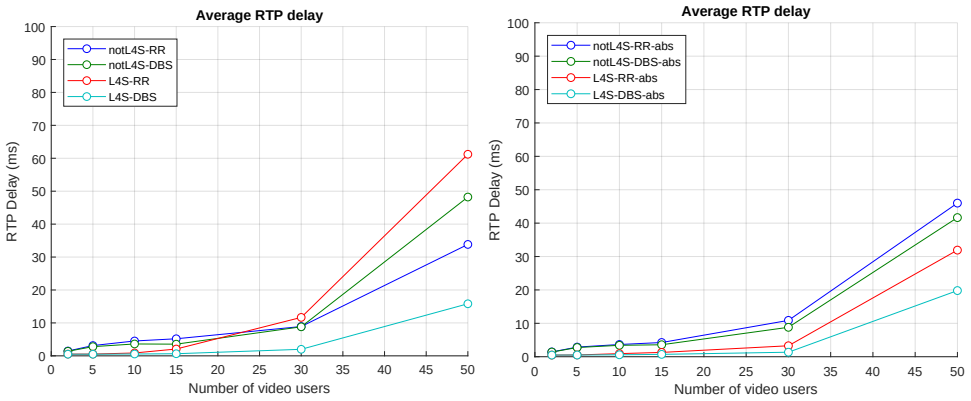


(a) (b)

Figure 5.8: 99th percentile video frame delay.

5.3.2 RTP Delay

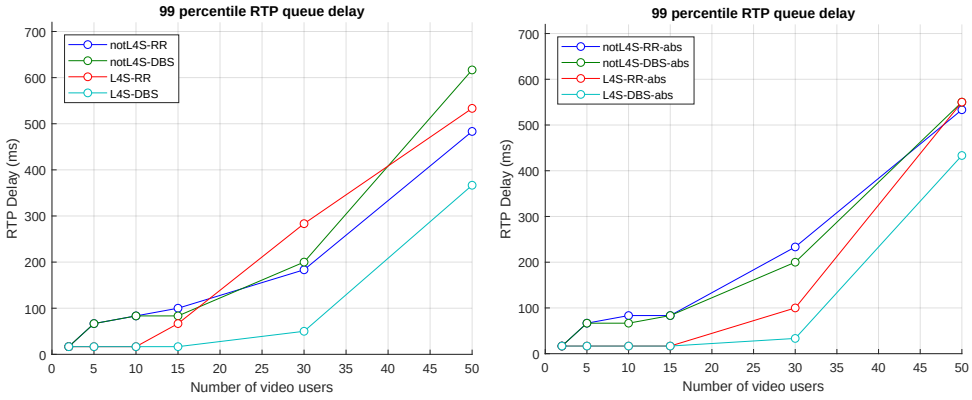
By RTP delay is meant the delay in the RTP queue present at the transmitter side, in this case, the server that provides the video content. Before being injected into the network and transmitted, the RTP packets are buffered in a queue, as explained in Section 4.4. It can happen that when the RTP queue grow excessively, with the aim of keeping low latency, some packets can be discarded and never been sent. This delay in many cases, especially at high loads and when the users have poor link throughput, represents the major component of the video frame delay. It should be noted that L4S is not a technology to break down this delay directly, but in any case, better signaling of the network status implicitly involves better management and reduction of the RTP delay on the sender side.



(a) (b)

Figure 5.9: Average RTP delay.

Looking at Figures 5.9 and 5.10 a trend is evident: the RTP delay is lower using L4S and DBS. When using RR, things are different, and they also change between RR and RR with absolute priority. The explanation for this behavior is given as a consequence of the L4S marking process, and the response of the congestion control algorithm (SCReAM). What happens in the case of RR, is that the video gaming traffic and the background traffic have the same priority. Consequently, the queue delay at the PDCP layer increases faster than the scenario with RR with absolute priority or DBS. To keep low queue delay, L4S marks more video gaming traffic packets, and the congestion control mechanism reacts to L4S signals by reducing the transmitted rate. When the transmitted rate gets close to (or below) the lowest possible encoder bitrate it causes an increasing in the RTP delay. If the RTP queue delay grows exces-



(a) (b)

Figure 5.10: 99th percentile RTP delay.

sively, SCReAM responds by discarding from the RTP queue, that will end up in the generation of a new I-frame by the video encoder. This is a vicious circle which unfortunately is not easy to manage, given the numerous components involved. To avoid running into this problem, a possible solution is to use a scheduler that provides different priority to L4S traffic and background traffic. Another possible solution is to allow resolution lower than 1080p, that make it possible to the video encoder to operate at lower bitrates. The last possible alternative is to terminate the sessions that have low bitrate, either from the user application or the network side.

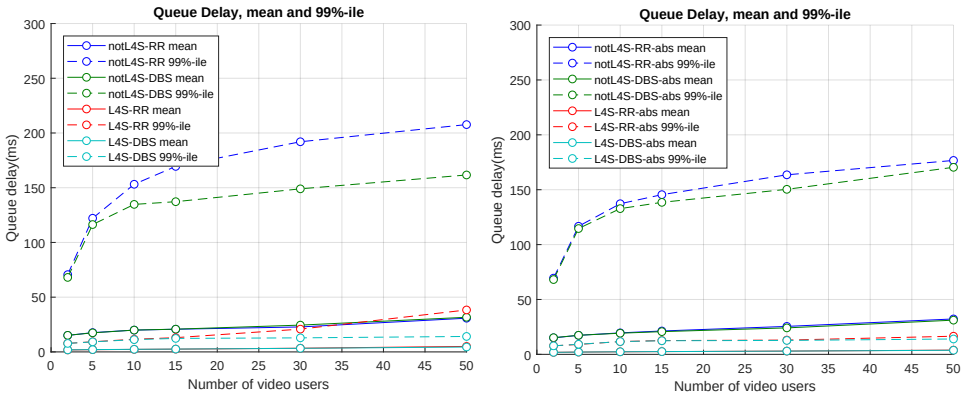
The higher RTP delay also explains the higher video frame delay (Fig. 5.7 and 5.8), since RTP delay is one of the component of the video frame delay.

5.3.3 Network Queue delay

Network Queue delay is another component of the video frame delay, and it is the delay that L4S aims to minimize. The Network Queue Delay is estimated by the SCReAM congestion control algorithm, and represent the sum of the queue delay experienced in every node of the network. In this specific case, the Network Queue Delay corresponds to the queue delay at PDCP layer, that is the only point in the network where packets can be queued in this work. Figure 5.11 shows both the average and the 99th percentile Network Queue Delay.

Fig. 5.11 confirms the explanation given in Section 5.3.2: the queue delay is higher using RR without priority, therefore L4S will mark a greater number of packets as one can see in Fig. 5.12, creating the vicious circle, wherein

RTP packets are queued up on the sender side because the bitrate provided by the video coder is higher than the actual link throughput. It is interesting to note how significant the improvement introduced by L4S is: apart from the RR case (Fig. 5.11 (a)), 99th percentile of the queue delay obtained with L4S is always lower than the average queue delay obtained without L4S. The 99th percentile is approximately 6 times lower when L4S is used.



(a) (b)

Figure 5.11: Average and 99th percentile Network Queue Delay.

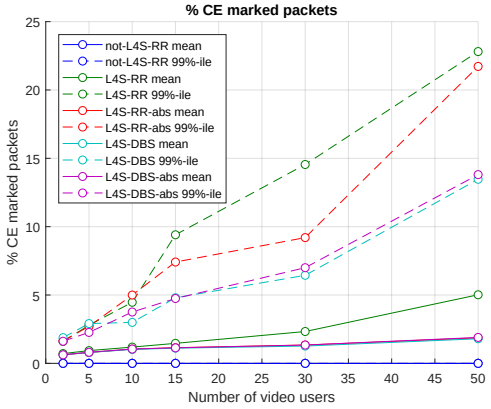


Figure 5.12: Percentage of CE marked packets.

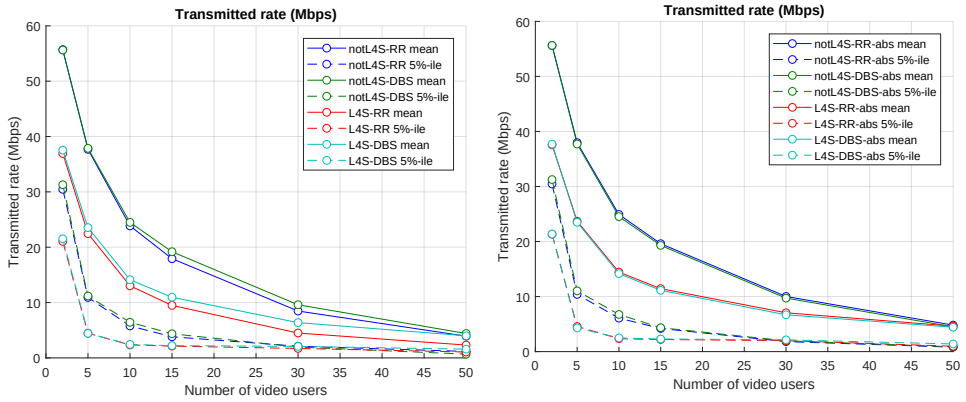
Fig. 5.12 shows that the number of marked packets using RR with the default priority involves the highest rate of CE marked packets. On average the other three scheduler involves the same amount of CE marked packets, but looking at the 99th percentile, the DBS is the one with the lower CE marking

rate. The CE marking rate is zero for the not-L4S cases since the packets does not support L4S marking.

5.3.4 Transmitted Rate

In this subsection the transmitted rate is analyzed. The transmitted rate is the application layer throughput (a.k.a. layer 5 throughput). Figure 5.13 shows the average transmitted rate, and the 5th percentile, that represents the performance relating to the worst 5% of users.

It can be noted that in both Fig. 5.13 (a) and (b) the trend is always the same: the transmitted rate decreases as the load on the system increases. Furthermore, as expected, the transmitted rate in the case without L4S is higher (because of the throughput-latency trade-off), even if the difference between L4S and not-L4S is getting smaller at high load in the system. At high load L4S and not-L4S have the same average transmitted rate, because SCReAM tries to keep low latency also in the case without L4S. Thus, since at high load in the system the queues grow faster, SCReAM reduces the transmitted rate to reduce the latency. This leads to small difference in terms of transmitted rate with and without L4S when the system load increase. The last observation from Fig. 5.13 (a) is that using DBS leads to higher throughput than using RR.



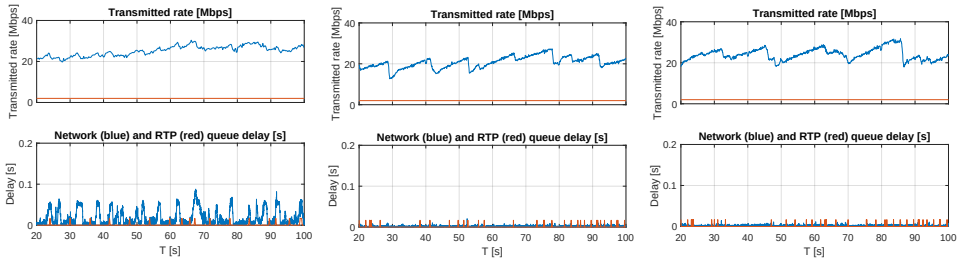
(a) (b)

Figure 5.13: Average and 5th percentile transmitted rate.

5.3.5 Time traces

This section is intended to provide the time traces of transmitted rate, RTP and Network Queue Delay of a specific user. Each Figure reports 3 cases:

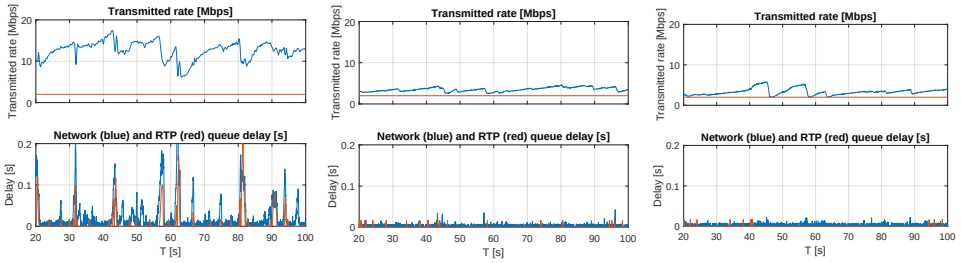
not-L4S with RR, L4S with RR and L4S with DBS. The time traces refer to user in the scenario described in Section 4.3 in the specific case with 15 video gaming users in the system. Time traces start from 20 seconds as happens for statistical analysis. This was intended to avoid considering the first few seconds of simulation, where throughput and delay are not stable due to the startup phase. Fig. 5.14 shows the trend of a user who is in a good position, and this can be seen from the high transmitted rate. Fig. 5.15 instead reports the results for a user that is not in an optimal position, and for this reason is subjected to lower transmitted rate than the user of Fig. 5.14. Finally Fig. 5.16 shows the case of a user that is in an unfavorable position, with a consequent very low transmitted rate.



(a) Not-L4S RR (b) L4S RR (c) L4S DBS

Figure 5.14: Time trace comparison, user in good position.

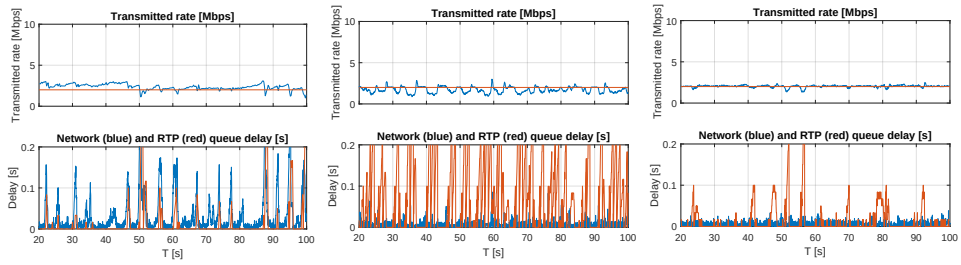
Looking at Fig. 5.14 it is clear that the transmitted rate in the not-L4S case is higher than when L4S is used. However, the network and RTP queue delay in the not-L4S case is much higher and less stable. This is exactly what is expected from the use of L4S: improvement in the queue delay with the compromise of a reduction in transmitted rate. A further observation is that comparing the two L4S cases, L4S with DBS is the one that provides greater transmitted rate, as was also highlighted in Section 5.3.4.



(a) Not-L4S RR (b) L4S RR (c) L4S DBS

Figure 5.15: Time trace comparison, user in reasonably good position but subjected to reduced throughput.

Figure 5.14 highlights a reduction in transmitted rate when using L4S, however in Fig. 5.15 the reduction in transmitted rate is clearer. Also in Fig. 5.15 L4S means lower transmitted rate and lower queue delay. It is also confirmed that DBS is better than RR because it can provide slightly higher transmitted rate and at the same time it avoids sporadic queue delay spikes that happen with RR.



(a) Not-L4S RR (b) L4S RR (c) L4S DBS

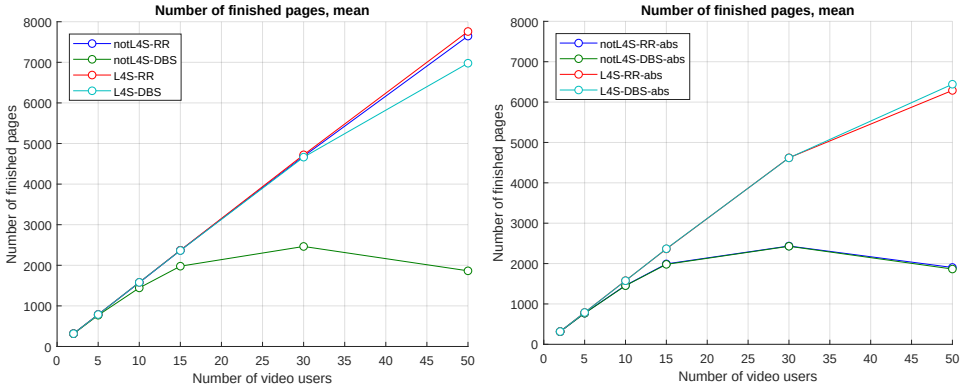
Figure 5.16: Time trace comparison, user in bad position.

Fig. 5.16 reports the case of a user subject to a very low transmitted rate even in the not-L4S case. Using L4S with RR involves very high RTP queue delay. This because the transmitted rate is lower than 2 Mbps for the majority of the simulation time. 2 Mbps is the minimum output bitrate of the video encoder, and therefore if the transmitted rate is lower than 2 Mbps many packets are kept in the RTP queue at sender side. Using a DBS, the user is served with a transmitted rate almost always greater than 2 Mbps. This is reflected in a lower RTP queue delay when compared with the RR case. In any case, it should be noted that in both L4S cases the network queue delay is kept low. The purpose of L4S is to maintain a low queue delay, and the time traces con-

firm that L4S work well even in cases where the transmitted rate performance are poor.

5.3.6 Web traffic performance

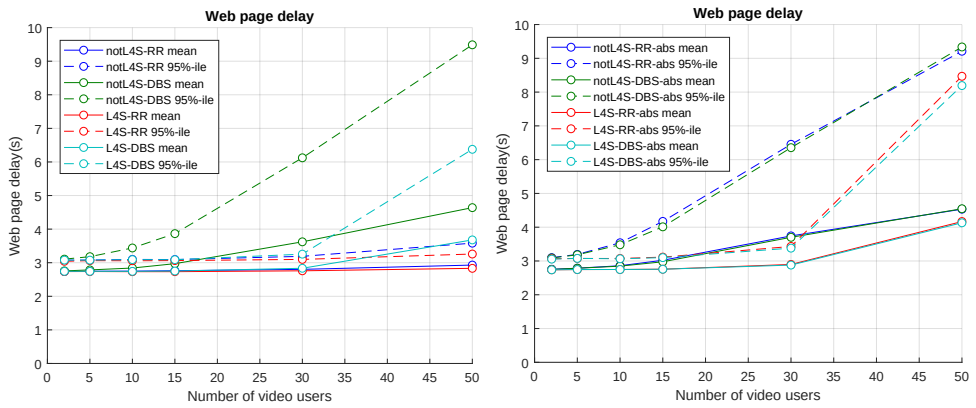
This section illustrates the results relating to the performance of the background web traffic. As explained in Section 3.2, the number of web users in the system is 10 times greater than the number of video users. Fig. 5.17 shows the performance in terms of the number of pages downloaded and read by users, while Fig. 5.18 the download page delay. The model used for the web traffic is explained in Section 4.4.



(a) (b)

Figure 5.17: Number of finished pages.

In Fig. 5.17 (a), the RR scheduler shows a linear increment of the number of finished pages when the load increases, both for L4S and not L4S case. When a scheduler with priority supports the not-L4S capable traffic, the result is that the web traffic is pushed back, as can be seen from the convex behavior in 5.17 (a) and (b). The reason for the convex behavior is that the scheduler is prioritizing the video gaming traffic over the web traffic. When L4S support is present, the web traffic has higher number of finished pages, because the L4S marking causes a rate adjustment from the congestion control, leaving space for the web traffic. This is a good feature, as the combination of L4S and priority scheduling leaves resources also for the web traffic and thus avoids starvation of the latter. The results indicate that the priority scheduling used in this thesis should only be used in companion with a clear congestion signal as L4S marking.



(a) Round Robin

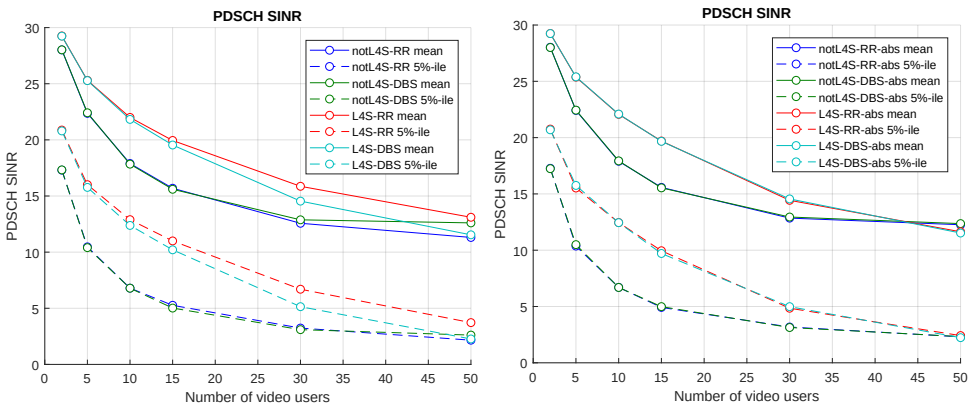
(b) Round Robin with absolute priority

Figure 5.18: Web page delay.

Fig. 5.18 reports the web page delay, that is the time that pass between the web page request to the moment the whole page is received. As happens in Fig. 5.17 (b), also in Fig. 5.18 (b) two different behavior can be distinguished. The web page delay is higher in the not-L4S case for the same cause explained above: scheduler is prioritizing the video gaming traffic pushing back the web traffic. When there is L4S support for the traffic prioritized, the congestion signal and the congestion control mitigate the delay. From Fig. 5.18 (a) it can be seen that RR can support not-L4S traffic better than a scheduler with priority, since it is able to keep a low delay also at high load. This is a confirm that RR is the scheduler that should be used when no congestion signal as L4S are present. This because the traffic that is not prioritized can incur in high delay and possibly retransmission timeout.

5.3.7 SINR

Signal to Interference Plus Noise Ratio (SINR) is an indicator of the quality of the channel on which it is transmitting. It is computed as explained in 3.3. Figure 5.19 shows the SINR trend on the Physical Downlink Shared Channel.



(a) (b)

Figure 5.19: SINR.

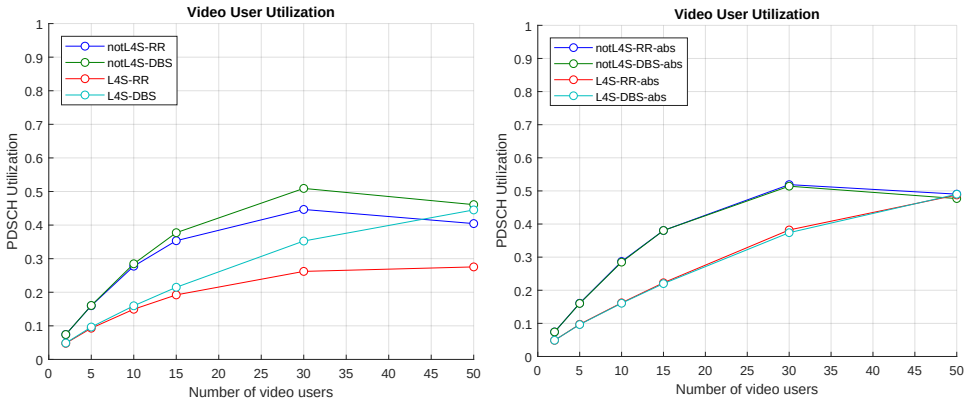
In both Fig. 5.19 (a) and (b) as expected there is a reduction in the SINR when the load in the system increase, since more users in the system means more interference. Looking at Fig. 5.19 (b), there are two distinct behavior when L4S is enabled in the system and when not. Without L4S support, the SINR is lower when there is lower system load, but at high load L4S has the lower SINR. The reason for this behavior lies in the fact that at higher load the use of a scheduler with priority and without L4S support causes instability and possible retransmission timeout for the background traffic as highlighted in 5.3.6.

The reason for higher SINR at low system load when there is L4S support in the system is a consequence of the fact that L4S implies a reduction of the transmitted rate. Therefore, for some TTIs there may be few packets to transmit, and thus less interference. In the case of not-L4S instead, the queue at the PDCP with high probability contains packets to be sent, and therefore the increase in the number of transmissions causes the consequent lower SINR.

5.3.8 Resource Utilization

This subsection reports the results related to the use of physical transmission resources. In particular, since the work of this thesis is focused on the downlink aspect, only the utilization of the downlink channel PDSCH is reported. The resources utilization is computed as the ratio between the number of physical resources blocks used and the total physical resources blocks available in the system. The latter is a fixed value if the simulation time and the scenario are fixed, and it is computed as the multiplication of simulation time (in sec-

onds), number of transmission slot in a second, number of cells and number of physical resources blocks in a transmission slot. To completeness, results relating to the use of the uplink channel and downlink control channel are reported in Appendix C. Fig. 5.20 shows the utilization of the downlink channel PDSCH considering only the video gaming traffic. Fig. 5.21 reports the resource utilization related to web traffic. Finally, Fig. 5.22 represents the total utilization of the PDSCH channel, which is the sum of Fig. 5.20 and 5.21.

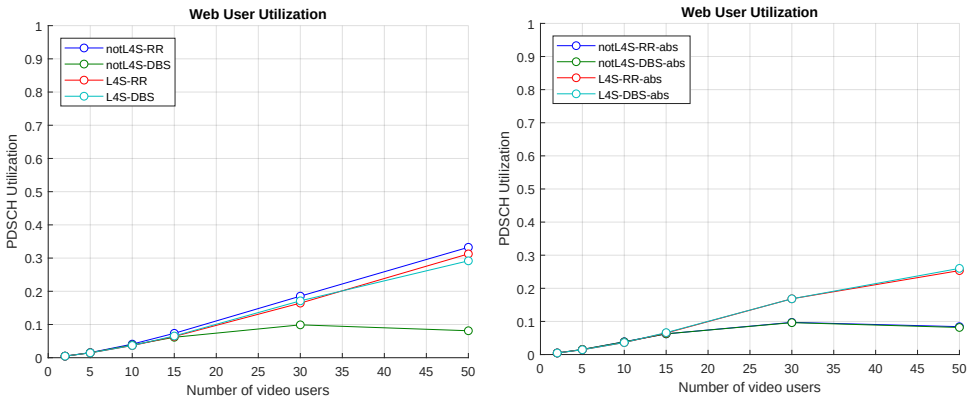


(a) (b)

Figure 5.20: PDSCH Video Utilization.

Fig. 5.20 highlights how the use of resources by video traffic is greater in the not-L4S case. As already explained, this is caused by the fact that L4S marking causes a reduction in the transmitted rate to the point that in some TTIs there are very few packets to be sent. However, the use of priority schedulers is beneficial as the use of resources in the L4S case increases.

In the RR case the video gaming and web traffic queues have the same priority, therefore the video gaming queues grow faster since the scheduler have more queues to schedule. When video gaming queues grow above the thresholds, L4S marks more packets to keep the queue delay low. This in turn causes a reduction in the transmitted rate. A lower transmitted rate implies fewer packets in the PDCP queue, and consequently more resources available for the web traffic. If instead, the video gaming queues have priority over the web, the congestion signals introduced by L4S are less since the number of queues among which the RR scheduler works is lower. Then, L4S marks less packets, thus the reduction in the transmitted rate is lower. Therefore, since the transmitted rate is higher, there are more packets in the queues that can be scheduled for a transmission, and this leads to an higher channel utilization.



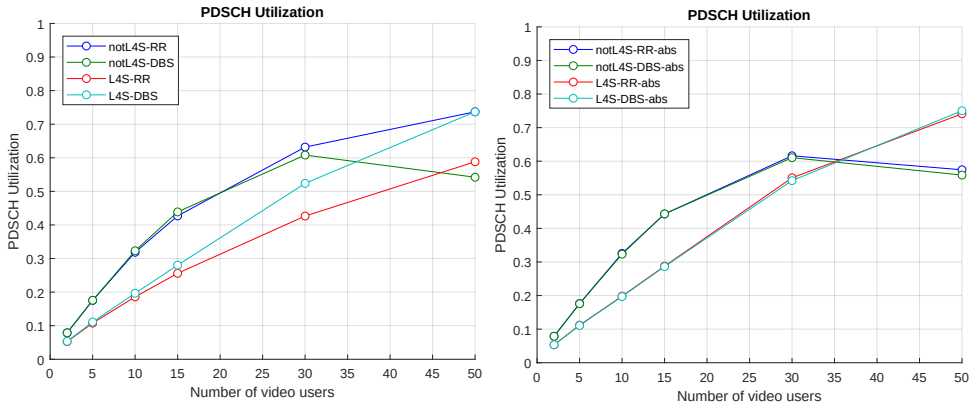
(a) (b)

Figure 5.21: PDSCH Web Utilization.

Fig. 5.21 confirms the explanation provided so far. In particular, it is evident that the web traffic has lower resources utilization when a scheduler with priority is used to support not-L4S capable traffic. This behavior occurs because the queues related to web traffic have a lower priority, and if the video gaming queues have packets to be sent, these are sent at the expense of web traffic. In any case, the web traffic is not completely starved thanks to scheduler mechanisms that are used to prevent that traffic with lower priority is never served.

The decreasing web user utilization at high load levels when priority scheduling is used without L4S, reveals that web traffic may experience severe issues with e.g. retransmission timeout. This is a further indication that priority scheduling requires the use of explicit congestion signals such as L4S marking.

Looking at the web utilization, when there is L4S support for the video gaming traffic, the resources utilization is higher, regardless the scheduler used. The motivation is that L4S, by marking the packets and decreasing the transmitted rate, leaves room for traffic with lower priority.



(a) (b)

Figure 5.22: PDSCH Utilization.

Fig. 5.22 represents the total use of the PDSCH channel, that is the sum of Fig. 5.20 and 5.21. In general, L4S implies less use of physical resources, except for the case with 50 video gaming users in the system. This is highlighted clearly in Fig. 5.22 (b), where one can see that the not-L4S lines cross the L4S lines following a convex trend. There are a few reasons behind this behavior. One is that higher priority for a given service can potentially give a situation where a large part of (or all) the resources are allocated to that users. The effect is that lower priority web traffic can be starved out with additional effects like TCP retransmission timeout. The outcome of these effects is that a lower resource utilization can indeed occur at higher loads and is a clear indication that the use of prioritized scheduling for a service must be accompanied with an efficient congestion indicator such as L4S to avoid unwanted side effects.

Chapter 6

Discussion

From the results reported in the previous chapter, it is clear that there is a difference in performance between the case where video traffic is supported by L4S compared to the case without it. Although it has to deal with the throughput-latency trade-off, the benefits introduced by L4S are evident. In this chapter all the benefits introduced are analyzed. Initially, in Section 6.1 an analysis to compare the performance obtained with the different schedulers and priorities is given. Section 6.2 discusses the throughput-latency trade-off, focusing on the performance related to users with very low throughput. Section 6.3 presents the results when the AQM is enabled in the queues, and a comparison with the results obtained without AQM is provided. Finally, Section 6.4 discusses the behavior that can be obtained by using a rate adaptive video frame encoder.

6.1 Scheduler comparison

Looking at the results reported in Chapter 5, one can notice at a glance that when using L4S the delay performance are always lower than without L4S (except for the RTP delay using RR). It is clear, however, that L4S brings benefits, and thus, it is preferable to have L4S support if the goal is reduce the latency.

This section then analyzes the performances changing the scheduler and the priority for the video gaming traffic, when the system supports L4S. All the figures in this section report also the not-L4S performance with default RR implementation as a baseline for a comparison. It is reported only the not-L4S wit RR because, as explained in Section 5.3.6 and 5.3.8, it is not suggested to support the video gaming traffic with scheduler that assign priority if a mech-

anism to signal congestion is not present in the network. Also here, the results refer to the scenario explained in Section 4.3 and the AQM is not enabled in the system.

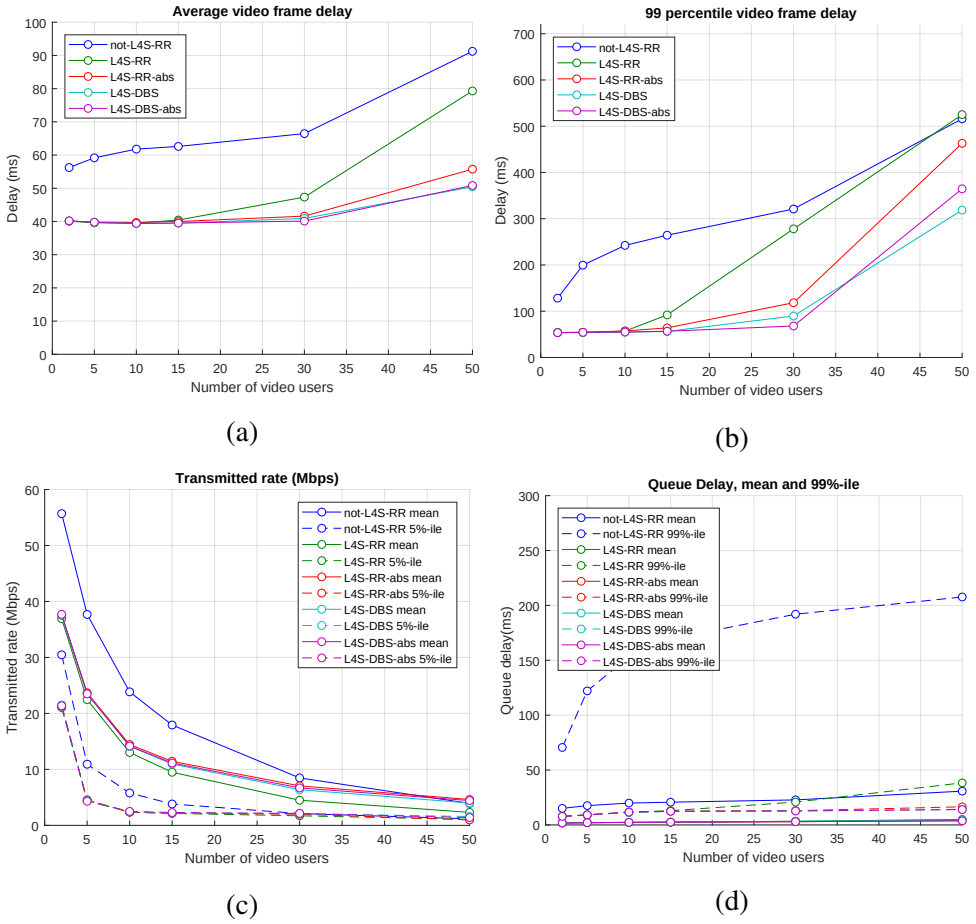


Figure 6.1: Scheduler comparison.

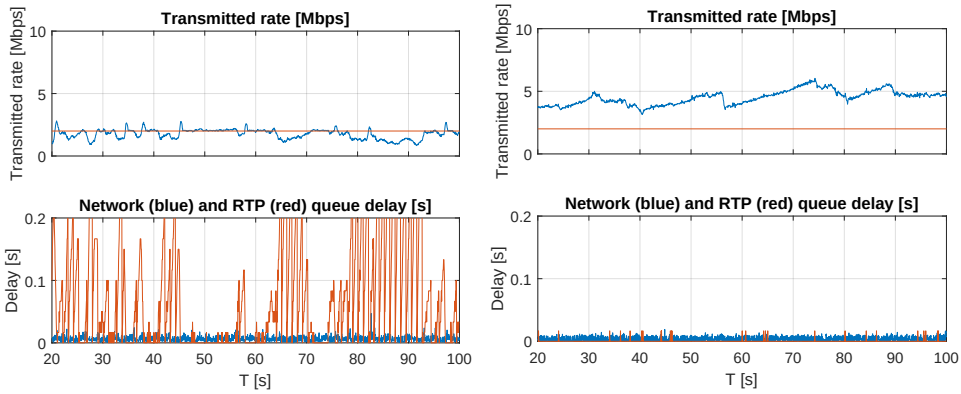
Fig 6.1 shows that using schedulers with priority gives better performance in all statistics: higher transmitted rate and lower delay. Round Robin without the use of priority appears to be the scheduler that causes the highest video frame delay, lower transmitted rate and also the highest queue delay if looked at 99th percentile. This evidence lead us to the conclusion that, while L4S can be used with normal RR scheduling for e.g. mass market Over-The-Top (OTT) video streaming, additional QoS support will give improved performance for premium and industrial use cases.

Another important aspect is that the 3 schedulers that gives priority to video gaming traffic are more or less overlapping, except for the video frame

delay with a load of 15 or more users. In fact, with 30 users in the system, the 99th percentile video frame Delay obtained with RR and absolute priority starts to be higher if compared to DBS, and with 50 video gaming users it approaches the performance of RR without priority. The explanation for the behavior of RR with absolute priority lies in the fact that some users are subjected to poor throughput. Looking at Fig. 6.1 (c) the case of RR with absolute priority shows the lowest 5th percentile. The relation between low transmitted rate and the increase in the video frame delay is presented in the next section.

6.2 Low transmitted rate, high delay

We have just analyzed how the performance degrades both in terms of transmitted rate and delay at high load in the system. The reason for that degradation is that at high load many users have throughput performance lower than 2 Mbps, the minimum video encoder output bitrate. When the throughput falls below this threshold, the RTP queue delay increases considerably, while the network queue delay is maintained at a low level. Fig. 6.2 shows time traces of one user subject to very low throughput and one with a throughput greater than 2 Mbps. As one can see from the corresponding delay, the correlation between low throughput and high delay is evident. Anyhow, in both cases the network queue delay is kept at a low level: this indicates that the L4S based network congestion control operates as expected.



(a) User with transmitted rate below 2 Mbps

(b) User with transmitted rate above 2 Mbps

Figure 6.2: Time-traces of tramsmitted rate and video frame delay.

The motivation for this behavior can be found in the sender side applica-

tion: the video encoder has a minimum bitrate set at 2 Mbps. This means that in any case, the encoder will produce at least 2 Mbps traffic for each user, but some users, perhaps in poor channel conditions, get less than 2 Mbps link throughput. This implies that many RTP packets are queued at the sender side, considerably increasing the RTP delay, and consequently also the video frame delay.

Unfortunately, there are no trivial solutions to this problem, and probably what can happen in the real-life is that the video user closes the application since the video quality is not decent. One option can be the use a Proportional Fair (PF) scheduler with minimum bitrate setting for the video gaming users but this option has not been tried in this work. Another possible solution is to allow the video encoder to lower the minimum video bitrate by lowering the video resolution, that will result however in a lower video gaming quality. In any case, these are implementation choices that can change from application to application, and it is difficult to establish which is the most effective solution.

To confirm this correlation, 99^{th} percentile of the video frame delay is considered in Fig. 6.3, analyzing also the case where the users with poor transmitted rate are not considered in the statistics. The four plots represents the 99^{th} percentile of all users (a), 99^{th} percentile not considering the 5% of users with worst transmitted rate (b) and finally the 99^{th} percentile not considering all the users with transmitted rate below 2 Mbps (c). Fig. 6.3 (d) shows the number of users that have transmitted rate below 2 Mbps, and thus that are not considered in the statistics of Fig. 6.3 (c). From Fig. 6.3 (d), with 50 video gaming users in the system and the RR scheduler, almost half of the users were below 2 Mbps. An OTT application, which by nature is designed to do its best with the given network, can actually work well if it e.g., reduces the gaming resolution further to work with bitrates even lower than 2 Mbps.

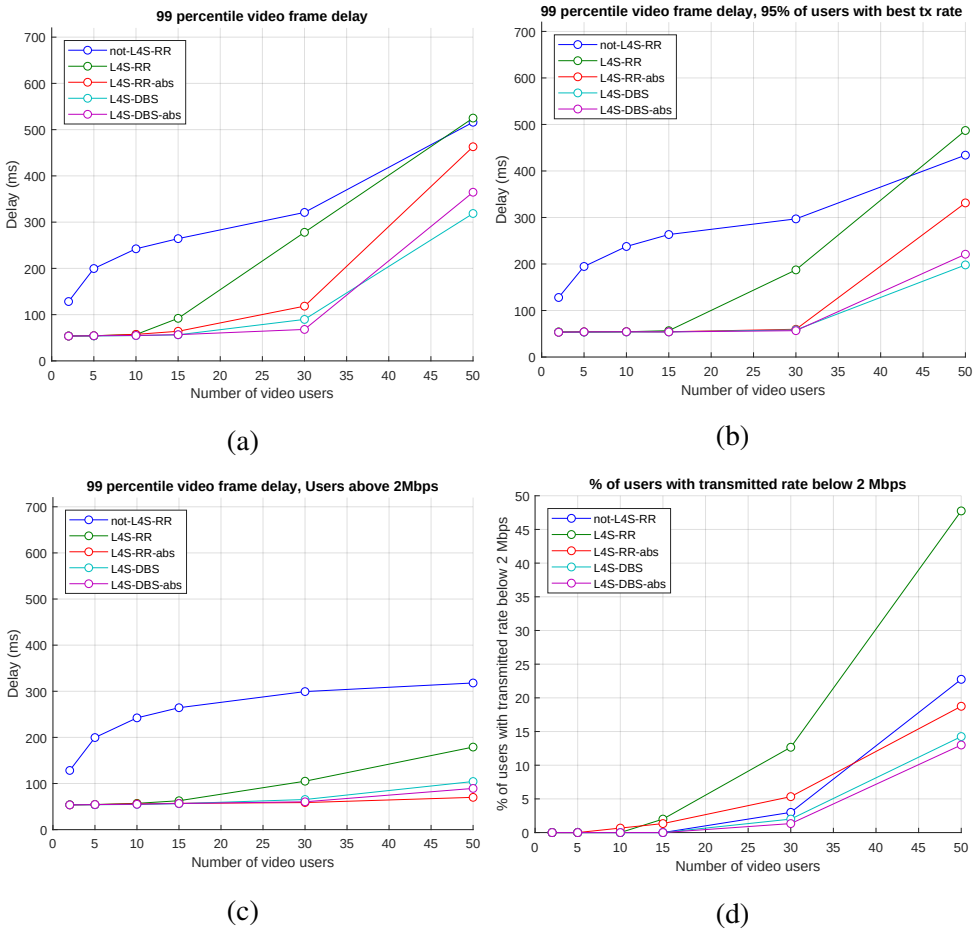


Figure 6.3: Comparison of video frame delay, considering the transmitted rate performance.

Fig. 6.3 (b) and (c) show that when the users are served with a good throughput (above 2 Mbps) the video frame delay performance are considerably improved, even at high load in the system.

6.3 Enabling AQM

All the results presented so far are obtained with queues that have unlimited capacity and no AQM techniques is used, except for the L4S marking strategy. This means that congestion never happen and packets are never lost in the network.

This section reports the results when AQM is enabled in the queues, to mimic a real case scenario where buffers implements AQM to avoid conges-

tion. The scenario used is described in Section 4.3, and it is the same used for the results presented in Section 5.3 with the only difference that AQM is enabled. The AQM policy consist in dropping only the oldest packet from the queue if the queue delay is higher than 100 ms, and eventually all the oldest packets are discarded from the queue while the delay is higher than 500 ms. It has to be specified that when a packet loss occur, the packet contributes only in the packet loss statistic and in the network queue delay statistic (it is not considered in the video frame delay and in the transmitted rate since it does not reach the receiver).

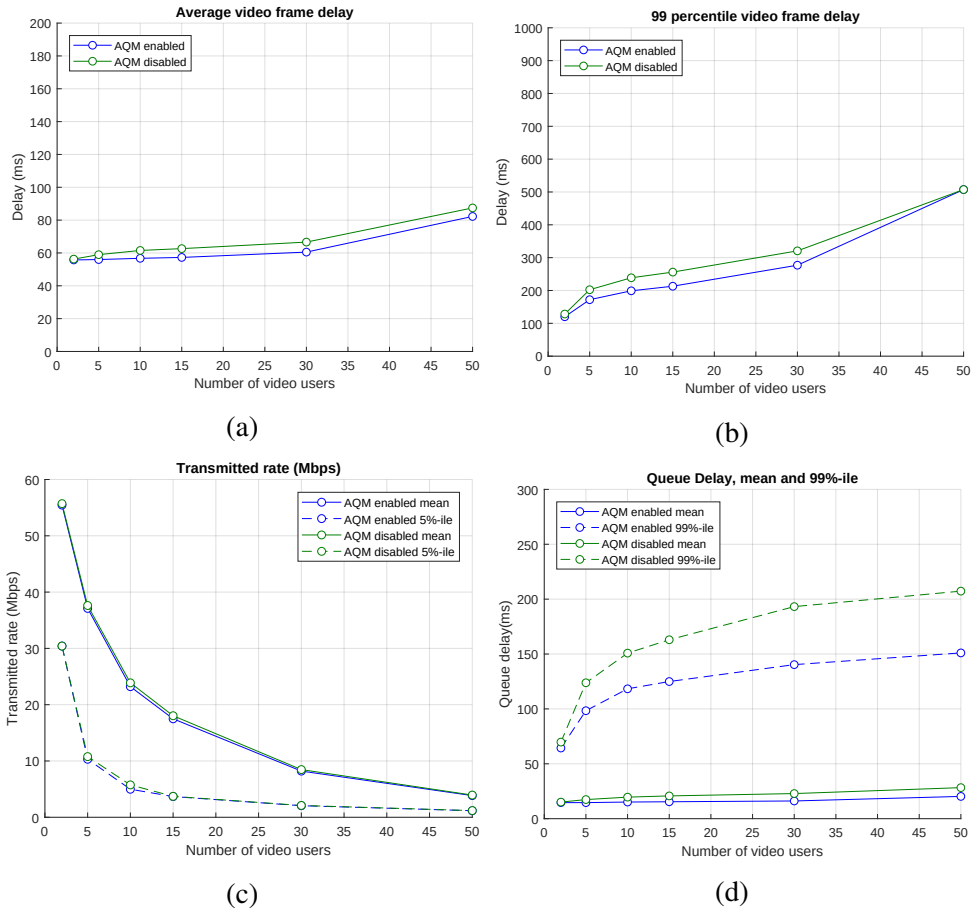


Figure 6.4: Comparison between the use of AQM or not, not-L4S with RR.

Fig. 6.4 refers to the not-L4S implementation with a RR scheduler, and shows a comparison between AQM disabled and AQM enabled scenarios. In particular, from Fig. 6.4 (b) and (d) one can see that by using the AQM the 99th percentile of video frame delay and the network queue delay are lower (so

improved) when the AQM is used. The lower delay happen at one cost: packet loss. Fig. 6.5 reports the packet loss rate for the analyzed scenario.

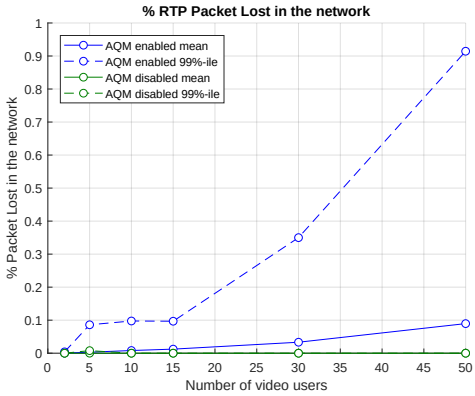


Figure 6.5: Packet loss rate, not-L4S with RR.

Fig. 6.5 shows that, as expected, AQM enabled means packet lost in the network. The 99^{th} percentile packet loss rate at high load in the system is almost the 0.9% of the transmitted packets.

Figure 6.6 instead shows the comparison between AQM disabled and AQM enabled scenarios when there is L4S support in the network and RR scheduler is used.

Even if RR is the worst scheduler to use in support of L4S traffic as explained in Chapter 5 and Section 6.1, Fig. 6.6 shows that the lines are always almost overlapping. This means that enabling the AQM in the scenario does not change the performance. The behavior was predictable and these results confirm the correct functioning of L4S. The thresholds of a normal AQM are much higher than the thresholds of L4S, therefore the queue at the PDCP layer, where L4S is implemented, rarely grow so much as to trigger the intervention of the AQM. The small discrepancies in the performance, where the lines are not overlapping, are due to packet loss. In Fig. 6.7 the packet loss rate for the L4S case is reported.

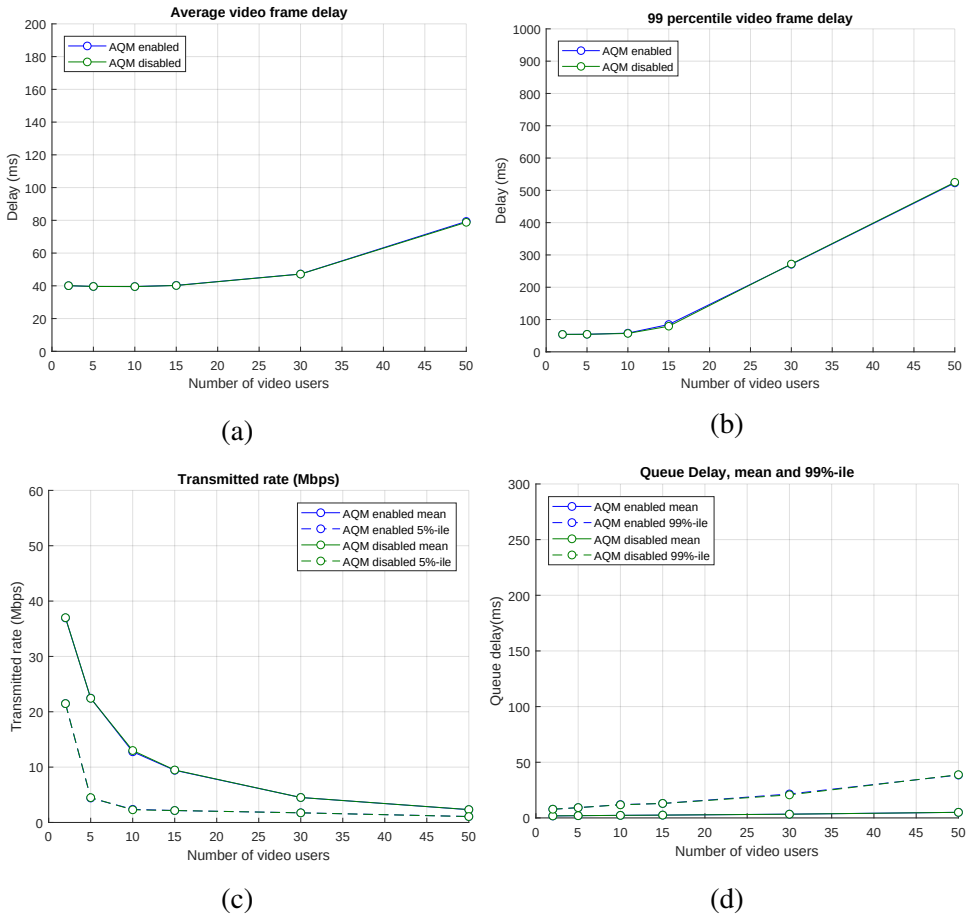


Figure 6.6: Comparison between the use of AQM or not, L4S with RR.

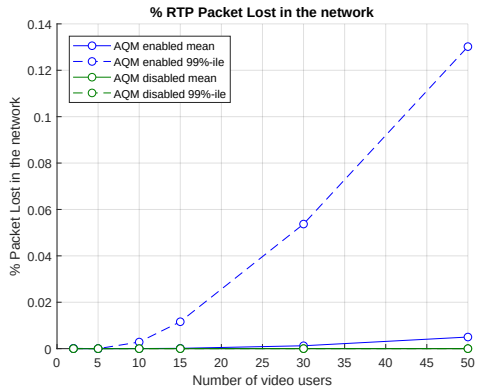


Figure 6.7: Packet loss rate, L4S with RR.

From Fig. 6.7 one can see that the average packet loss rate is lower than

the 0.01% and the 99th percentile is around 0.13% at high load in the system. If compared to Fig. 6.5 there the packet loss rate is 10x lower on average and 7x lower if looked at the 99th percentile.

To make a complete analysis, the performance obtained with AQM enabled changing the scheduler are reported in Fig. 6.8. In Fig. 6.8 the not-L4S implementation is reported only with RR support, and it is the baseline for the comparison. The results obtained without L4S support and scheduler with priority are not reported and considered in the analysis because of the justification provided in Section 5.3.8.

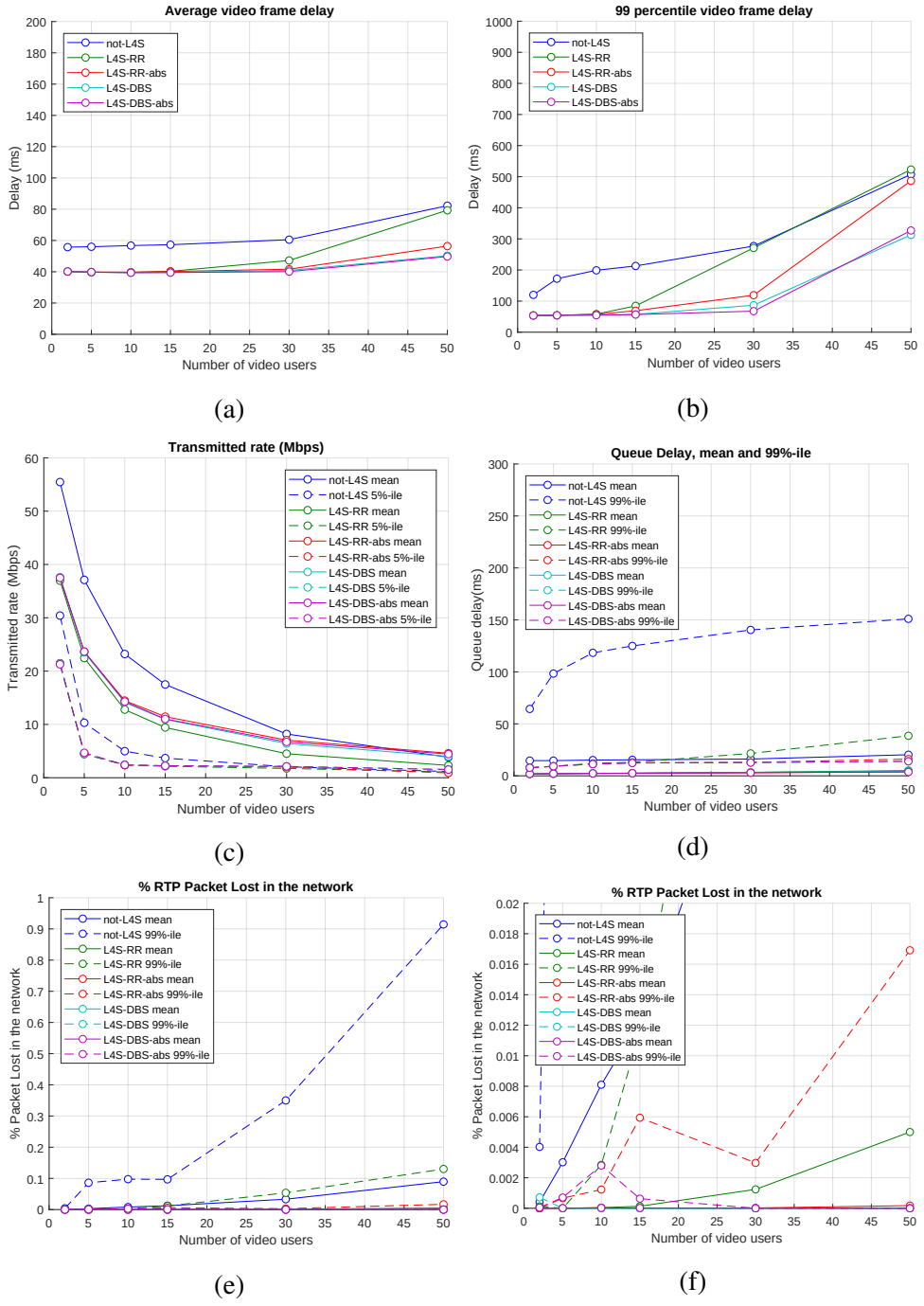
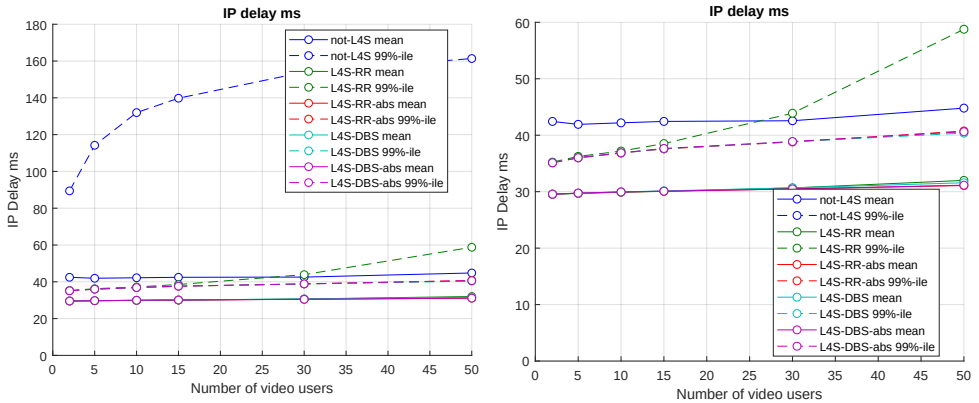


Figure 6.8: Comparison between L4S and not-L4S when AQM is enabled.

From Fig. 6.8 (a) and (b) it is evident that using a DBS leads to a lower video frame delay, both on average and also at the 99th percentile. Looking

at the transmitted rate in (c), the DBS is the scheduler that provides highest transmitted rate among the L4S results, that in any case at low load is lower than the not-L4S transmitted rate because of the throughput-latency trade-off. Fig. 6.8 (e) and (f) both represent the packet loss rate, but Fig. 6.8 (f) focuses on L4S and scheduler with priority. Note that the scales of Fig. 6.8 (e) and (f) are different. From Fig. 6.8 (f), the worst 99th percentile packet loss rate using the DBS with absolute priority is 0.003%, that means an improvement of 43x if compared with the case L4S and RR, and an improvement of 300x if compared with not-L4S and RR.

Finally, Fig. 6.9 reports the the IP packet delay, which is the delay calculated as the time between from when IP packet is transmitted by the server until it is received by the end user. This delay differs from the video frame delay, because the IP delay does not consider the RTP queue delay at sender side.



(a) (b)

Figure 6.9: IP packet delay.

From Fig. 6.9 (a) it can be seen that the delay experienced by a single packet is significantly lower if L4S is used, both in terms of average and also 99th percentile. Furthermore, by looking at Fig. 6.9 (b) that wants to highlight the comparison between the L4S scenarios, it is confirmed that RR is the scheduler that provides the higher IP delay.

One important consideration from Fig. 6.9 is that the results refer to layer 3 (IP) latency. Thus, regardless of the application, every type of IP traffic could benefit from L4S support. In other words, this results confirm that L4S is a general technology that can be implemented and used in different context and use-cases, to achieve lower latency.

In conclusion, from the results reported in this section it is clear how L4S improves the performances especially when supported by a DBS. Indeed, lower video frame delay and lower network queue delay can be reached, the probability to have a packet loss is drastically reduced, and at the same time the transmitted rate is not affected significantly.

6.4 The graceful degradation

When working with real-time latency-critical video, in the server-side application it is possible to choose whether to implement a mechanism where the bitrate of the video encoder is fixed, or in alternative to use a variable bitrate based on the congestion signals received from the network.

In this section the results obtained both with variable bitrate and with fixed bitrate are presented, to compare and understand the possible advantages of the former and the latter solution. In particular, the results obtained are shown in Fig. 6.10, where the average video frame delay and 99^{th} percentile, the transmitted rate and the Network Queue Delay are reported. Variable bitrate means that the video encoder is capable of adapting the video bitrate (in this work between 2 Mbps and 70 Mbps), while in the case of Fixed bitrate the values 2 Mbps, 5 Mbps, 10 Mbps, 20 Mbps and 40 Mbps are used. The results are obtained using the scenario explained in Section 4.3, using L4S and DBS, without enabling the AQM in the system.

In Fig. 6.10 the results related to the 20 Mbps and 40 Mbps cases are omitted to improve the readability.

We can deduce from these graphs that by using a variable bitrate, is observed a graceful degradation, i.e. the performance gradually deteriorate with the increase in the number of users in the system. On the other hand, with fixed bitrate the delay is excellent and lower than the variable bitrate case up to a certain load point that can be denoted as "breaking point". Beyond the breaking point, the performance become very bad. This "hard breaking point" behavior is generally unwanted when the application is used in networks where the load can vary with time, which is the to be expected in cellular networks.

The performance of video frame delay below the breaking point is lower in the fixed bitrate case because all users in the system are served with a higher transmission rate than the rate produced by the video encoder, therefore no packets will be delayed in the queues. In the case of adaptive bitrate instead, the video encoder must adapt the bitrate to exploit the network in the best way, but this happens at the cost of delaying some packets, which will then be marked to indicate that the rate should be lowered.

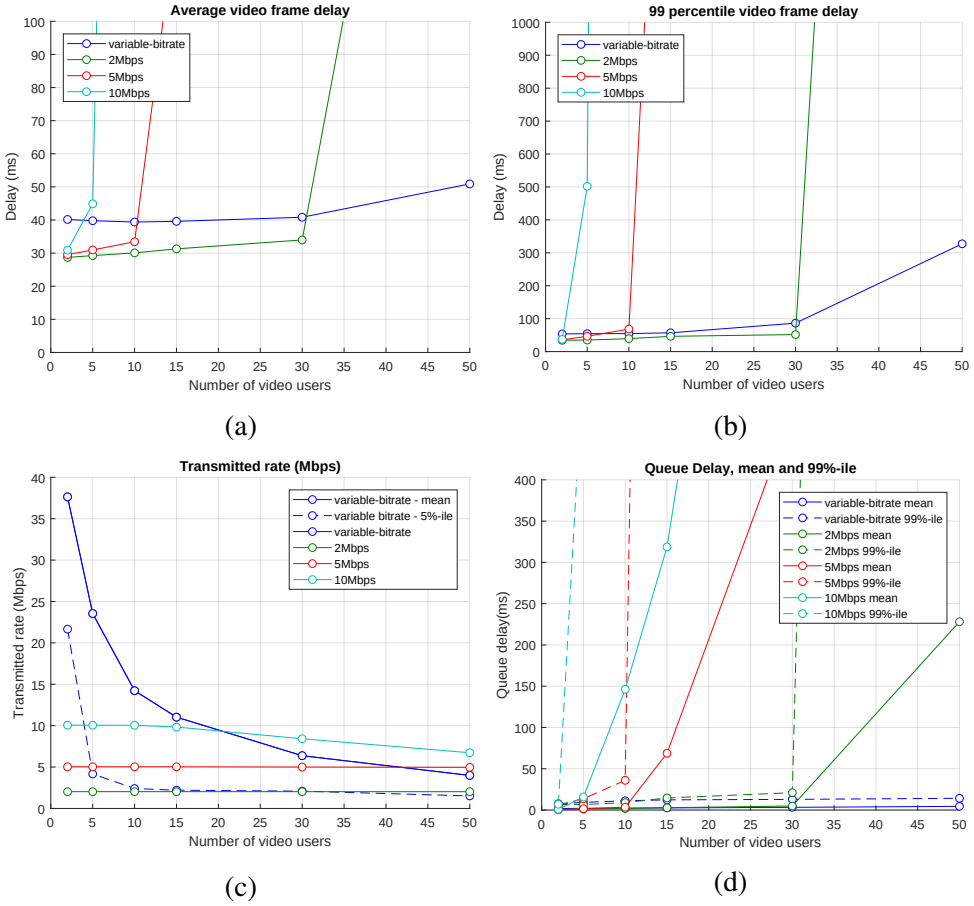


Figure 6.10: Comparison between Fixed bitrate and Variable bitrate.

Although the video frame delay with fixed bitrate is very low before the breaking point, one needs to reason about when this breaking point is. One see that with fixed bitrate at 10 Mbps, the breaking point is less than 5 users, with 5 Mbps it is 10 users, while at 2 Mbps the breaking point is 30 users. With adaptive bitrate also with 50 users the performance is decent, except for few users.

However, the most interesting results for this thesis is Fig. 6.10 (d). Although L4S was used as a strategy to reduce queue delay, the queue delay increases significantly after a certain system load if fixed bitrate is used. When the fixed bitrate is run, the congestion control is bypassed to provide always the same bitrate. This works well until there are no bottlenecks in the network. The bottlenecks appear at the system load we called “breaking point”, thus after that point, packets starts to be queued in the bottleneck node. Since the congestion control is bypassed and the sending rate is not reduced, packets

continues to be queued up leading to the behavior seen in Fig. 6.10 (d). Only by using the adaptive bitrate it is possible to keep the queue delay low even at high user loads.

In these simulations AQM has not been used, therefore in the real case where there can be packet dropped by the AQM, using fixed bitrate the performance could even be worse, while adaptive bitrate the use of AQM does not impact the performances as seen from results in Section 6.3.

Chapter 7

Conclusions

7.1 Conclusions

In this thesis work the L4S features have been analyzed in a cellular network context, to support the traffic generated by high-rate latency-critical applications. By reviewing the literature and analyzing the IETF and 3GPP standards, it is possible to conclude that tunnels such as IPSec and GTP-U are not a problem for ECN signals propagation. Therefore, using network nodes compliant with the specifications provided by the standardization bodies, it is possible to deploy an end-to-end L4S capable 5G network.

In a base station, at the RLC layer packets are encrypted, thus it is impossible to implement L4S marking. For this reason, the state of the art network simulator used in this work has been modified implementing the L4S features at the PDCP layer, where it is possible to change the ECN bits to signal a congestion experienced.

It has been observed that changing the thresholds related to L4S marking implies a change in the throughput and latency performance. Enlarging the threshold span implies increasing the throughput but also the delay, vice versa reducing the threshold span involves lower throughput and lower delay.

The final and most important result has been obtained by comparing the performance achieved with and without L4S. Simulations with L4S always shown lower video frame delay and lower queue delay while providing good throughput. The benefits introduced by L4S are further highlighted when AQM is enabled in the queue at the PDCP layer, since thanks to L4S the packet loss rate is considerably decreased.

In conclusion, the results of this thesis work confirm that the use of L4S, specially if combined to a priority scheduler is beneficial to support the traffic

of latency-critical applications. Moreover, since the layer 3 latency is lowered thanks to L4S, all the applications and use-cases where the traffic is carried by IP could experience benefits by adopting L4S technology.

7.2 Future Work

This work focused on downlink video, with SCReAM for congestion control, and 9 MHz LTE physical transmission bandwidth. Given the encouraging results obtained in this study, it would be interesting to continue the analysis by increasing the transmission bandwidth and switching to full NR stack. It is expected that a larger system bandwidth can improve the performance with L4S further as the impact from individual background flows becomes smaller due to the larger amount of resource blocks that can be scheduled. Also, could be useful testing L4S with other types of traffic supported by different congestion control algorithms. Furthermore, it would be interesting to see the behavior of L4S in support of different types of traffic simultaneously, e.g. video gaming and remote control together in the same cell. Another future work could consist in the implementation of an L4S marking strategy based not only on the level of the queue, but also on the physical resources available. In this way it could be possible to obtain a further improvement in performance.

Moreover, SCReAM was not originally designed with L4S in mind, rather it is an algorithm that gives decent performance also without L4S support. One concern is that the L4S implementation in SCReAM may be too cautious and backs off unnecessarily in response to congestion. Future research is likely to give algorithms that can better exploit the potential of L4S for low latency video.

As last point, in this thesis work L4S was implemented only in the gNB, since it is usually the bottleneck of the system, but to have a full L4S support, the L4S features should be implemented in every node of the network. In this way, with a full 5G network L4S capable it will be possible to asses exhaustively the performance and benefits.

7.3 Reflections

L4S is a technology proposed with the aim of decreasing the queue delay and consequently the end-to-end delay. The second L4S objective is to drastically reduce the probability of having a packet loss on the network. These characteristics can have an impact also from a sustainability perspective since they can

contribute to better management of resources and a consequent energy saving. Every bit or packet transmitted through a communication network has a cost in terms of energy, and if a packet is lost during the path, the energy required to transmit the packet is wasted. Moreover, if a retransmission is needed after a loss, additional energy is required to send again the packet that was lost. It is clear that if thanks to L4S the probability of a packet loss is reduced, and from the results the packet loss rate with L4S is very close to zero, energy waste is avoided.

Furthermore, given that through L4S the buffers will be used to a lesser extent, it will be possible to consider reducing the size of the buffers in the base stations and other network nodes. This would lead to a reduction in hardware production cost and at the same time a decrease in the carbon footprint related to the production of each network device.

Bibliography

[1] T. Henderson, S. Floyd, A. Gurtov, and Y. Nishida, "The NewReno Modification to TCP's Fast Recovery Algorithm", IETF, Tech. Rep. RFC 3782, 2004. [Online]. Available: https://tools.ietf.org/html/rfc3782.

[2] S. Ha, I. Rhee, and L. Xu, "Cubic: A new tcp-friendly high-speed tcp variant", *Operating Systems Review*, vol. 42, pp. 64–74, Jul. 2008.

[3] B. Braden, D. Clark, J. Crowcroft, B. Davie, and S. Deering, "Recommendations on Queue Management and Congestion Avoidance in the Internet", IETF, Tech. Rep. RFC 2309, 1998. [Online]. Available: https://tools.ietf.org/html/rfc2309.

[4] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", IETF, Tech. Rep. RFC 3168, 2001. [Online]. Available: https://tools.ietf.org/html/rfc3168.

[5] B. Briscoe, K. De Schepper, M. Bagnulo Braun, and G. White, "Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture", IETF, Tech. Rep. Draft ietf-tsvwg-l4s-arch-06, 2019. [Online]. Available: https://tools.ietf.org/html/draft-ietf-tsvwg-l4s-arch-06.

[6] T. Cagenius, A. Ryde, J. Vikberg, and P. Willars, "Simplifying the 5G ecosystem by reducing architecture options", 2018. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/simplifying-the-5g-ecosystem-by-reducing-architecture-options.

[7] B. Briscoe, "Tunnelling of Explicit Congestion Notification", IETF, Tech. Rep. RFC 6040, 2010. [Online]. Available: https://tools.ietf.org/html/rfc6040.

[8] G. Fairhurst and M. Welzl, "The Benefits of Using Explicit Congestion Notification (ECN)", IETF, Tech. Rep. RFC 8087, 2017. [Online]. Available: https://tools.ietf.org/html/rfc8087.

[9] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance", *Networking, IEEE/ACM Transactions on*, vol. 1, pp. 397–413, Sep. 1993.

[10] K. Nichols and V. Jacobson, "Controlling queue delay", *Communications of The ACM - CACM*, vol. 55, pp. 42–50, May 2012.

[11] R. Pan, P. Natarajan, F. Baker, and G. White, "Proportional Integral Controller Enhanced (PIE): A Lightweight Control Scheme to Address the Bufferbloat Problem", IETF, Tech. Rep. RFC 8033, 2017. [Online]. Available: https://tools.ietf.org/html/rfc8033.

[12] F. Baker and G. Fairhurst, "IETF Recommendations Regarding Active Queue Management", IETF, Tech. Rep. RFC 7567, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7567.

[13] S. Bensley, D. Thaler, P. Balasubramanian, L. Eggert, and G. Judd, "Data Center TCP (DCTCP): TCP Congestion Control for Data Centers", IETF, Tech. Rep. RFC 8257, 2017. [Online]. Available: https://tools.ietf.org/html/rfc8257.

[14] B. Briscoe, K. De Schepper, O. Tilmans, M. Kuhlewind, J. Misund, O. Albisser, and S. A. A., "Implementing the 'Prague Requirements' for Low Latency Low Loss Scalable Throughput(L4S)", in *Proc. Netdev 0x13*, 2019. [Online]. Available: https://www.files.netdevconf.info/f/4d6939d5f1fb404fafd1/?dl=1.

[15] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control", *ACM Queue*, vol. 14, September-October, pp. 20–53, 2016. [Online]. Available: http://queue.acm.org/detail.cfm?id=3022184.

[16] I. Johansson and Z. Sarker, "Self-Clocked Rate Adaptation for Multimedia", IETF, Tech. Rep. RFC 8298, 2017. [Online]. Available: https://tools.ietf.org/html/rfc8298.

[17] B. Briscoe and K. De Schepper, "Resolving Tensions between Congestion Control Scaling Requirements", Simula, Technical Report TR-CS-2016-001, 2017. [Online]. Available: https://riteproject.files.wordpress.com/2015/10/ccdi_tr.pdf.

[18] M. Kühlewind, D. Wagner, J. Espinosa, and B. Briscoe, "Using data center tcp (dctcp) in the internet", Dec. 2014.

[19] K. De Schepper, B. Briscoe, and G. White, "DualQ Coupled AQMs for Low Latency, Low Loss and Scalable Throughput (L4S)", IETF, Tech. Rep. Draft ietf-tsvwg-aqm-dualq-coupled-10, 2019. [Online]. Available: https://tools.ietf.org/html/draft-ietf-tsvwg-aqm-dualq-coupled-10.

[20] E. Dahlman, S. Parkvall, and J. Sköld, *5G NR, The next generation wireless access technology*. Academic Press, 2018, pp. 74–96, ISBN: 978-0-12-814323-0.

[21] C. Et. al., "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", IETF, Tech. Rep. RFC 3095, 2001. [Online]. Available: https://tools.ietf.org/html/rfc3095.

[22] "5G, NG-RAN, NG general aspects and principles", 3GPP, Tech. Rep. 3GPP TS 38.410, 2018.

[23] "5G, NG-RAN, Xn general aspects and principles", 3GPP, Tech. Rep. 3GPP TS 38.420, 2018.

[24] "5G, NG-RAN, F1 general aspects and principles", 3GPP, Tech. Rep. 3GPP TS 38.470, 2018.

[25] "5G, NG-RAN, E1 general aspects and principles", 3GPP, Tech. Rep. 3GPP TS 38.460, 2019.

[26] S. Rommer, P. Hedman, M. Olsson, L. Frid, S. Sultana, and C. Mulligan, *5G CORE NETWORKS, Powering Digitalization*. Academic Press, 2018, pp. 74–96, ISBN: 978-0-12-814323-0.

[27] RITE. (). Reducing Internet Transport Latency, [Online]. Available: https://riteproject.eu/.

[28] K. De Schepper, O. Bondarenko, I. Tsang, and B. Briscoe, "'Data Center to the Home': Ultra-Low Latency for All", RITE Project, Technical report, 2015. [Online]. Available: http://riteproject.eu/publications/.

[29] NOKIA. (2020). Enjoying a real-time Internet, enabled by L4S, [Online]. Available: https://onestore.nokia.com/asset/207072.

[30] G. White, K. Sundaresan, and B. Briscoe. (2019). Low Latency DOCSIS: Technology Overview, [Online]. Available: https://www.cablelabs.com/technologies/low-latency-docsis.

[31] R. Pan, P. Natarajan, F. Baker, and G. White, "The FlowQueue-CoDel Packet Scheduler and Active Queue Management Algorithm", IETF, Tech. Rep. draft-ietf-aqm-fq-codel-06, 2016. [Online]. Available: https://tools.ietf.org/html/draft-ietf-aqm-fq-codel-06.

[32] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", IETF, Tech. Rep. RFC 2474, 1998. [Online]. Available: https://tools.ietf.org/html/rfc2474.

[33] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", IETF, Tech. Rep. RFC 3550, 2003. [Online]. Available: https://tools.ietf.org/html/rfc3550.

[34] B. Briscoe, "Propagating Explicit Congestion Notification Across IP Tunnel Headers Separated by a Shim", IETF, Tech. Rep. Work in progress, draft-ietf-tsvwg-rfc6040update-shim-10, 2020. [Online]. Available: https://tools.ietf.org/html/draft-ietf-tsvwg-rfc6040update-shim-10.

[35] M. Alizadeh, A. Javanmard, and B. Prabhakar, "Analysis of dctcp: Stability, convergence, and fairness", vol. 39, Jan. 2011, pp. 73–84.

[36] "5G; Study on channel model for frequencies from 0.5 to 100 GHz ", 3GPP, Tech. Rep. 3GPP TS 38.901, 2017.

[37] *How to Reduce Latency or Lag in Gaming*. [Online]. Available: https://www.actiontec.com/wifihelp/wifibooster/how-to-reduce-latency-or-lag-in-gaming-2/.

[38] C. Hoffman. (2020). Does Online Gaming Really Use a Ton of Bandwidth?, [Online]. Available: https://www.howtogeek.com/664369/does-online-gaming-really-use-a-ton-of-bandwidth/.

[39] W. Hu and G. Xiao, "Self-clocking principle for congestion control in the internet", *Automatica*, vol. 48, pp. 425–429, Feb. 2012.

[40] S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlewind, "Low Extra Delay Background Transport", IETF, Tech. Rep. RFC 6817, 2012. [Online]. Available: https://tools.ietf.org/html/rfc6817.

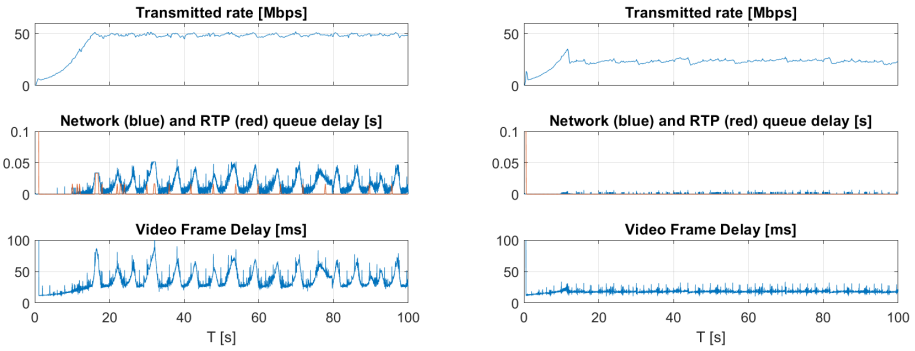
[41] T. Wiegand, G. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the h.264/avc video coding standard", *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, pp. 560–576, Aug. 2003.

Appendix A

Throughput analysis

A.1 Throughput-Latency Trade-off

The relation between throughput and latency is governed by an intrinsic trade-off. Higher throughput means higher latency, and vice versa, achieve low latency implies a reduction in throughput. The throughput-latency tradeoff is the explanation for throughput reduction when using L4S. L4S aims to achieve low latencies, but this entails a reduction in throughput. Fig. A.1 shows the time course of the throughput, queue delay and video frame delay values of a video gaming user. In this case, the scenario refers to a fixed communication network, where the capacity of the connection between the server and the client is 50 Mbps. **l4sHighTh** is set to 10 ms and **l4sLowTh** is set to 5 ms.



(a) Not-L4S (b) L4S

Figure A.1: Not-L4S vs L4S.

One can see how with L4S the throughput is reduced to about 25 Mbps

and at the same time, there is a clear reduction in queue delay and video frame delay.

One reason for the decrease in throughput is found in the challenging video model used. Fig. A.2 shows the change in frame size over time. It is possible to observe how every 2 seconds there is the creation of a new I-frame, that is the wider frame. To accommodate the I-frames, SCReAM with L4S give headroom to frame with large size. In this way, it is possible to maintain low delay, but at the same time involves the throughput reduction.

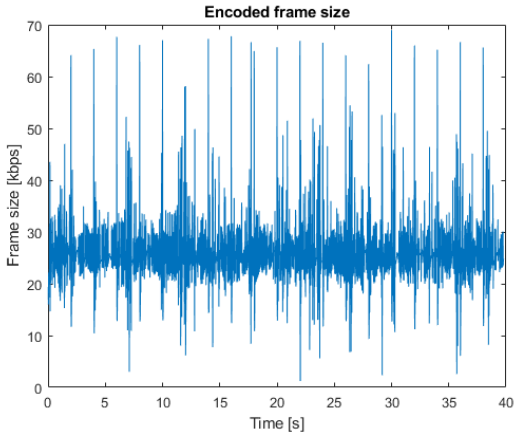
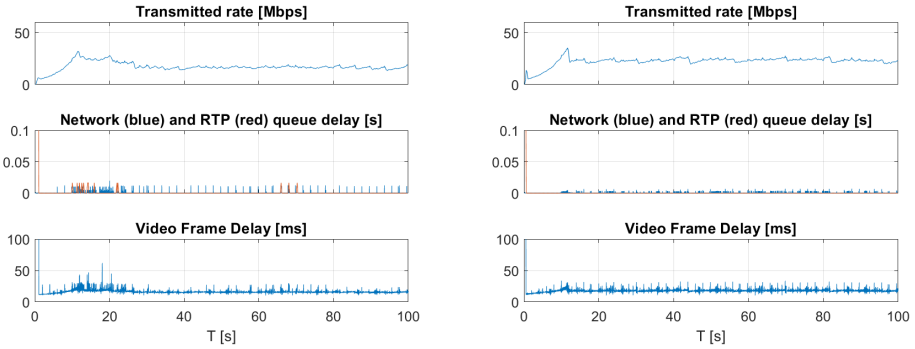


Figure A.2: Encoded video frame size.

However, to justify that L4S is a good solution to mitigate the trade-off, in Fig. A.3 the L4S case is compared to the not-L4S case where classic AQM is used to reach low latency. Queue delay and Video frame Delay are almost identical in both cases, but L4S highlights higher throughput. The parameters of the AQM have been tuned iteratively to reach queue delay level similar to L4S, namely, discard the oldest packet from the queue when the queue delay is above 14 ms, and discard all the oldest packets when the queue delay is above 20 ms.



(a) not-L4S with AQM (b) L4S

Figure A.3: Not-L4S with AQM vs L4S.

The case L4S with AQM is not reported because the results are identical to the L4S case without AQM. This because, even the AQM parameters are set to achieve lower queue delay, they are still higher than the L4S thresholds. Thus, L4S never triggers the AQM.

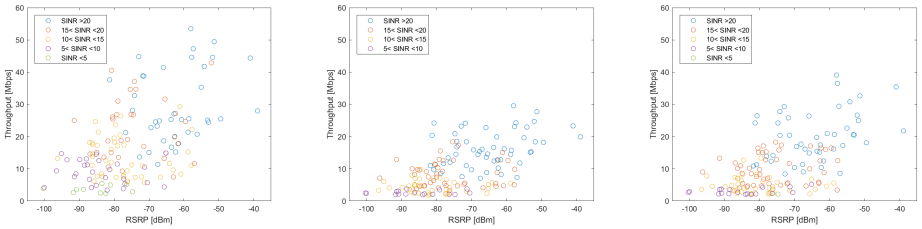
Since Fig. A.1 and A.3 are useful for a qualitative analysis, in Tab. A.1 are also reported the statistics for a quantitative analysis among the methods used. As one can see, with L4S the reduction in throughput is rewarded by a lower queue delay, RTP delay and Video Frame Delay. It is also noted that the trade-off obtainable with L4S is more worthwhile than the one obtained using a normal AQM: even if the latency metrics are the same for both cases, L4S provides higher throughput.

Metric	**not-L4S**	**not-L4S with AQM**	**L4S**
Average Transmitted rate [Mbps]	44.2	17.3	23.0
Average Network Queue Delay [ms]	15.4	0.49	0.52
99%-ile Network Queue Delay [ms]	49.7	10.5	3.79
Average RTP Queue Delay [ms]	4.72	3.89	0.85
Average Video Frame Delay [ms]	39.3	18.6	18.8
99%-ile Video Frame Delay [ms]	83.2	29.9	30.6

Table A.1: Comparison of all metrics.

A.2 Throughput-SINR-RSRP analysis

In a radio access network, the throughput of the end-user may be influenced by the distance between the user and the base station and also from interference caused by other transmissions. SINR and RSRP are two parameters that indicate the interference and power received from the user device respectively. Fig A.4 shows how throughput varies based on RSRP and SINR.

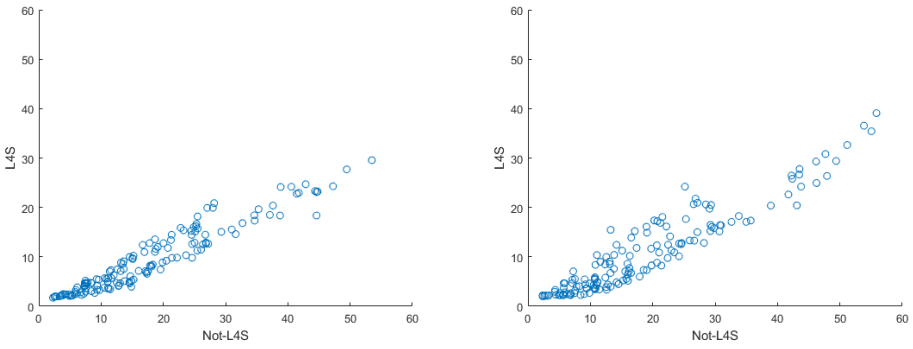


(a) not-L4S RR (b) L4S RR (c) L4S DBS

Figure A.4: Throughput - SINR - RSRP.

It can be seen that lower throughput values correspond to low RSRP, lower SINR or a combination of the two. However, there are cases in which the user, although in a good position and with acceptable interference, is still subject to low throughput. Looking at the not-L4S RR case one can note that the throughput is generally higher, but also, in this case, there are few users with good SINR and RSRP and lower throughput.

These results lead to affirm that the trade-off throughput latency is the main factor for the throughput reduction. Further confirmation can be observed in Fig. A.5, where each point of the graph represents the throughput of a user, in the L4S case and the Not-L4S case. It can be seen that the trend is linear and there is a proportionality between the x-axis (not-L4S throughput) and the y-axis (L4S throughput). It is also clear that the throughput with L4S is generally lower.



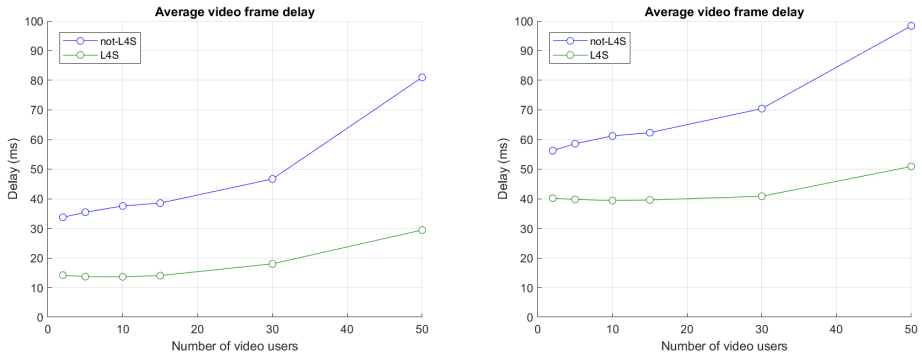
(a) RR (b) DBS

Figure A.5: Throughput comparison L4S vs not-L4S.

Appendix B

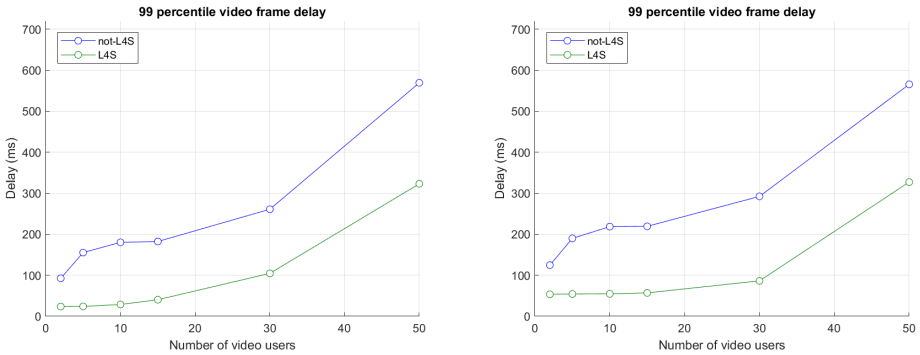
Lower RTT

Result presented in the report refer always to a case where the RTT between the server that provides the video content and the base radio station is fixed to 50 ms. This RTT is perhaps excessive for a real use case, where it can be assumed that the servers for content such as video gaming are spread throughout the territory, with RTT of few milliseconds. For this reason, other simulations were performed by fixing the RTT between the server and the base station at 5 ms. The results are reported here, obtained with DBS scheduler and 5-10 ms threshold span.



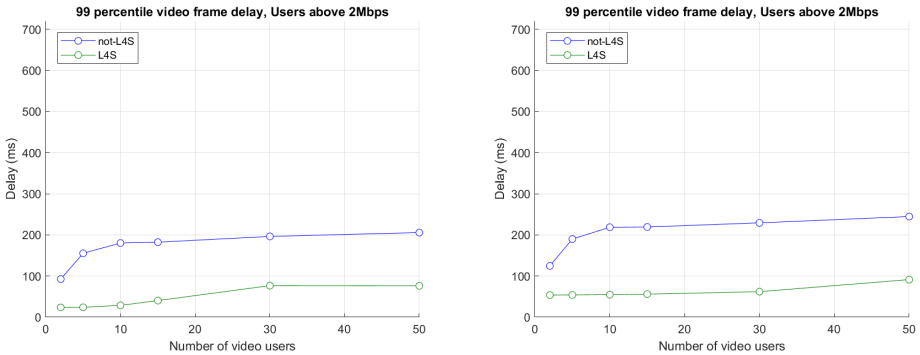
(a) 5 ms RTT (b) 50 ms RTT

Figure B.1: Video Frame Delay, average.



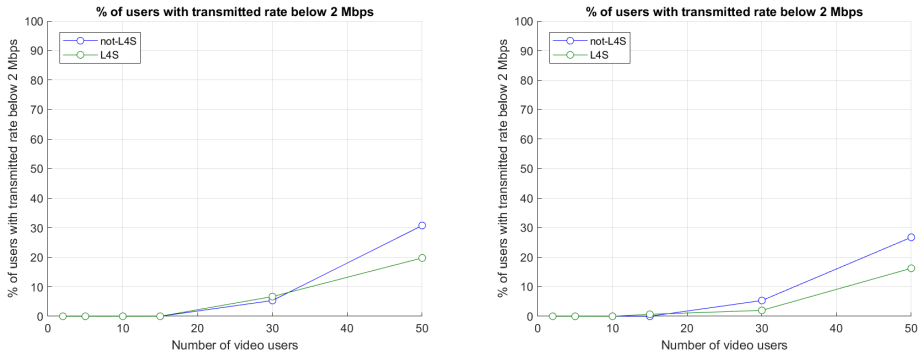
(a) 5 ms RTT (b) 50 ms RTT

Figure B.2: Video Frame Delay, 99th percentile.



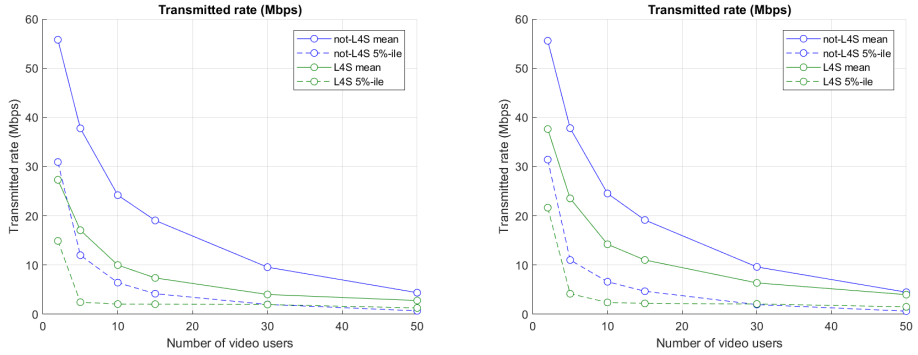
(a) 5 ms RTT (b) 50 ms RTT

Figure B.3: Video Frame Delay, 99th percentile, discarding users with throughput below 2 Mbps.



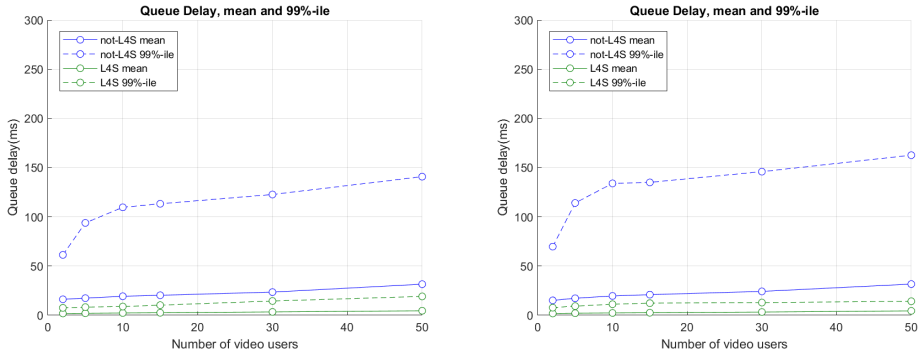
(a) 5 ms RTT (b) 50 ms RTT

Figure B.4: Percentage of users with throughput below 2 Mbps.



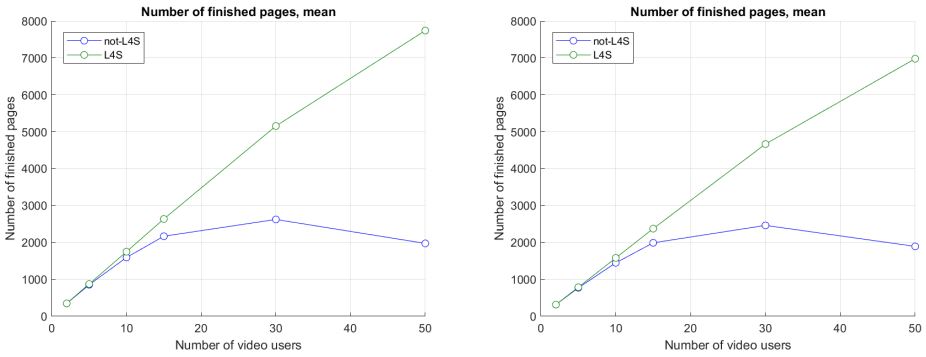
(a) 5 ms RTT (b) 50 ms RTT

Figure B.5: Transmitted rate.



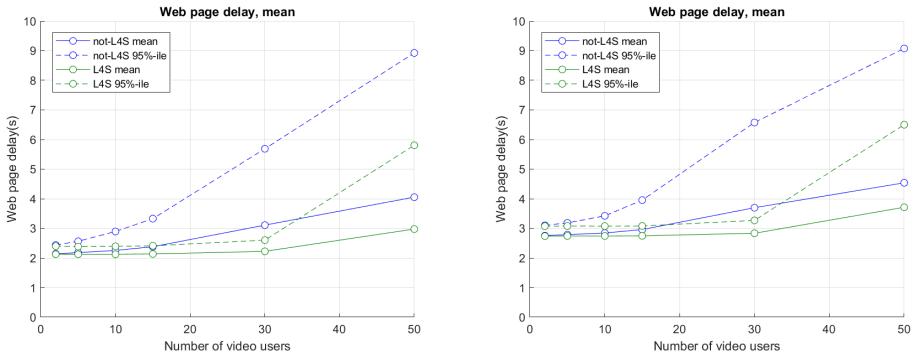
(a) 5 ms RTT (b) 50 ms RTT

Figure B.6: Network Queue delay.



(a) 5 ms RTT (b) 50 ms RTT

Figure B.7: Number of finished pages.



(a) 5 ms RTT (b) 50 ms RTT

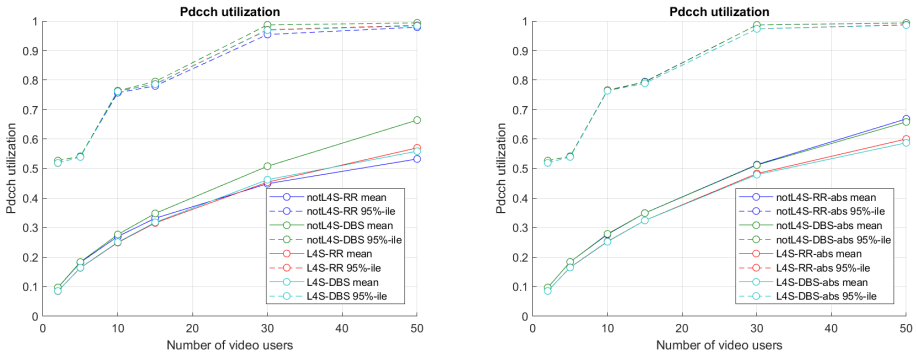
Figure B.8: Web page delay.

It can be observed that a lower RTT entails a reduction in the average Video Frame Delay, but also a decrease in the transmitted rate. The 99%-ile video frame delay shows no clear difference between the two cases, and the network queue delay is also very similar. One can notice a greater number of downloaded web pages and a lower web page delay when the RTT is 5 ms. This is also the explanation for a reduced throughput for video gaming users: if the download page time is low, the web traffic users will request web page faster, hammering the network more. Thus, the video gaming traffic is pushed back, with a decrease in the throughput. Another web traffic model in which the web traffic users are e.g. created according to a poisson arrival process would likely give a lower RTT dependency as a lower RTT would then not give more web page downloads.

Appendix C

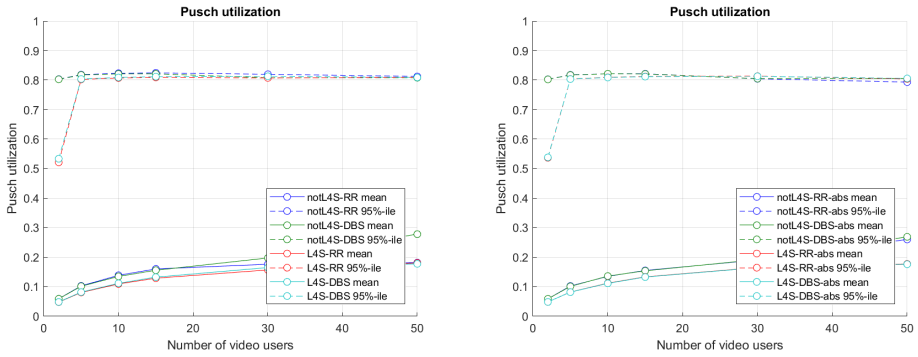
PDCCH and PUSCH Utilization

Here are reported the results related to the resource utilization for the downlink contorl channel (PDCCH) in Fig. C.1 , and the Uplink data channel (PUSCH) in Fig.C.2.



(a) (b)

Figure C.1: PDCCH Utilization.



(a) (b)

Figure C.2: PUSCH utilization.

TRITA-EECS-EX-2020:575