QoE-Based Scheduling Algorithms for Adaptive HTTP Video Delivery in LTE

Frederico Freire Rodrigues

Thesis to obtain the Master of Science Degree in

Electrotecnical and Computer Engineering

Supervisors

Prof. António José Castelo Branco Rodrigues

Prof. Maria Paula dos Santos Queluz Rodrigues

Dr. Ivo Luís de la Cerda Garcia e Sousa

Examination Committee

Chairperson: Prof. José Eduardo Charters Ribeiro da Cunha Sanguino

Supervisor: Prof. António José Castelo Branco Rodrigues

Member of the Committee: Prof. Pedro Manuel de Almeida Carvalho Vieira

November 2017

Acknowledgements

I would like to thank my supervisors, Prof. Antonio Rodrigues, Prof. Maria Paula Rodrigues and Dr. Ivo Sousa for the invaluable guidance, patience and support given throughout the development of this work.

A heartfelt thank you also goes to my family and friends, without whom none of this would have been possible.

Resumo

Nos últimos anos, o consumo de conteúdo multimédia aumentou exponencialmente, em particular o de vídeo. Hoje em dia, os operadores móveis têm dois desafios. Por um lado, necessitam de satisfazer as altas expectativas que os utilizadores têm nos serviços utilizados. Por outro lado, a capacidade da rede móvel não consegue ser aumentada tão rapidamente como a necessidade o exige. Uma solução possível passa pelo desenvolvimento de estratégias de alocação de recursos inteligentes que aloquem os recursos da rede de forma eficiente e que satisfaçam o maior número de clientes, proporcionando-lhes uma elevada qualidade de experiência (QoE).

Motivado pelos algoritmos de alocação de recursos que se podem encontrar na literatura, nesta tese apresenta-se o seu estado da arte, analisando as debilidades dos mesmos, bem como se propõe uma nova e eficaz solução, que pode ser usada numa rede *Long Term Evolution* (LTE). É proposto um algoritmo de alocação de recursos, *Maximum Buffer Filling* (MBF), que permite aumentar o número de utilizadores móveis satisfeitos que fazem *streaming* de vídeo. Para isso, o MBF faz uso do estado do buffer dos utilizadores, que se trata de uma métrica de QoE reportada pelos clientes que implementam a especificação *Dynamic Adaptive Streaming over HTTP* (DASH), estandardizada pela MPEG e também conhecida por MPEG-DASH. Considera também o estado do canal de rádio que os utilizadores reportam. O MBF aloca recursos de acordo com o nível do buffer dos utilizadores e com a quantidade de segundos que conseguirão fazer download. O algoritmo pode operar em dois modos (o modo 1 e o modo 2), de acordo com a objetivo do operador: aumentar o número de utilizadores satisfeitos ou minimizar o número de utilizadores insatisfeitos. Esta tese compara a capacidade suportada pelo MBF com as capacidades dos algoritmos de alocação *Round Robin* (RR), *Blind Equal Throughput* (BET), *Proportional Fair* (PF) e *Proportional Fair with Barriers for Frames* (PFBF), quanto ao número de utilizadores com boa ou excelente QoE que fazem *streaming* de vídeo. Também é analisado o número de utilizadores não satisfeitos (com má ou pobre QoE). Se se procurar garantir que 90 % dos utilizadores têm uma boa ou excelente QoE, independentemente do modo de operação do MBF, este suporta claramente mais utilizadores conectados do que o RR, BET, PF e PFBF. Em comparação com o RR, o MBF suporta mais 15% e 12% de utilizadores conectados, operando no modo 1 e modo 2, respetivamente. Para além disto, o MBF permite ter menos utilizadores com uma má ou pobre QoE do que o RR, o PF e PFBF, independentemente do modo de operação.

Palavras-chave: streaming móvel de vídeo, DASH, algoritmos de alocação de recursos, qualidade de experiência, throughput, buffer.

Abstract

In the last years, the multimedia content consumed by mobile users has exponentially increased, particularly video content. Today, mobile operators have two challenges. On the one hand, they need to satisfy the high expectations that clients have on the delivered quality of the services. On the other hand, mobile network capacity cannot be increased as fast as the demand growth. A possible solution is the development of intelligent schedulers that allocate resources very efficiently and satisfy the maximum possible number of clients, providing them with a good Quality of Experience (QoE).

Motivated by the scheduling algorithms that can be found in the literature, this thesis presents an overview of the state-of-the-art – notably to understand the current weaknesses – and proposes a new and effective solution that can be used in a Long Term evolution (LTE) network. It proposes a scheduler, Maximum Buffer Filling (MBF), that increases the number of users who are satisfied with their video streaming session. To do so, it makes use of the user's current buffer level which is a QoE metric reported by the clients which implement the Dynamic Adaptive Streaming over HTTP specification (DASH), standardized by MPEG and also known as MPEG-DASH. The reported radio channel status and video segment requests sent by the users to the base station are also considered in the scheduling process. MBF allocates resources according to the current buffer level of the users and their achievable buffer filling. It can operate in two modes (mode 1 and mode 2), according to operator's intention: to maximize the number of satisfied users or to minimize the number of non-satisfied ones. This thesis assesses the capacity provided by MBF and compares it with the ones provided by Round Robin(RR), Blind Equal Throughput (BET), Proportional Fair (PF) and Proportional Fair with Barriers for Frames (PFBF) schedulers, regarding the maximum number of satisfied users who have a good or excellent QoE streaming video. It is also analyzed the number of non-satisfied users (with a poor or bad QoE). To ensure that 90% of the users have a good or excellent QoE, MBF, regardless the mode in which it operates, clearly supports a higher number of users streaming video than RR, BET, PF and PFBF. With respect to RR, it supports more 15% and 12% of connected users, when operating in mode 1 and mode 2, respectively. Furthermore, MBF leads also to a smaller number of users with a bad or poor QoE than RR, PF and PFBF, regardless the mode in which it operates.

Keywords: mobile video streaming, DASH, scheduling algorithms, quality of experience, throughput, buffer.

Contents

List of figures

List of tables

List of acronyms

ABS	Adaptive Bitrate Streaming
ACR	Absolute Category Rating
APN	Access Point Name
BET	Blind Equal Throughput
BMFF	Base Media File Format
CDF	Cumulative Distribution Function
CLO	Cross-Layer Optimization
CPU	Central Processing Unit
CQI	Channel Quality Indicator
DASH	Dynamic Adaptive Streaming over HTTP
DCPF	Delay-Constrained Proportional Fairness
DM	Device Management
DRM	Digital Rights Management
DSCQS	Double Stimulus Continuous Quality Scale
DSIS	Double Stimulus Impairment Scale
EDF	Earliest Deadline First
FDD	Frequency Division Duplex
FIFO	First In First Out
FSIG	Frame Significance
GUI	Graphical User Interface
HDS	Adobe HTTP Dynamic Streaming
HLS	Apple HTTP Live Streaming
HOL	Head OF Line
HTTP	Hypertext Transfer Protocol
IBFF	ISO Base File Format
IEC	International Electrotechnical Commission
IP	Internet Protocol
ISO	International Organization for Standardization
LR	Loss Rate
LTE	Long Term Evolution
LWDF	Largest Weighted Delay First

MBF	Maximum Buffer Filling
MDC	Multiple Description Coding
MLBS	Mean Lost Burst Size
MOS	Mean Opinion Score
MPD	Media Presentation Description
MSE	Mean Square Error
MS-SSIM	Multi-Scale Structural Similarity
MT	Maximum Throughput
M-LWDF	Modified Largest Weighted Delay First
OFDMA	Orthogonal Frequency-Division Multiple Access
OIPF	Open IPTV Forum
PBCH	Physical Broadcast Channel
PC	Pair Comparison
PDP	Packet Data Protocol
PF	Proportional Fair
PFBF	Proportional Fair with Barrier for Frames
PSNR	Peak-Signal-to-Noise-Ratio
PSS	Primary Synchronization Signal
PUCCH	Physical Uplink Control Channel
PUSCH	Physical Uplink Shared Channel
QAAD	QoE-enhanced Adaptation Algorithm over DASH
QMC	Quality of Experience Measurement Collection
QoE	Quality of Experience
QoS	Quality of Service
RB	Resource Block
RNN	Random Neural Network
RR	Round Robin
RRM	Radio Resource Management
RTCP	Real-Time Transport Control Protocol
RTP	Real-Time Transport Protocol
RTSP	Real-Time Transport Stream Protocol
RTT	Round Trip Time
SAP	Stream Access Point

SINR	Signal-Interference plus Noise Ratio
SSIM	Structural Similarity
SSCQE	Single Stimulus Continuous Quality Evaluation
SSS	Secondary Synchronization Signal
SVC	Scalable Video Content
TCP	Transport Control Protocol
TDD	Time Division Duplex
TS	Transport Stream
TTA	Throughput To Average
TTI	Transmission Time Interval
UDP	User Datagram Protocol
URI	Uniform Resource Identifiers
VoD	Video on Demand
VoIP	Voice over IP
VSSIM	Video Structural Similarity
VQM	Video Quality Metric
XML	Extensible Markup Language

Chapter 1

Introduction

This chapter provides the scope and objectives of this thesis, and presents the motivation and the global context inspiring the problem to be solved. Finally, the thesis' main contributions and structure are also outlined.

1.1 Context and motivation

In the last years, the multimedia content consumed by mobile users has exponentially increased, particularly video content. Users are constantly downloading video streams, using video conferencing applications, or even broadcasting their own video streams to the Internet through social applications, congesting the network. Therefore, the need of quality assessment became essential since network operators want to control their network resources while maintaining a high degree of user satisfaction. The fact is that the measurement of technical parameters fails to give an account of the user experience, and therefore operators and content providers are searching for mechanisms capable of assessing such user satisfaction through Quality of Experience (QoE) techniques.

Unlike Quality of Service (QoS) metrics, such as packet loss rate, packet delay rate, packet jitter rate and throughput, that are typically used to indicate the impact on the video quality level from the network's point of view, QoE goes beyond pure technical measures. It considers the end-customer perception, who interacts with the device, its user-interface, the network, and the service behind it [1]. QoE has been defined as "the degree of delight or annoyance of a person experiencing an application, service, or system. It results from the fulfillment of his or her expectations with respect to the utility and/or enjoyment of the application, service or system in the light of the person's personality and current state" [2].

In [3], a protocol stack to form a conceptual relationship between QoS and QoE is proposed. It is shown in Figure 1.1 and has three levels: Network QoS, Application QoS and User QoS (QoE).

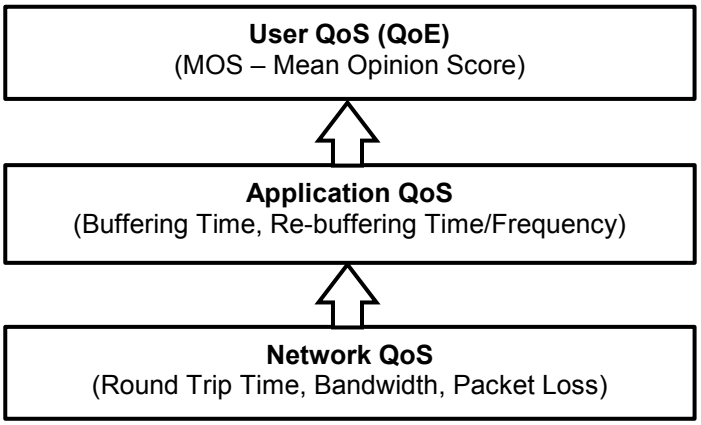

Figure 1.1 - Three levels of QoS proposed in [3].

Network QoS is the network path performance between a server and a client. Application QoS, on the other hand, reflects the performance from an application point of view. The QoE level reflects the overall performance from the user's point of view. It is actually on the top of the stack because it depends on the performance of the levels below. QoE is usually expressed using a Mean Opinion Score (MOS) where people evaluate a video after compression and transmission with a score ranging from 1 ("Bad") to 5 ("Excellent").

Clearly users' expectations about the services provided are getting higher. Nowadays users want video streaming services that are available (they want to reach the videos they want), consistent (they don't want any re-buffering events), and high-quality (they want content that looks great). In order to satisfy all these requirements, it is needed a high-functioning data center, high-throughput and resilient networks, and powerful mobile devices that handle the video streaming along with anything else the user might be doing at the same time. Furthermore, in a wireless network there are some users experiencing bad radio channel conditions. This results in undesired effects such as long initial buffering times, video pauses (stalls), long re-buffering periods, frequent changes in video quality, frame rate drops and/or in audio/video desynchronization, degrading user's QoE. Therefore, the development of high-performance physical-layer techniques is necessary, as well as intelligent resource management strategies to provide high throughput and efficient use of resources, while maintaining a high degree of user satisfaction. Among these strategies, scheduling algorithms have an important role and a lot of research still needs to be done. There is already some developed work that can be found in the literature but it is mostly found developed scheduling algorithms that improve only the mean video quality of the video downloaded or the mean frequency of stalls, etc. Particularly, there are very few studies that analyze the capacity of the schedulers regarding the number of users that can be streaming video with a good or excellent QoE in order to achieve a certain percentage of satisfied users and the QoE fairness level between the connected users.

1.2 Objectives

One of the most challenging issues that next-generation wireless networks will face is to provide quality of service and quality of experience guarantees to the large amount of multimedia applications that are emerging every day. In this context, much work has already been developed on the network resources scheduling process. However, most of the published work do not assess the performance of the proposed scheduling algorithms regarding the number of users who have a good experience streaming video.

In this context, the main objective of this thesis is to design, implement and assess an improved solution for scheduling network resources in a LTE network scenario where several mobile clients implementing the MPEG-DASH specification are streaming video. In particular, the goal is to design a scheduler that increases the network capacity in terms of the number of satisfied users streaming video, while maintaining a high level of QoE fairness. This scheduler makes use of the reported QoE metrics by DASH clients and their radio channel quality status. The proposed solution performance is then compared to the most well-known schedulers in the literature.

1.3 Main contributions

The main contributions of this thesis are related to the way the proposed scheduling algorithm allocates network resources, based on the QoE metrics reported by the DASH clients and their channel quality status. The scheduling algorithm not only achieves high network capacities regarding the number of users who are satisfied with their streaming session, but also maintains a high QoE fairness level between all the connected users. Furthermore, the provided QoE fairness level can be adjusted according to the mode in which the scheduler operates and the mobile operator's intention.

The performance assessment of some of the most well-known schedulers in the literature in terms of the number of satisfied and non-satisfied users is also done, in a scenario where several mobile users who have time-varying radio channels that were simulated using OMNET++, are connected to a LTE network, streaming video from the server.

Another contribution is the developed simulator for the analysis of performance of scheduling algorithms, rate adaptation algorithms for requesting video segments and QoE models, in a simulated LTE network scenario. In this dissertation, the essential base information for further development and implementation is also provided.

1.4 Thesis outline

This thesis is organized in seven chapters, with this first one introducing the work in terms of context, motivation and main objectives.

Chapter 2 presents a review of the most relevant concepts of MPEG-DASH specification. Firstly, the architecture of this technology and the protocols behind it are presented; next it is described how the video content exists and is organized in the servers; finally, the available DASH profiles are presented.

Chapter 3 briefly reviews the factors that influence QoE and the QoE assessment methods. It also presents the QoE metrics that a DASH client can report to the server and the reporting procedure.

Chapter 4 presents the proposed QoE-aware scheduling algorithm for adaptive HTTP video delivery in a LTE network.

Chapter 5 describes in detail the simulator developed in the scope of this work to assess the performance of the scheduling algorithms in study.

Chapter 6 presents the performance assessment methodology and the simulation parameters. Also, the assessment results are presented and discussed.

Chapter 7 concludes this thesis with a summary and suggestions for future work.

The Appendix A presents the pseudocode of the implemented scheduling algorithms, used to make the comparison with the proposed scheduler.

Chapter 2

DASH overview

2.1 Introduction

The proliferation of powerful mobile devices and resource-demanding multimedia applications is outpacing the capacity enhancements in next generation wireless networks. Internet traffic is growing quickly mostly because users are constantly downloading video streams, using video conferencing applications, or even broadcasting their own video streams (on live or on-demand) to the Internet, congesting the network. Furthermore, when it comes to wireless networks, they face problems such as background noise, narrow frequency spectrum and varying degrees of network coverage and signal strength. These lead to loss of data or re-transmissions causing delays, degraded video quality and re-buffering events when streaming the video, affecting user's QoE. In response to that, technologies like Adaptive Bitrate Streaming (ABS) have been developed. It starts by detecting a user's bandwidth and CPU capacity in real time and adjusting the quality of a video stream accordingly. It requires the use of an encoder which can encode a single source video at multiple bitrates. The player client [4] switches between streaming the different encoded contents that exist on the servers depending on available resources. So, when the user notice that video quality is going down, this is due to bitrate adaptation – it is getting lowered – to make sure that video streaming is not interrupted. ABS is built into technologies and products like Adobe HTTP Dynamic Streaming (HDS), Apple HTTP Live Streaming (HLS) and Microsoft Smooth Streaming. However, these solutions are closed systems with its own manifest formats, content formats and streaming protocols. To overcome this lack of interoperability among devices and servers of different vendors, MPEG developed a specification with the help of many experts and in collaboration with other standards groups, such as the Third Generation Partnership Project (3GPP) and the Open IPTV Forum (OIPF). The resulting MPEG standardization of Dynamic Adaptive Streaming over HTTP is now simply known as MPEG-DASH [5]. As described in ISO/IEC 23009-1 document, MPEG-DASH can be viewed as an amalgamation of the industry's three prominent adaptive streaming protocols – Adobe HDS, Apple HLS and Microsoft Smooth Streaming.

This chapter presents the main concepts and design principles about the adaptive streaming technique called MPEG-DASH. Firstly, a brief evolution towards adaptive bitrate streaming and a background on the motivations for MPEG-DASH are given. Then the architecture is presented, as well as it is explained how media content is organized in the servers. Finally, the profiles defined by MPEG-DASH are described.

2.2 Adaptive HTTP streaming

Especially in a wireless network environment, there are problems such as background noise, narrow frequency spectrum and varying degrees of network coverage and signal strength. To overcome these problems, it would be good to adapt the video bitrate and/or other parameters to better fit the

varying channel conditions. Thus, adaptive video streaming over a mobile network has been of concern in the research community, where a great number of solutions have been proposed. The evolution of video streaming towards adaptive streaming over Hypertext Transfer Protocol (HTTP) is briefly explained in Figure 2.1.

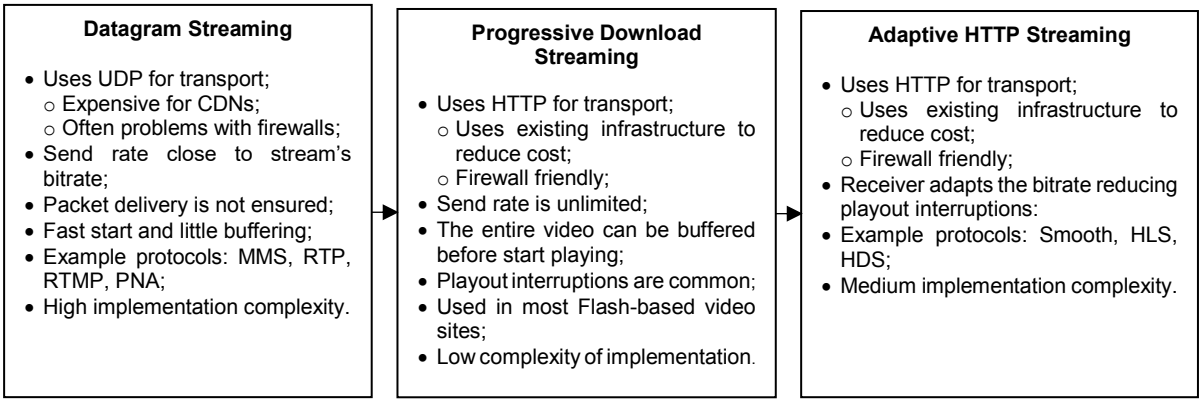

Figure 2.1 - Evolution of streaming toward adaptive HTTP streaming.

The traditional view of video streaming was based on a server sending encoded video data to the client at the rate of consumption, thus dictated by the typically variable bitrate that derives from compression of the video data. The protocol used for communicating client requests to the server was Real-Time Transport Stream Protocol (RTSP), with Real-Time Transport Protocol (RTP) and Real-Time Transport Control Protocol (RTCP) used for carrying the data and out of band stream management information, respectively. The initial idea was to use the above protocols over User Datagram Protocol (UDP) for some reasons:

- Real time transmission;
- Avoidance of any congestion control mechanisms;
- Avoidance of transmission rate control mechanisms present in TCP connections

Web Servers use HTTP as the main communication protocol at the application layer. The existing web infrastructure, and the fact that the port number 80 used by HTTP is usually open to firewalls, are strong reasons for adopting HTTP as the communication protocol for video data, rather than those described above in traditional streaming.

Progressive HTTP download was then adopted by most video servers, such as YouTube, whereby the HTTP request by the client is serviced by the server and the client media player presents the data contained in the file while the file is being downloaded. With progressive downloading, the client does not need to download the whole file before viewing parts of it. In the case that a user pauses the media player, progressive download will continue downloading the file until it is completed, and upon the viewer resuming play, the presentation will continue. If the user chooses not to resume download, the HTTP mechanism would clearly waste bandwidth. With traditional RTSP based streaming, pausing would send a message to the server to stop sending the data [6].

Both the RTSP and HTTP progressive downloading streaming mechanisms are based on the transmission of a given video file from the server to the client. However, this is a static selection that is made before transmission starts, and once started the same file is used till the end. In fact, there is a

great variability from user to user in terms of available bandwidth and mobile devices' capabilities which dictates the need for different encoding qualities and thus bitrates. Dynamic adaptation to network conditions is, therefore, an interesting way to overcome these variabilities. Adaptive bitrate streaming mechanism consists of changing the bitrate according to currently available resources (bandwidth, Central Processing Unit (CPU) load, battery charge left, screen size, etc.). The client chooses the bitrate as most of the information needed to select new bitrates is only known by the client, not the server. Often this adaptation is performed with coded-based approaches such as Multiple Description Coding (MDC) or Scalable Video Content (SVC) but they have almost no support in the market. Thus, another bitrate adaptation mechanism based on traditional codecs such as H.264 or HTTP has granted far greater popularity. This technology, known as adaptive HTTP streaming, is offered by companies like Microsoft, Apple or Adobe. Although this technology brings benefits for the content distributor (e.g., cheaper than specialized servers at each node, better scalability and reach to end network), it is the end user who gets the most out of it. Adaptive streaming supports fast start-up and seek times. Beginning at the lowest bitrate, it changes to a higher one and, as long as the user meets the minimum bitrate requirements, there should be no stalls followed by re-buffering events, disconnects or playback stutter. Adaptive HTTP streaming has become the most common transport protocol for adaptive streaming because it inherits all the benefits of progressive streaming while offering a solution to the fluctuating bandwidth problem. In Figure 2.2, a scenario is shown where the client switches smoothly to a stream with lower bitrate that exists on the HTTP Server whenever the buffer is almost empty or to a higher bitrate if the bandwidth permits it.

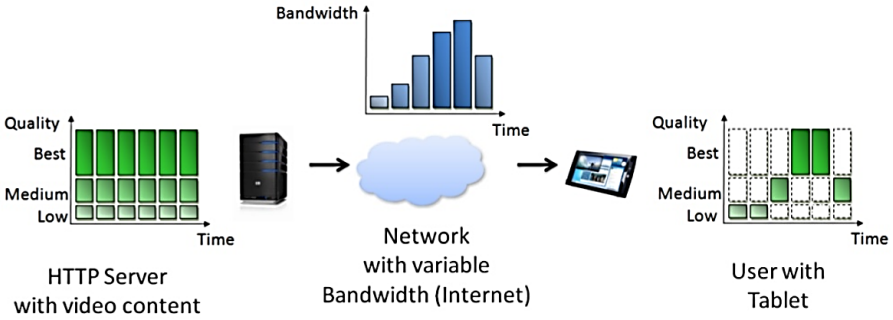

Figure 2.2 - Workflow of adaptive HTTP Streaming.

Several adaptive HTTP streaming formats are presently accessible, most significantly Apple's HTTP Live Streaming, Microsoft's Smooth Streaming and Adobe's HTTP Dynamic Streaming. These solutions are closed systems with its own manifest formats, content formats and streaming protocols. To allow interoperability, in 2009 MPEG developed a streaming standard in collaboration with many companies and organizations such as 3GPP and OIPF, giving birth to MPEG-DASH.

2.3 Scope of MPEG-DASH

Figure 2.3 illustrates a simple streaming scenario between an HTTP server and a DASH client. Only the orange blocks are defined by the specifications and are responsible for creating content and handle metadata, such as resources location and format. The green elements are outside of DASH

scope. So, the delivery of the metadata file and media encoding formats containing the segments is not specified. This also applies to the client's action for fetching, adapting and playing the content [7].

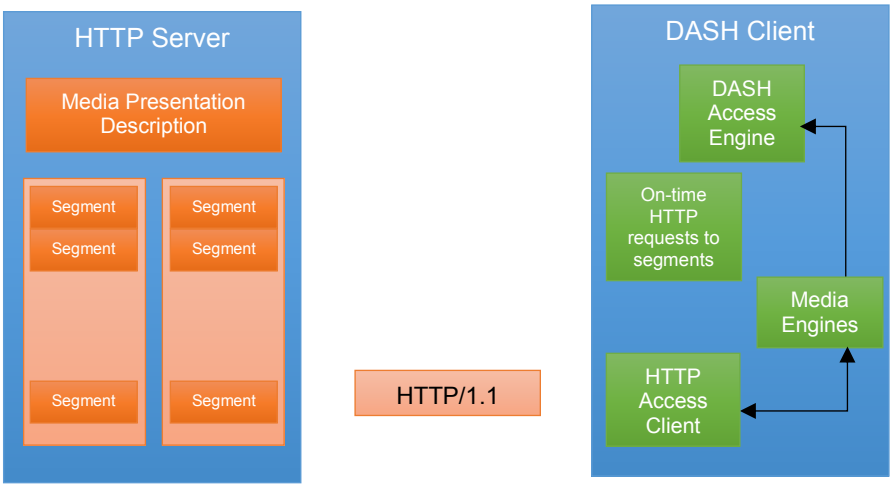

Figure 2.3 - MPEG-DASH system description.

The server holds variable encoded media data of the same content that is formed by a content creator generator. DASH specifies how content is produced and how metadata is handled. The content exists on the server in two parts that are stored into a regular web server:

- **Segments**: contain the actual multimedia bit streams in the form of chunks, in single or multiple files.
- **Media presentation description (MPD)**: a XML document that describes the temporal and structural relationships between segments. It provides sufficient information for the DASH Client to provide a streaming service to the user by requesting Segments from an HTTP server and demultiplexing, decoding and rendering the included media streams. It describes the content that is available in the server, including the URL addresses of stream chunks, byte-ranges, different bitrates, resolutions, and content encryption mechanisms.

DASH does not prescribe any client-specific playback functionality; rather, it just addresses the formatting of the content and associated MPD [8]. To play the content, the DASH client first obtains the MPD usually using the HTTP. The MPD allows the user to learn about the program timing, media-content availability, media types, resolutions, minimum and maximum bandwidths, and the existence of various encoded alternatives of multimedia components, accessibility features and required digital rights management (DRM), media-component locations on the network, and other content characteristics. Using this information, the DASH client selects the appropriate encoded alternative and starts streaming the content by fetching the segments using HTTP GET requests. After appropriate buffering to allow for network throughput variations, the client continues fetching the subsequent segments and also monitors the network bandwidth fluctuations. Using these measurements, the DASH client decides how to adapt to the available bandwidth by fetching segments of different alternatives (with lower or higher bitrates) to maintain an adequate buffer occupation [7].

2.4 DASH client model

Figure 2.4 illustrates the logical components of a conceptual DASH client model.

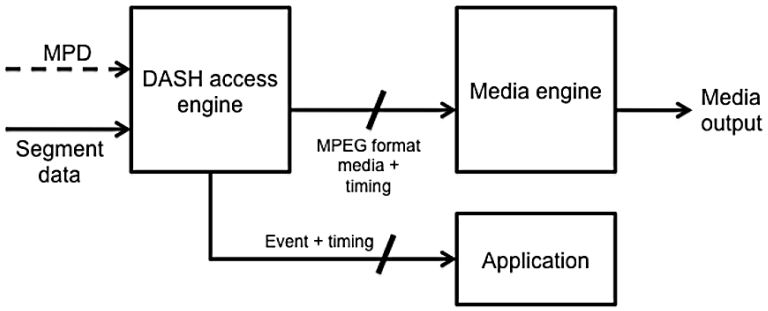

Figure 2.4 - DASH client model [9].

The DASH access engine receives the MPD, constructs and issues requests and receives Segments or parts of Segments. The output of the DASH access engine consists of media in MPEG container formats (ISO/IEC 14496-12 ISO Base Media File Format or ISO/IEC 13818-1 MPEG-2 Transport Stream), or parts thereof, together with timing information that maps the internal timing of the media to the timeline of the Media Presentation. In addition, the DASH access client may also receive and extract Events that are related to the media time. The events may be processed in the DASH client or may be forwarded to an application that is being executed. Examples for events are indication of MPD updates on the server, possibly providing the detailed update as part of the messages. The event mechanisms may also be used to deliver media time related application events, for example information about ad insertion, etc.

To play the multimedia file, the client player uses an URL to request the MPD file. By analyzing the MPD, the DASH client determines what are the viable options for this particular stream in terms of stream duration, bitrates available, location of the media, required DRM, resolutions. Using this information, the client selects the appropriate encoded alternative and starts streaming the content by fetching the segments using HTTP GET requests, as it can be seen in Figure 2.5.

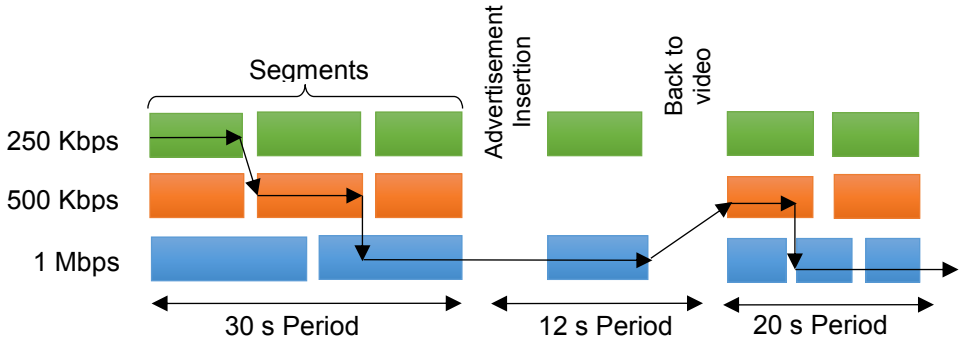

Figure 2.5 - Client streaming process.

The client entirely controls the delivery of services. After learning the information provided by the MPD the player client chooses which adaptive stream bitrate and resolution to play and switches to different bitrate streams depending on available resources: buffer conditions, network conditions, user change in resolution (e.g. full screen), device activity, etc.

2.5 DASH data model

A Media Presentation is a collection of data that is accessible to a DASH Client to provide a streaming service to the user. It is described by a MPD file that contains metadata required by the client to construct appropriate HTTP-URLs to access Segments and to provide the streaming service to the user. DASH is based on the hierarchical data model represented in Figure 2.6.

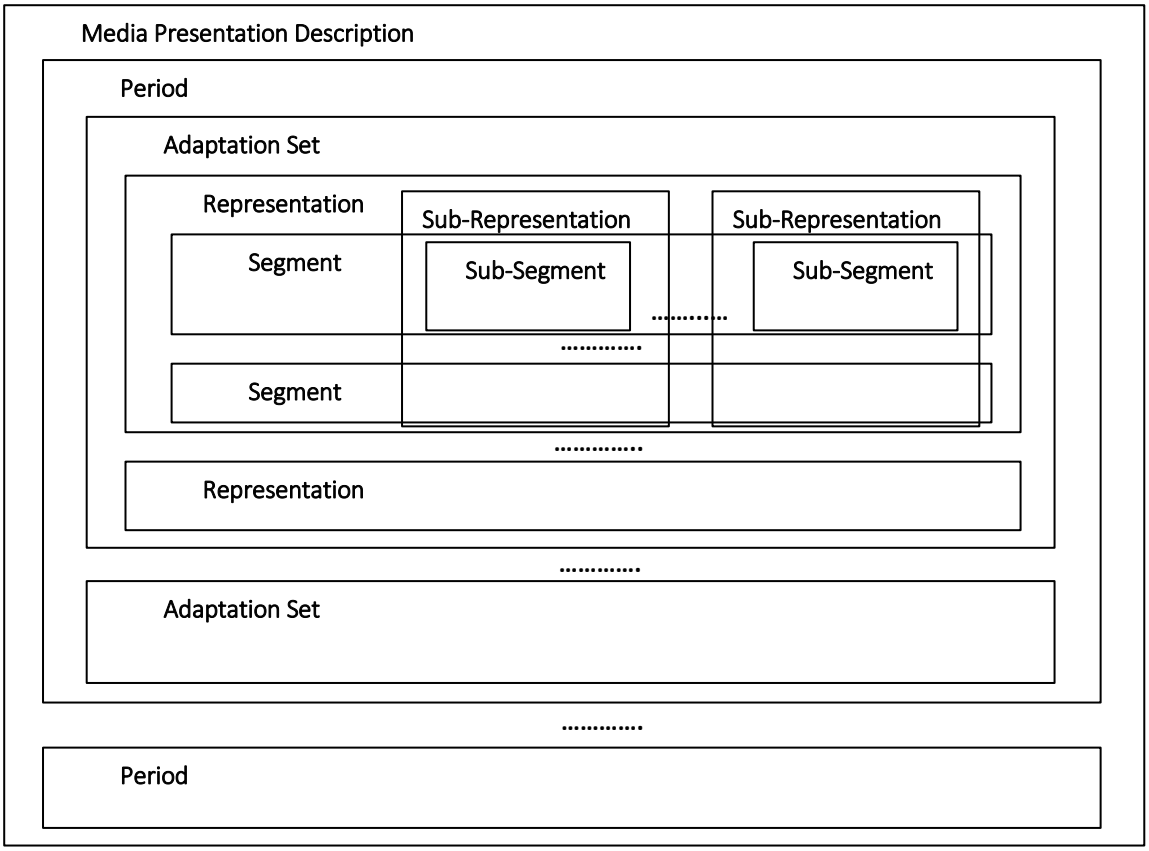

Figure 2.6 - DASH data model [8].

This means:

- A Media Presentation consists of a sequence of one or more consecutive non-overlapping Periods.
- Each Period contains one or more Adaptation Sets which include one or more Representations from the same media content.
- Each Representation consists of one or more Segments.
- Segments contain media data and/or metadata to decode and present the included media content.

A. Period element

The MPD describes the sequence of periods in time that make up the Media Presentation. Each MPD element consists of one or more Periods. A Period has a start time and typically represents a media content period during which a consistent set of encoded versions of the media content is available

i.e. the set of available bitrates, languages, captions, subtitles etc. does not change during a Period. Nevertheless, the client can adapt during a Period according to bitrates, resolutions and available codecs for that given Period. In addition, a Period might be used to separate content – insert an ad, change the angle of a live transmission – for example, if there is a need to introduce the ad exclusively in high definition while the rest of the content is available in a larger range of resolutions, it is just a matter of establishing a single Period for the ad containing only a high resolution [8].

B. Adaptation Set element

Within a Period, material is arranged into Adaptation Sets that allow to form logical groups with different components. An Adaptation Set represents a set of interchangeable encoded versions of one or several media content components. For example, there may be one Adaptation Set for the main video component and a separate one for the main audio component. If there is other material available, for example captions or audio in other languages, then these may each have a separate Adaptation Set [8].

C. Representation element

An Adaptation Set contains alternate Representations, i.e. only one Representation within an Adaptation Set is expected to be presented at a time. Different Representations in One Adaptation Set represent perceptually equivalent content but are encoded in different ways – codec, resolution, bandwidth, etc. Typically, this means that clients may switch dynamically from Representation to Representation within an Adaptation Set in order to adapt to network conditions or other factors. The Adaptation Set and the contained Representations shall be prepared and contain sufficient information such that the switching across different Representations in one Adaptation Set is seamless. Switching refers to the presentation of decoded data up to a certain time t, and presentation of decoded data of another Representation from time t onwards.

Representations may also include Sub-Representations to describe and extract partial information from a Representation. The Sub-Representation element describes properties of one or several media content components that are embedded in the Representation. It may for example describe the exact properties of an embedded audio component (e.g., codec, sampling rate, etc.), an embedded sub-title (e.g., codec) or it may describe some embedded lower quality video layer (e.g. some lower frame rate, etc.) [8].

D. Segment element

Within a Representation, the content may be divided in time into Segments for proper accessibility and delivery. In order to access a Segment, each one is described by an URL. Consequently, a Segment is the largest unit of data that can be retrieved with a single HTTP request. Segments are assigned a duration, which is the duration of the media contained in the Segment when presented at normal speed. Typically, all Segments in a Representation have the same duration. However, Segment duration may differ from Representation to Representation. Segments may be further subdivided into Sub-Segments each of which contains a whole number of complete access units. If a Segment is divided into Sub-Segments they are described by a compact Segment index within a Media Segment separately from MPD, which provides the presentation time range in the Representation and corresponding byte range

in the Segment occupied by each Sub-Segment. Clients may download this index in advance and then issue requests for individual Sub-Segments [8].

DASH does not impose a specific length for segments. Larger segments allow using less overhead bits as each segment needs to be requested by HTTP and each request demands an overhead. On the other hand, smaller segments are used for live situations or high-varying bandwidth scenarios as they allow faster transitions between bitrates [7] [8].

Figure 2.7 represents a possible composition of a Media Presentation, which includes several Periods, Adaptation sets and Media Segments. The Periods slice the video over time. The adaptation sets separate the content logically. The Representations represent the same period of content but with different characteristics (e.g. bandwidth) and finally the segments divide the content in shorter periods of time to enable the client for proper accessibility and delivery. Usually, segmentation produces an initialization segment that contains metadata which is necessary to present the media streams encapsulated in Media Segments.

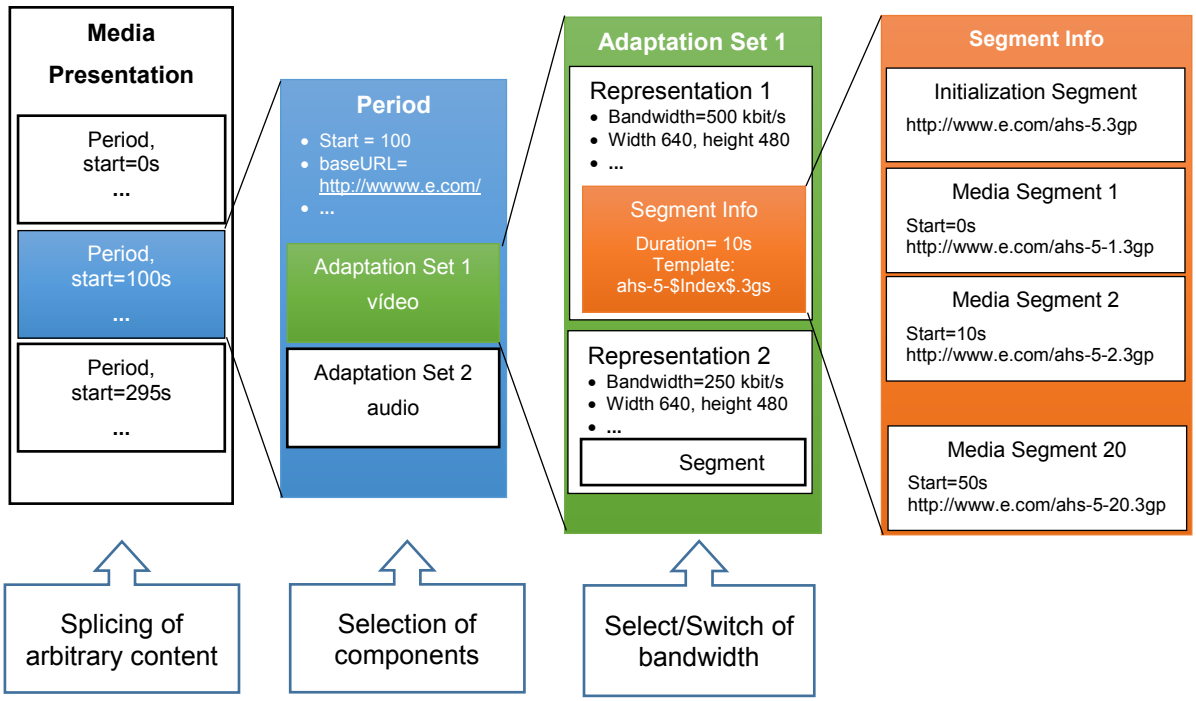

Figure 2.7 - DASH data model representation.

2.6 Protocols

Figure 2.8 shows a protocol stack for services in the context of the 3GPP DASH specification [9].

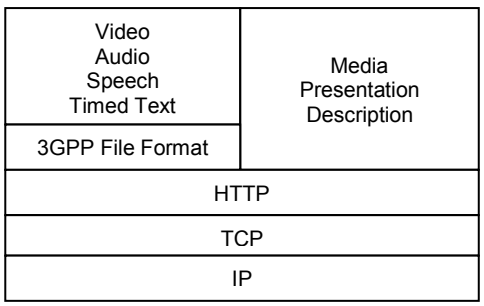

Figure 2.8 - Overview of the protocol stack [9].

The use of HTTP as an application layer protocol provides some advanced features such as caching, redirection or authentication. DASH clients should use the HTTP GET method or the HTTP partial GET method, as specified in RFC 2616 [10], clause 9.3, to access media offered at HTTP-URLs. The use of Transport Control Protocol (TCP) provides reliable, ordered, and error-checked delivery of a stream of octets between the server and the user. At the lower levels of the protocol stack, due to network congestion, traffic load balancing, or other unpredictable network behavior, IP packets may be lost, duplicated, or delivered out of order. TCP detects these problems, requests re-transmission of lost data, rearranges out-of-order data and even helps minimize network congestion to reduce the occurrence of the other problems. Once the TCP receiver has reassembled the sequence of octets originally transmitted, it passes them to the receiving application [11]. This way the media content has mathematically the same quality in the server and in the client side. However, the accuracy and reliability provided by TCP is paid with a higher delay when delivering content.

2.7 Media content

2.7.1 Format

In order to deliver media content, firstly it needs to be compressed to reduce its size during transportation, over Internet. To do so, it is used a codec, usually MPEG-1, MPEG-2, WMV or H.264 for video and AAC for audio. Once compressed, the content is packaged in containers to be transported to the end user. The most familiar containers are MP4, MPEG-2 TS and Flash. Under the MPEG DASH standard, the media segments can contain any type of media data. However, the standard provides specific guidance and formats for use with two types of segment container formats – MPEG-2 Transport Stream (MPEG-2 TS) and ISO base media file format (ISO BMFF). MPEG-2 TS is the segment format that HLS currently uses, while ISO BMFF (which is basically the MPEG-4 format) is what Smooth Streaming and HDS are using [5]. Finally, the codec will be also responsible for decompressing the content on the end user side.

2.7.2 Segmentation

The media content must be firstly segmented according to what was seen in section 2.5 and stored on servers so it can be requested by the users. The segmentation can result into multiple separated segment files or just one segment file with multiple sub-segments. Usually, segmentation produces an initialization segment (as it is represented in Figure 2.6) which initializes the media for playback. It contains metadata that is necessary to present the media streams encapsulated in Media Segments. This segment contains the stream access points (SAP), marked frames within the streaming, allowing the segment switching on the playback. A SAP is a position in a Representation enabling playback of a media stream to be started using only the information contained in Representation data starting from that position onwards. If there is not an initialization segment for a Representation, it means that each media segment within the Representation shall be self-initialized.

Segmentation and sub-segmentation may be performed in ways that make switching simpler. For example, in the very simplest cases each Segment or Sub-Segment begins with SAP and the

boundaries of Segments or Sub-Segments are aligned across the Representations of one Adaptation Set. In this case, switching Representation involves playing to the end of a (Sub) Segment of one Representation and then playing from the beginning of the next (Sub) Segment of the new Representation. Also the Media Presentation Description and Segment Index may make switching simpler as they provide various indications, which describe properties of the Representations [8].

2.7.3 Reproduction

The content exists on the server in two parts: the MPD and the segments that contain the multimedia itself. Firstly, the client requests the MPD and then he parses it in order to learn about the program timing, media-content availability, media types, resolutions, minimum and maximum bandwidths, and the existence of various encoded alternatives of multimedia components, etc., so he can select a collection of Adaptation Sets suitable for its environment. Within each Adaptation Set it selects one Representation, typically based on the value of an attribute of that Representation, called *Bandwidth*, but also taking into account client decoding and rendering capabilities. The value of the attribute bandwidth is the bitrate of the channel (supposing it is constant) that a client needs, to have enough data for continuous playout.

To access the content, firstly the initialization segment is requested (if there is one) and then the media ones. The client buffers a certain amount of media before starting the presentation. The content playback will begin as soon as it reaches a minimum amount of data. These requested files should be of low quality to minimize the initial buffering time and start playing the content as fast as possible. Once the presentation has started, the client continues consuming the media content by continuously requesting Media Segments or parts of Media Segments. The client may switch Representations taking into account updated MPD information and/or updated information from its environment, e.g. change of observed throughput. If a certain frame that was not yet downloaded is reached, the playback will stop (stall) and a re-buffering event will begin, degrading the QoE of the user.

2.8 DASH profiles

DASH specify profiles to enable interoperability and the signaling of the use of features. In Figure 2.9, the six available profiles defined by DASH are represented, each one addressing specific applications/domains. Each profile imposes a set of specific restrictions. Those restrictions are typically on features of the MPD document related to encoding of the segments and the source of the stream (it can be live or On Demand) and on Segment formats. The restrictions may be also on content delivered within Segments, such as on media content types, media formats, codecs, and protection formats, or on quantitative measures such as bitrate, Segment duration and size, as well as horizontal and vertical visual presentation size. The MPD has an attribute, called *Profiles*, which specifies a list of Media presentation profiles.

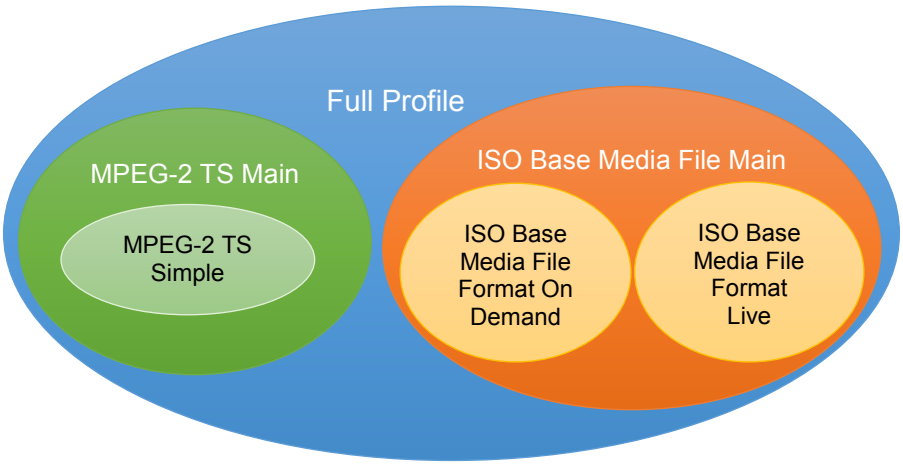

Figure 2.9 - MPEG-DASH profiles as defined in ISO/IEC 23009 [8].

There are three profiles defined relying on the **ISO Base File Format (IBFF)** encoded files [8]:

- **ISO Base media file format On-Demand** – Intended to provide basic support for on-demand content, supporting large Video on Demand (VoD) libraries with minimum amount of content management. The primary constraints imposed by this profile are the requirement that each Representation is provided as a single Segment, that Sub-segments are aligned across Representations within Adaptation Set and that Sub-segments must begin with SAPs. This allows scalable and more efficient use of HTTP servers and simplifies seamless switching.
- **ISO Base media file format Live** – Optimized for live encoding and low latency by encoding and immediate delivery of short Segments consisting of one or more movie fragments of ISO file format. Each fragment may be requested as soon as available, so usually it is necessary to request MPD update prior to each Segment request. Despite this profile is optimized for live services, it may work as well as an On-Demand one in case the MPD@Type attribute is set to 'static'.
- **ISO Base media file format Main** – Support for both On Demand and Live content. ISO Base media file format Live ISO Base media file format On-Demand are a subset of this profile.

There are also two profiles related to **MPEG-2TS** encoded files [8]:

- **MPEG-2TS Main** – Imposes little constraints on the Media Segment format for MPEG-2 Transport Stream content.
- **MPEG-2TS Simple** – It is a subset of MPEG-2TS Main profile. It poses more restrictions on the encoding and multiplexing in order to allow simple implementation of seamless switching.

The previous profiles simplify the content organization of the metadata file according to the encoding of the segments. There is also a sixth profile called the **Full Profile** which supports both ISO Base and MPEG-2 TS media file formats. This profile includes all features and Segment Types defined in ISO/IEC 23009.

A DASH client in agreement to a specific profile is only required to support those features and not the entire specification. It is also important to mention that external organizations may define themselves

profiles by posing restrictions, permissions and extensions. However, it is recommended that such external profiles be not referred to as profiles but as an Interoperability Point.

Chapter 3

QoE overview

3.1 Introduction

QoE is defined in [2] as “The degree of delight or annoyance of a person experiencing an application, service, or system”. Based on such experience, the user decides if he/she will continue to pay for that service or not. Therefore, it is important to understand what influences QoE, how QoE can be measured and how it can be improved. By accessing user’s QoE, operators can schedule resources over time in a way that all users achieve the same degree of satisfaction, for instance when using a video streaming service.

Firstly, in this chapter it is explained what are these factors that affect QoE and therefore MOS; then how QoE can be measured both in subjective and objective ways and finally the procedure the DASH Clients use to report QoE to the streaming servers. The metrics they report can actually be used to better schedule network resources between the users and therefore improve the overall QoE when they are streaming video in a wireless network environment.

3.2 Factors influencing QoE

QoE is usually expressed using MOS where people evaluate a video after compression and transmission with a score ranging from 1 ("Bad") to 5 ("Excellent"). The fact is that two different persons will probably perceive the experience in different ways. For instance, when they are streaming the exactly same video they may have different opinions about it. Any characteristic of a user, system, service, application, or context whose actual state or setting may have influence on the QoE for the user, can be regarded as an influencing QoE factor. The factors that influence QoE can be grouped in three categories: human, system and context factors [2]:

- **Human factors**: the user characteristics that can affect QoE. The users that are evaluating a video streaming experience are all different. They may have different gender, age, visual and auditory acuity, occupations or nationalities. Also their mood, motivation or educational background can influence their perception of quality. The previous experiences watching a video that a user had had might also influence “the degree of delight or annoyance” he/she experiences.
- **System factors:** factors related to all the transmission system that can change in some way the quality of the video that is being transmitted. It involves the encoding process, transmission network and user end-device. When encoding the content, many decisions influence the quality of the video such as the bitrate used (variable or constant), video frame rate and spatial resolutions. The codec itself can be lossy degrading the quality of the video. The transmission network is responsible for packet losses, delay and jitter which have impact on QoE. For instance, when a user is streaming a video from YouTube, the frequency of stalls, the frequency

of video resolution changes, or the time needed to fast-forward a video are factors that affect QoE. Lastly, also the users' devices can influence the experience they have when streaming a video. The screen size, computation power of the device and the application used itself affect the perceived quality.

- **Context factors:** external factors that affect QoE, such as the time of the day and the day of the week. Usually, users perceive a better quality during the evenings and during the weekends because they are more relaxed. The service type (live streaming or video on-demand and subscription type), video's popularity and duration affect the perceived quality too. For instance, a popular video is more likely to be tolerated when its quality is bad.

Clearly, QoE can be very subjective whereby it is relevant to know how QoE can be measured. QoE measurement methods are going to be explained in section 3.3. This is particularly important for network operators because the user is who decides if he/she will continue to pay for the service he obtains or not and this decision will depend on his/her degree of service satisfaction.

3.3 QoE measurement methods

QoE reflects the overall performance from the user's point of view. The most used metric for accessing QoE is MOS which can be assessed based on two approaches: subjective and objective. The only reliable and most appropriate and accurate method for assessing perceived video quality is the subjective one, where people are just asked for their opinion about the quality of the video.

3.3.1 Subjective assessment

Subjective testing for visual quality assessment has been formalized in ITU-R Rec. BT.500 [12] and ITU-T Rec. P.910 [13]. These recommendations define some of the most commonly used procedures for subjective quality assessment such as Double Stimulus Continuous Quality Scale (DSCQS), Double Stimulus Impairment Scale (DSIS), Single Stimulus Continuous Quality Evaluation (SSCQE), Absolute Category Rating (ACR) and Pair Comparison (PC). The outcome of any subjective experiment are quality ratings from viewers, which are then averaged for each test clip into Mean Opinion Scores (MOS). However, MOS is very expensive and time-consuming due to the need of a large amount of data and the involvement of many users. Furthermore, it is impractical in certain types of services such as live streaming. Therefore, there has been a trend toward using objective perceptual video quality algorithms.

3.3.2 Objective assessment

The idea of the objective assessment is to predict the QoE (MOS) using objective metrics through a QoS-QoE mapping function. There has been much research on finding the most suitable functions that predict QoE using QoS metrics.

Peak-Signal-to-Noise-Ratio (PSNR) and Mean Square Error (MSE), which are examples of the objective approach, only evaluate the spatial quality of videos, therefore they are not suitable or sufficient on its own for HTTP video streaming. However, when streaming a video on a mobile device, it is not

only important the image quality but all the user's experience because mobile video comes through the wide heterogeneity of content provider platforms, protocols, smartphones, mobile operating systems and video applications [14].

In [3], the authors point Round Trip Time (RTT), bandwidth, packet loss and jitter as the network QoS metrics (see Figure 1.1). Network QoS metrics can be estimated in the nodes inside the network. Relatively to the second level, the application QoS level, the authors propose the following metrics:

- **Initial buffering time**: This metric measures the period between the starting time of loading a video and the starting time of playing it.
- **Mean re-buffering duration**: When the amount of buffered video data decreases to a low value, the playback will pause, and the player will enter into a re-buffering state. This metric measures the average duration of a re-buffering event.
- **Re-buffering frequency**: This metric measures the frequency of the re-buffering events.

Application QoS metrics need to be measured at the application level, hence at the client side, and may require modifications to the application.

Initially in [3], the authors show how the network QoS metrics affect the application performance metrics. The higher the delay or loss rate, the higher the initial buffering time, mean re-buffering time or re-buffering frequency. Regarding the bandwidth, it is shown that the higher its value, the lower initial buffering time, mean re-buffering time or re-buffering frequency. Then, they focus on finding how the Application QoS level affect the QoE level. Based on a subjective quality assessment, where subjects were asked to score different videos with different initial buffering delays, mean re-buffering durations and re-buffering frequencies, the authors acquire the relationship between QoE (MOS) and application QoS metrics by performing a regression analysis. They also identified the re-buffering frequency as the main factor responsible for the MOS variance.

Additionally, to the applications QoS metrics identified in [3], other application performance metrics can be referred since they also affect QoE when streaming a video. Users' experience is influenced for instance by:

- **Spatial resolution:** generally higher frame resolutions are perceived as better (for example 1080p vs. 640 x 480 SD).
- **Video representation switches:** if adaptive streaming is used, perceptual quality is also impacted by the number of image quality changes during playback, especially those from a higher to a lower.
- **Frame rate:** frame rates for streamed video typically vary from around 30 frames per second down to 15 frames per second or below. A higher frame rate is more essential for high-motion scenes, such as a sporting event. It appears smoother, but increases the bandwidth and CPU requirements for the user system.
- **Frame rate variation:** a frame rate of 15 fps or below may appear "choppy," but in general a consistent frame rate is preferable to one that varies during playback scenes.
- **Audio/video synchronization:** audio that is out of sync with the video may be very annoying, particularly in scenes with dialogue.

All the metrics featured in bold are application level metrics but they influence QoE, and thus affect the predicted MOS. Some models that predict QoE based on some of the metrics featured in bold can be found in the literature such as the one that is extensively explained in section 5.4.

3.4 QoE in DASH

The development of evaluation methodologies, performance metrics and reporting protocols play a key role for optimizing the delivery of HAS services. In addition, QoE monitoring and feedback are beneficial for detecting and debugging failures, managing streaming performance, enabling intelligent client adaptation (useful for device manufacturers), and allowing for QoE-aware network adaptation and service provisioning (useful for the network operator and content/service provider). Having recognized these benefits, both 3GPP and MPEG bodies have adopted QoE metrics for HAS services as part of their DASH specifications.

3.4.1 QoE triggering and reporting

MPEG does not define mechanisms for reporting metrics, however it does define a set of metrics and a mechanism that may be used by the service provider to trigger metric collection and reporting at the clients, if a reporting mechanism is available on the client side. The trigger mechanism is based on the *Metrics* element in the MPD. This element contains the list of DASH Metrics for which the measurements are desired, the time interval and the granularity for the measurements, as well as the scheme according to which the metric reporting is desired. DASH clients should collect metrics based on the *Metrics* element in the MPD and report the collected metrics using one of the reporting schemes in the *Reporting* descriptor in the *Metrics* element. It is expected that elements containing unrecognized reporting schemes are ignored by the DASH client. If multiple *Reporting* elements are present, it is expected that the client processes one of the recognized reporting schemes. ISO/IEC 23009 does not specify any reporting scheme. It is expected that external specifications may define formats and delivery for the reporting data [8]. In this context, the 3GPP DASH specification provides mechanisms for triggering QoE measurements at the client device as well as protocols and formats for delivery of QoE reports to the servers [15]. Figure 3.1 depicts the QoE monitoring and reporting framework specified in 3GPP TS 26.247 [9].

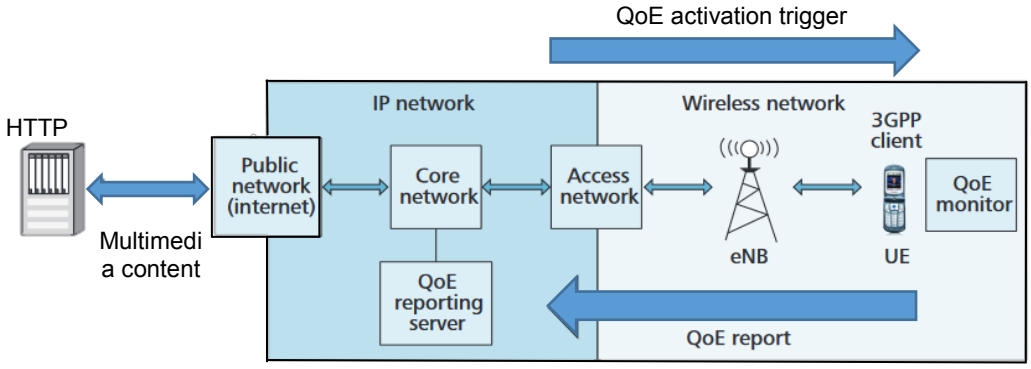

Figure 3.1 - Reporting framework for 3GPP DASH [15].

At a high level, the QoE monitoring and reporting framework is composed of the following phases:

- A server activates/triggers QoE reporting, requests a set of QoE metrics to be reported, and configures the QoE reporting framework.
- A client monitors or measures the requested QoE metrics according to the QoE configuration.
- The client reports the measured parameters to a network server.

3GPP TS 26.247 [9] specifies three options for the activation or triggering of QoE reporting. The first option is via the *Quality Metrics* element in the MPD, and the second option is via the OMA Device Management (DM) QoE Management Object (see Annex F of [9], for detailed information). The third one is using the QoE Measurement Collection (QMC) functionality. The trigger message from the server would include reporting configuration information such as the set of QoE metrics to be reported, the Uniform Resource Identifiers (URIs) for the server(s) to which the QoE reports should be sent, the format of the QoE reports (e.g., uncompressed or gzip), information on QoE reporting frequency and measurement interval, percentage of sessions for which QoE metrics will be reported, and access point name (APN) to be used for establishing the packet data protocol (PDP) context to be used for sending the QoE reports [15].

In 3GPP DASH, QoE reports are formatted as an Extensible Markup Language (XML) document complying with the XML schema provided in TS 26.247, [9]. The client uses HTTP POST request carrying XML-formatted metadata in its body to send the QoE report to the server. For further information see section 10.5 and 10.6 in [9].

3.4.2 QoE metrics

QoE reporting is optional, but if a DASH client reports DASH metrics, it shall report all requested metrics, according to the QoE configuration. The following metrics shall be supported by DASH clients supporting the QoE reporting feature and reported upon activation by the server [8] [9]:

- **HTTP request/response transactions** – This metric essentially registers the result of each HTTP request and corresponding HTTP response. For every HTTP request/response transaction, the client measures and reports:
  - Type of request (e.g., MPD, initialization segment, media segment);
  - Times for when the HTTP request was made and corresponding HTTP response was received;
  - HTTP response code;
  - Contents in the byte-range-spec part of the HTTP range header;
  - TCP connection identifier;
  - Throughput trace values for successful requests.

  From HTTP request/response transactions, it is also possible to derive more specific performance metrics such as the fetch durations of the MPD, initialization segment, and media segments [15].
- **List of representation switch events** - This metric is used to report a list of representation switch events that took place during the measurement interval. A representation switch event

signals the client's decision to perform a representation switch from the currently presented representation to a new representation that is later presented. As part of each representation switch event, the client reports the identifier for the new representation, the time of the switch event when the client sends the first HTTP request for the new representation, and the media time of the earliest media sample played out from the new representation [15].

- **Average throughput** - This metric indicates the average throughput that is observed by the client during the measurement interval. As part of the average throughput metric, the client measures and reports:
    - Total number of content bytes (i.e., the total number of bytes in the body of the HTTP responses) received during the measurement interval;
    - Activity time during the measurement interval, defined as the time during which at least one GET request is still not completed;
    - Wall clock time and duration of the measurement interval;
    - Access bearer for the TCP connection for which the average throughput is reported;
    - Type of inactivity (e.g., pause of presentation) [15].
- **Initial playout delay** - This metric signals the initial playout delay at the start of the streaming of the presentation. The initial playout delay is measured as the time in milliseconds from the fetch of the first media Segment (or sub-segment) and the time at which media is retrieved from the client buffer.
- **Buffer level** - This metric provides a list of buffer occupancy level measurements carried out during playout at normal speed. As part of the buffer level metric, the client measures and reports the buffer level that indicates the playout duration for which media data is available starting from the current playout time along with the time of the measurement of the buffer level.
- **Play list** - A list of playback periods. A playback period is the time interval between a user action and whichever occurs soonest of the next user action, the end of playback or a failure that stops playback. Decoded samples are generally rendered in presentation time sequence, each at or close to its specified presentation time. A compact representation of the information flow can thus be constructed from a list of time periods during which samples of a single representation were continuously rendered, such that each was presented at its specified presentation time to some specific level of accuracy (e.g. +/- 10 ms). Such a sequence of periods of continuous delivery is started by a user action that requests playout to begin at a specified media time (this could be a "play", "seek" or "resume" action) and continues until playout stops either due to a user action, the end of the content, or a permanent failure [15].
- **MPD information** - This metric can be used to report Representation information from the MPD, so that reporting servers without direct access to the MPD can understand the used media characteristics. The metric is reported whenever the client sends any other quality metrics report containing references to a Representation which MPD information has still not been reported.
- **Device information** - This metric contains information about the displayed video resolution as well as the physical screen characteristics. If the video is rendered in full-screen mode, the video resolution usually coincides with the characteristics of the full physical display. If the video is

rendered in a smaller sub-window, the characteristics of the actual video window shown shall be reported. If known by the DASH client, the physical screen width and the horizontal field-of-view shall also be reported. The metric is reported at the start of each QoE reporting period, and whenever the characteristics changes during the session (for instance if the UE is rotated from horizontal to vertical orientation, or if the video sub-window size is changed).

3.4.3 Quality metadata

In addition to the QoE reporting feature described in the sections 3.4.1 and 3.4.2, MPEG specifies some video quality metrics that are provided together with the content that exists on the servers. Quality metadata refers to video quality metrics based on associated measurements of the media data. The quality metrics can be carried in an ISO Base Media File Format and can be used to dynamically monitor and adjust image quality. For example, a network digital video server can examine the quality of video being transmitted in order to control and allocate streaming resources. Even the client action could take into consideration quality metadata when requesting Representations. These metrics should be accessible for all segments and Representations in the adaptation set, so that the DASH client can independently fetch them ahead of time before loading actual data segments. Some of the already existing metrics are:

- **Peak Signal to Noise Ratio (PSNR)**: for encoded video sequence it is defined based on per-picture mean square error (MSE) differences and computed as an average of all picture-level PSNR values obtained for all pictures in the sequence;
- **Structural Similarity (SSIM)**: for encoded video sequence it is defined based on SSIM index map obtained for each picture. It combines image structural information, as pixels mean, variance and covariance, using the full image resolution.
- **Multi-Scale SSIM (MS-SSIM)**: MS-SSIM for video sequence is computed as an average of all picture-level MS-SSIM values obtained for all pictures in the sequence. It applies SSIM over multiple scales of the image, through a process of multiple stages of sub-sampling.
- **Video Quality Metric (VQM)**: measures the perceptual effects of video impairments such as blurring, jerky/unnatural motion, global noise, block distortion and color distortion, using a quality metric for each artifact type, and combining all metrics into a single measure.
- **Mean Opinion Score (MOS)**: MOS for encoded video sequence is defined as the arithmetic average of result of a set of standard subjective tests [1] where several viewers rate the video sequence.
- **Frame Significance (FSIG)**: characterizes the relative importance of frames in a video sequence, and the sequence level visual impact from various combinations of frame losses.

For further and detailed information about these metrics, please see [16]. These metrics describe the video quality and exist on the server together with the content. However, when streaming a video on a mobile device, it is not only important the image quality but all the user's experience because mobile video comes through the wide heterogeneity of content provider platforms, protocols, smartphones, mobile operating systems and video applications [14]. Both quality metadata and the metrics reported by DASH Clients to the servers (section 3.4.2) are important for accessing QoE and

for scheduling purposes. Having access to the metrics reported by DASH Clients to the servers, it is possible to develop QoE-aware scheduling algorithms to improve the QoE of all users in a wireless network, when they are using a video streaming application.

Chapter 4

Scheduling algorithms

4.1 Introduction

Nowadays, mobile operators have two challenges. On the one hand, they need to satisfy the high expectations that customers have on the delivered quality of the services. On the other hand, mobile network capacity cannot be increased as fast as the demand growth. Thus, the development of high-performance physical-layer techniques, as well as intelligent resource management strategies is necessary to provide high throughput and efficient use of resources. Among these strategies, scheduling algorithms have an important role. Scheduling is conducted by allocating or reserving resources to users who want to download or upload content from or to the internet in a communication system. The scheduling algorithms may have different goals. They may try to maximize the resources efficiency, to provide a certain level of QoE to all the users or even to achieve the lowest possible standard deviation of QoE between the users. The network operators should choose the scheduling algorithm according to their objectives and/or complexity. These allocation algorithms can be based on one or multiple performance metrics. The data rate experienced by each user, fairness in resource allocation among users and average packet delay experienced by them are some of the possible metrics that can be used. The choice of what performance metric to optimize influences how resources are scheduled to the users and thus, impacts the overall QoE.

Regarding the LTE network, radio resources are allocated into the time/frequency domain. In the time domain, the total time is split in 10 ms frames, each one composed of 10 consecutive sub-frames or Transmission Time Intervals (TTI), each one lasting 1 ms. Furthermore, each TTI is made of 2 time slots with length 0.5 ms. Each time slot corresponds to 7 OFDM symbols. In the frequency domain, the total bandwidth is divided into sub-channels of 180 kHz. A time/frequency radio resource spanning over 1 time slot in the time domain and over one sub-channel in the frequency domain is called Resource Block (RB) (see Figure 4.1). On the frequency domain, it occupies 12 sub carriers' space which corresponds to 180 kHz (12×15 kHz). In time domain, it occupies one slot which is half of a sub-frame (0.5 ms). Each RB carries 12×7 symbols.

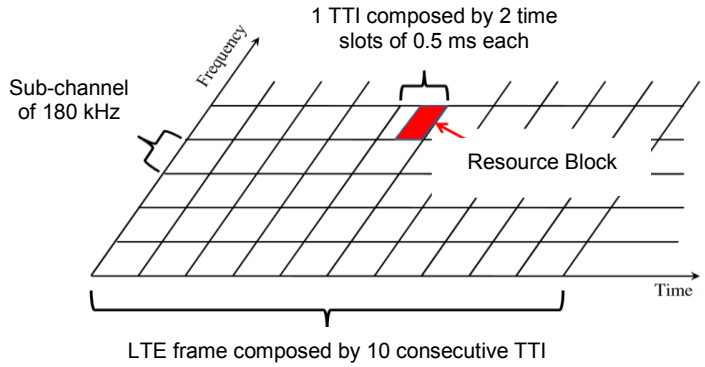

Figure 4.1 - Time-frequency radio resources grid [20].

Although resource blocks are defined over 1 time slot, the basic time-domain unit for scheduling in LTE is one sub-frame, consisting of two consecutive slots. This means that the scheduler makes allocation decisions every 1 ms (TTI). The reason for defining the resource blocks over one slot is that *distributed downlink transmission* and *uplink frequency* hopping are defined on a slot basis [17]. The minimum scheduling unit, consisting of two time-consecutive resource blocks within one sub-frame (one resource block per slot), is referred as a resource-block pair [17]. The bandwidths defined by the standard are 1.4, 3, 5, 10, 15, and 20 MHz. Table 4.1 shows how many resource blocks there are in each bandwidth.

Table 4.1 - Number of resource-block pairs available for each bandwidth.

Bandwidth [MHz]	Resource-block pairs per sub-frame (per 1ms)
1.4	6
3	15
5	25
10	50
15	75
20	100

For example, the 20 MHz bandwidth has 200 RB available per sub-frame. However, there are only 100 addressable locations of 2 RB each. This means that for each user, a minimum of 2 RB (1 RB pair) can be allocated per TTI. A maximum of 100 users can be served per TTI. Therefore, from now on, a resource-block pair is going to be called simply a resource block.

To sum up, what a scheduler is expected to do is to allocate these RB to the users that are uploading or downloading content to or from the network. Resource allocation is usually based on the comparison of per-RB metrics: the k^{th} RB is allocated to the j^{th} user if his metric $m_{j,k}$ is the biggest one among all users, i.e., if it satisfies the equation (4.1):

$$m_{j,k} = max_i\{m_{i,k}\} \tag{4.1}$$

These metrics can be interpreted as the transmission priority of each user on a specific RB. The $m_{i,k}$ metric can depend on the radio channel condition (the better the condition the higher the metric), buffer state (the higher the available space in the receiving buffer, the higher the metric), resource allocation history (the lower the past achieved throughput the higher the metric), etc. The choice of metric to use and therefore the way the resources are allocated results in providing a higher or lower throughput to a given user and thus in a certain level of satisfaction depending on his/her system requirements and expectations.

As it is going to be described in section 4.3, there are schedulers that are channel-aware and others that are channel-unaware. The first ones take into account the radio channel conditions of the users. In fact, it is important to mention that in a LTE network all the users who are connected to a base station report a parameter that keeps the base station informed about users' channel conditions, the Channel Quality Indicator (CQI), periodically. This parameter can be used somehow in the metric $m_{j,k}$, mentioned

in equation (4.1). Periodic CQI reporting can be sent on both Physical Uplink Control Channel (PUCCH) and Physical Uplink Shared Channel (PUSCH), along with data. Periodicities are defined for different values of CQI-PMI-ConfigIndex (Table 7.2.2-1A for Frequency Division Duplex (FDD) and Table 7.2.2-1C for Time Division Duplex (TDD)) [18]. The minimum periodicity could be 2 ms for FDD and 1ms for TDD. The efficiencies are presented in Table 4.2, from a CQI=0 to a CQI=15, according to the Table 4.1, in [18]:

Table 4.2 - Efficiencies for each CQI.

CQI index	Modulation	Efficiency (eff) [bits/symbol]
0	Out of range	
1	QPSK	0.1523
2		0.2344
3		0.3770
4		0.6016
5		0.8770
6		1.1758
7	16QAM	1.4766
8		1.9141
9		2.4063
10	64QAM	2.7305
11		3.3223
12		3.9023
13		4.5234
14		5.1152
15		5.5547

As it can be seen the higher the CQI, the higher the efficiency, meaning it is possible to transmit more bits per RB. Other tables like Table 4.1 are presented in [18], where different modulations are used, namely higher-order ones, increasing the resource block efficiencies too.

This chapter presents initially in the section 4.2 the desirable features of a good scheduling algorithm. Then, in section 4.3 and 4.4 some of the most studied scheduling algorithms in the literature are described, both the QoE-unaware and the QoE-aware schedulers. Finally, in section 4.5 it is presented the proposed QoE-aware scheduler solution.

4.2 Design aspects

In general, scheduling algorithms aim at providing efficient resource sharing, better performance in terms of throughput, link utilization, fairness and complexity. For designing a scheduling algorithm, it is relevant to know some of the desirable features that a good scheduling algorithm should have [19, 20]:

- **Delay bound:** the algorithm must be able to provide bounded delay in order to support multimedia services that are delay sensitive.
- **Throughput:** the algorithm should be able to provide short-term throughput guarantee for error free sessions and throughput over a sufficiently long period guarantee to all sessions.
- **Spectral efficiency:** the algorithm should not penalize sessions that temporarily exceed their reserved channel bandwidths provided the bandwidth is unused. Effective utilization of radio resources is one of the main goals to be achieved. To this aim, several types of performance

indicators can be considered: for instance, a specific policy could aim at maximizing the number of users served in a time interval or, more commonly, the spectral efficiency (expressed in bit/s/Hz) by always serving first the users that are experiencing the best channel conditions.

- **Fairness:** the algorithm cannot just look for the best spectral efficiency in bits/s/Hz. Maximizing the overall cell throughput surely enables effective channel utilization in terms of spectral efficiency, however it also results in a very unfair resource sharing among users and possibly in the "starvation" of some of them. Fairness should then be considered to guarantee minimum performance to the users that are experiencing bad channel conditions, normally the ones located in the edge of the cell.
- **Complexity and scalability:** a low complexity algorithm results in a reduction of processing time and memory usage. Finding the best allocation decision through complex and non-linear optimization problems or through an exhaustive research over all the possible combinations would be too expensive in terms of computational cost and time. A LTE packet scheduler works with a time granularity of 1 ms: it makes allocation decisions every TTI. In addition, in terms of scalability, the algorithm should operate efficiently as the available bandwidth and the number of users connected to a base station increases.
- **Robustness:** the algorithm should be robust to channel quality degradation. The effects of this degradation should be smoothed in order to provide the users a quality experience as little oscillatory as possible.
- **Isolation:** the algorithm should try to maintain the quality of the service of the users with a regular channel when some users suddenly experience a misbehaving session, for example, when their channel quality drops abruptly.

4.3 QoE-unaware schedulers

4.3.1 Channel-unaware strategies

Channel-unaware strategies are unrealistic because they do not consider the channel conditions, ignoring that the channel quality is time-varying. This fact makes them unsuitable in cellular networks. Nevertheless, they are fundamental because some of them are the basis of other more complex algorithms. These strategies have been historically adopted to face fairness, flow priorities and deadline expiration in all packet switching networks. The most used channel-unaware schedulers are the following:

- **First in first out (FIFO):** this algorithm just serves the users according to the order of requests. This behavior can be translated expressing the metric of the i^{th} user on the k^{th} RB as shown in equation (4.2):

$$m_{i,k}^{FIFO} = t - T_i \tag{4.2}$$

where t is the current time and T_i is the time instant when the request was made by i^{th} user. This technique is very simple, but both inefficient and unfair.

- **Round robin (RR):** this algorithm allocates radio resources cyclically to the users. The users are served one after another. Once all the users have been served, the first user will be served again. The metric is similar to the one defined in equation (4.2) with the difference that, in this case, T_i refers to the last time when i^{th} user was served. The principal advantages of Round Robin scheduling are the guaranty of fairness for all users in terms of the amount of resource blocks allocated to each user and an easy implementation. However, this approach is not fair in terms of user throughput. In wireless networks the throughput does not depend only on the amount of occupied resources, but also on the experienced channel conditions. Furthermore, this scheduler does not consider that different applications may require different bitrates.
- **Resource preemption:** several types of priority schemes can be defined. The idea is that the users are grouped in several priority classes, and the users belonging to a given class cannot be served until all users belonging to higher priorities classes have been served [20]. For example, this approach can be exploited to handle the differentiation between delay sensitive and non-sensitive services.
- **Weighted fair queuing (WFQ):** This scheduling algorithm allows another way to introduce priorities but avoids starvation too. The resources are shared accordingly to the proportion among the weights w_i (the higher the weight w_i of i^{th} user, more resources are allocated to him), but no starvation is possible due to the influence of the RR. The metric can be calculated using equation (4.3):

$$m_{i,k}^{WFQ} = w_i \,.\, m_{i,k}^{RR} = wi\ (t - T_i) \tag{4.3}$$

  where T_i refers to the last time when i^{th} user was served [20].
- **Guaranteed delay:** Guaranteed delay services require that each packet must be received within a certain deadline to ensure the good functioning of the service. This can be achieved by including into the metric information about the packet timing like the time instant when the packet was created and its deadline. Some policies that aim at avoiding deadline expiration, regardless channel quality conditions are [20]:
  - **Earliest deadline first (EDF**): schedules the packet with the closest deadline expiration. Its metric can be easily calculated with equation (4.4):

$$m_{i,k}^{EDF} = \frac{1}{(\tau_i - D_{HOL,i})} \tag{4.4}$$

    where τ_i is the delay threshold of i^{th} user (or expiration time) and $D_{HOL,i}$ is the Head of Line (HOL) delay, i.e., delay of the first packet to be transmitted by the i^{th} user. The more the head of line delay approaches the expiration time, the higher the priority of this packet of being transmitted and that means resource blocks must be allocated to the i^{th} user.
  - **Largest weighted delay first (LWDF):** based on the system parameter δ_i, representing the maximum probability for the HOL packet delay of the i^{th} user to exceed the delay threshold or deadline expiration time of the i^{th} user. The metric is calculated with equation (4.5):

$$m_{i,k}^{LWDF} = \alpha_i . D_{OHL,i} \quad (4.5)$$

where α_i is obtained using equation (4.6).

$$\alpha_i = -\frac{\log(\delta_i)}{\tau_i} \quad (4.6)$$

where α_i weights the metric and thus, the user with strongest requirements in terms of acceptable loss rate and deadline expiration time will be preferred for allocation.

4.3.2 Channel-aware / QoS-unaware strategies

Channel-awareness is a fundamental concept for achieving high performance in a wireless environment, and it can be used by exploiting radio resource management (RRM) features of LTE network such as channel quality indicator reporting. If one can estimate the channel quality perceived by a user on a given RB, in fact, it is possible to allocate radio resources obtaining very high data rate. To this aim, the most significant parameter is the expected achievable throughput for the i^{th} user at the t^{th}TTI over the k^{th} RB, i.e. $d_k^i(t)$. It depends on signal-interference plus noise ratio (SINR) and can be calculated using the Shannon expression for the channel capacity, as it can be seen in equation (4.7) [21]. Shannon's theorem indicates the maximum rate that can be achieved over a communication channel of a specified bandwidth *B* in the presence of noise:

$$d_k^i(t) = \mathrm{B}.\log(1 + SINR_k^i(t)) \quad (4.7)$$

Nevertheless, spectral efficiency is not the unique objective for a cellular network operator because it should be able to guarantee a minimum quality level of service also to cell-edge users avoiding their starvation. The following scheduling algorithms are some of the most well-known channel-aware/QoS-unaware strategies used to schedule the radio resources:

- **Maximum throughput (MT):** this scheduling algorithm prioritizes resources to the user with the best channel conditions. It aims at maximizing the overall throughput by assigning each RB to the user that can achieve the maximum throughput in the current TTI. Its metric can be obtained using equation (4.8):

$$m_{i,k}^{MT} = d_k^i(t) \quad (4.8)$$

  where $d_k^i(t)$ expected achievable throughput for the i^{th} user at the t^{th} TTI over the k^{th} RB. Although this leads to the best possible spectral efficiency (in bits/s/Hz), MT is very unfair to the users with poor channel conditions because their expected achievable throughput is low as well as their metric. Thus, the users further away from the base station are not served so frequently and will suffer from starvation.

- **Blind equal throughput (BET):** although this algorithm does not estimate the expected data-rate $d_k^i(t)$ and does not consider the current channel condition, it considers the channel quality by storing the past average throughput achieved by each user and using it as metric to calculate user priority for scheduling. A high past average throughput means that the user had a good

channel condition whereas a low past average throughput means that the user had bad channel condition. BET provides throughput fairness among all users regardless of their current channel conditions. The metric of the i^{th} user on the k^{th} RB is calculated as the inverse of its past average throughput at time t, $\overline{R_i(t)}$, using equation (4.9):

$$m_{i,k}^{BET} = \frac{1}{\overline{R_i(t)}} \tag{4.9}$$

$\overline{R_i(t)}$ is calculated as a moving average and it is updated every TTI for each user. Thus, BET allocates resources to users who had lower average throughput in the past. In this way, users with bad channel conditions are allocated more often than the others with better conditions [22]. Under this allocation policy, the user experiencing the lowest average throughput, will be served as long as he does not reach the same average throughput of other users in the cell.

- **Proportional fair (PF):** this scheduling algorithm provides a good compromise between throughput and fairness among the users. While trying to maximize the total throughput, it tries at the same time to provide a minimum quality of service to all the users, ensuring that none of users are starving. The algorithm tends to increase the probability of the users who have experienced lower throughputs of being scheduled. The metric can be expressed as shown in equation (4.10):

$$m_{i,k}^{PF} = \frac{d_k^i(t)^\alpha}{\overline{R_i(t)}^\beta} \tag{4.10}$$

where $d_k^i(t)$ expected achievable throughput for the i^{th} user at the t^{th} TTI over the k^{th} RB and $\overline{R_i(t)}$ is the past average throughput at time t of i^{th} user. The parameters α and β tune the "fairness" of the scheduler. By adjusting them, it is possible to adjust the balance between serving the users with the best channel conditions often and serving the ones who have achieved lower throughputs often enough that they have an acceptable level of service quality. In the extreme case $\alpha = 0$ and $\beta = 1$, the scheduler acts as a BET scheduler and $\alpha = 1$ and $\beta = 0$ then the scheduler is a MT scheduler and will always serve the user who has the best channel conditions, without considering fairness.

- **Throughput to average (TTA):** this scheduling algorithm can be considered as an intermediate between MT and PF. Its metric can be expressed as shown in equation (4.11):

$$m_{i,k}^{TTA} = \frac{d_k^i(t)}{d^i(t)} \tag{4.11}$$

where $d_k^i(t)$ is the expected data-rate for the i^{th} user at t^{th} TTI on the k^{th} RB and $d^i(t)$ is the wideband expected data-rate (the expected data-rate if all RB are allocated to the i^{th} user). The achievable throughput over all the bandwidth in the current TTI is used as normalization factor of the achievable throughput on the considered RB. It means that it quantifies the advantage of allocating a specific RB, and thus guaranteeing that the best RBs are allocated to each user. In fact, from its metric it is easy to see that the higher the overall expected throughput of a user is,

the lower will be its metric on a single resource block. This scheduler exploits channel awareness for guaranteeing a minimum level of service to every user.

4.3.3 Channel-aware / QoS-aware strategies

These strategies consider both the communication channel conditions and the different service requirements. Different applications may have different requirements. Some may demand a guaranteed minimum data rate for users whereas others may require a minimum delay when delivering the packets such as Voice over IP (VoIP) or video live streaming. Thus, the scheduler can treat data to guarantee some minimum required performance, either in terms of data rate or delivery delay:

- **Schedulers for guaranteed data-rate:** these scheduling algorithms are solutions for flows requiring guaranteed data-rate.

  In [23] it is proposed a strategy that works both in time and frequency domain. The authors point two reasons: limiting the number of multiplexed users with a time domain scheduler helps controlling the signaling overhead. Moreover, the frequency domain scheduling complexity generally depends on the number of input users to the frequency domain scheduler. In time domain, the users with highest priority (users with flows bellow their target bitrate and thus need to be urgently allocated to meet QoS requirements) are managed using BET and the rest is managed using PF. Once the users have been selected in the time domain, resources in the frequency domain are scheduled using also the PF algorithm, with the difference that $\overline{R_i(t)}$ is an estimation of the average achieved data-rate by the i^{th} user after he has been scheduled, thus considering the channel conditions.
- **Schedulers for guaranteed delay-requirements:** these scheduling algorithms aim at guaranteeing that the packets are delivered within a certain deadline.
  - **Modified LWDF (M-LWDF):** this scheduling algorithm is a channel-aware extension of LWDF. Its metric is calculated with equation (4.12) [20]:

$$m_{i,k}^{M-LWDF} = \alpha_i . D_{OHL,i} . m_{i,k}^{PF} = \alpha_i . D_{OHL,i} . \frac{d_k^i(t)^\alpha}{\overline{R_i(t)}^\beta} \tag{4.12}$$

    where $D_{OHL,i}$ is the delay of the first packet to be transmitted (QoS-awareness) by the i^{th} user and α_i is calculated as shown in equation (4.6). M-LWDF shapes the PF behavior by using information about the accumulated delay, assuring a good balance among spectral efficiency, fairness and QoS provisioning.
  - **EXP/PF:** this algorithm has been proposed in [24] and considers both the characteristics of PF and of an exponential function of the end-to-end delay. For real-time applications, the metric is expressed as shown in equation (4.13):

$$m_{i,k}^{EXP/PF,\ RTapp} = \exp\left(\frac{\alpha_i . D_{HOL,i} - \chi}{1 + \sqrt{\chi}}\right) . \frac{d_k^i(t)}{\overline{R_i(t)}} \tag{4.13}$$

    where χ can be calculated with equation (4.14):

$$\chi = \frac{1}{N_{rt}} \sum_{i=1}^{N_{rt}} \alpha_i . D_{HOL,i} \tag{4.14}$$

where $D_{OHL,i}$ is the delay of the first packet to be transmitted (QoS-awareness) by the i^{th} user and α_i is calculated as shown in equation (4.6). N_{rt} is the number of active downlink flows.

For non-real-time applications the metric can be calculated with the equations (4.15) and (4.16) :

$$m_{i,k}^{\frac{EXP}{PF},\ NRTapp} = \frac{\mathrm{w(t)}}{M(t)} \frac{d_k^i(t)}{\overline{R_i(t)}} \tag{4.15}$$

$$w(t) = \begin{cases} w(t-1) - \varepsilon\,, & D_{HOL,max} > \tau_{max} \\ w(t-1) - \dfrac{\varepsilon}{k}\,, & D_{OHL,max} < \tau_{max} \end{cases} \tag{4.16}$$

where $M(t)$ is the average number of packets waiting to be transmitted at time t, ε and k are constants, $D_{HOL,max}$ is the maximum HOL packet delay of all real-time service users and τ_{max} is the maximum delay threshold of real-time service users.

- **LOG rule:** this algorithm has been proposed in [25] and its metric increases logarithmically as the HOL delay increases, according to equation (4.17):

$$m_{i,k}^{LOGrule} = b_i . \log\left(c + a_i . D_{OHL,i}\right) . \Gamma_k^i \tag{4.17}$$

where a_i, b_i, c are tunable parameters and Γ_k^i represents the spectral efficiency for the i^{th} user on the k^{th} sub-channel and is computer from its CQI.

- **EXP rule:** an enhancement of the aforementioned EXP/PF has been proposed in [25] and its metric is calculated using equation (4.18):

$$m_{i,k}^{EXPrule} = b_i . \exp\left(\frac{a_i . D_{OHL,i}}{c + \sqrt{(\frac{1}{N_{rt}} \sum_j D_{OHL,i})}}\right) . \Gamma_k^i \tag{4.18}$$

where a_i, b_i, c are tunable parameters and Γ_k^i represents spectral efficiency for the i^{th} user on the k^{th} sub-channel. EXP rule also takes into account the overall network status, because the delay of the considered user is somehow normalized over the sum of the experienced delays of all users.

4.4 QoE-aware schedulers

QoE-aware schedulers are the ones that somehow take into account the QoE that the connected users are having using a certain service, trying to optimize it. Although some of them can be very complex, they can be very interesting for both the operators and end-users. In the end, the operators want to provide a certain level for QoE (a good one) for the largest number of users.

According to [26], QoE-aware scheduling strategies can be divided in three sub-groups according to the role of the user and his device in the scheduling process: passive end-user device strategies,

active end-user device / passive user strategies and active end-user device / active user strategies (see Figure 4.2). These strategies are described in the following three sub-sections.

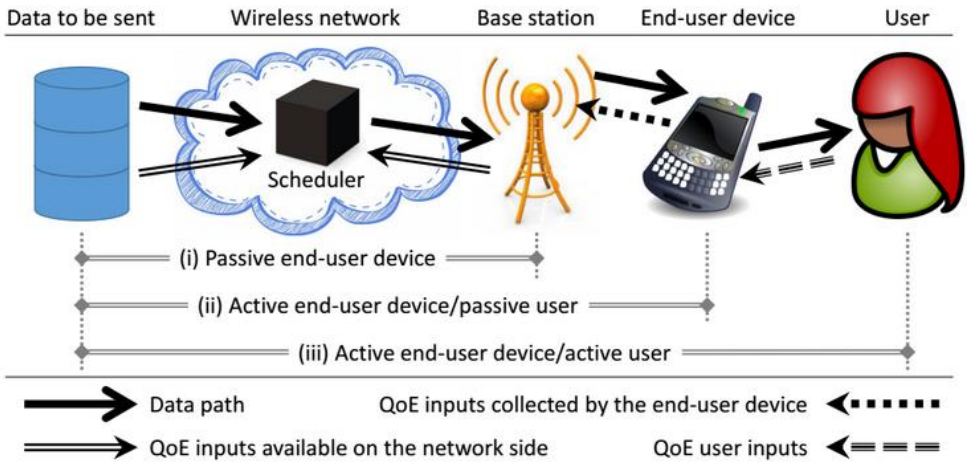

Figure 4.2 - QoE-oriented scheduling strategies classification [26].

4.4.1 Passive end-user device strategies

Regarding the passive end-user device strategies, neither the user nor his device needs to perform any exclusive QoE task such as measuring, monitoring or reporting relevant parameters. As the needed measurements can be carried out on the base station side, any extra information needs to be exchanged between the user's device and the network (see Figure 4.2). Many metrics cannot be known at the base station side and although some can be estimated, these strategies do not lead to the maximum QoE performance that can be obtained.

In [27] the authors apply a QoE based Cross-Layer Optimization (CLO) framework for efficiently allocating the network resources for video delivery in LTE mobile networks. They use an objective function to maximize the average user-perceived quality of all users by jointly optimizing the application layer and the lower layers of the radio communication system. They aim at optimizing the average user satisfaction by finding the optimal network resource allocation with a constraint of limited network resources. The authors use a linear mapping between Video Structural Similarity (VSSIM) and MOS with an upper and lower bound and use the MOS to indicate the user satisfaction of the video service.

In [28], the authors focus on a QoE-driven cross layer optimization for wireless video delivery. They aim to achieve a maximum average perceived quality of all users which can be interpreted as how efficient the network resources are used and distributed to all users. Furthermore, they try to achieve a similar perceived quality among all users. Their goal is to provide a mechanism that allow the network operators to dynamically adapt between achieving a maximum system efficiency and achieving the minimum unfairness of perceived quality of the services between the users. They define the system efficiency as the total sum of the MOS values perceived by all users and unfairness as the perceived quality difference between the user experiencing the highest MOS and the user experiencing the lowest MOS. MOS is also calculated using VSSIM metric.

In [29], the authors propose a cross-layer technique, which takes DASH related information at the base station scheduler into account, in order to optimize multi-user resource assignment for video

transmission over the LTE downlink. They measure the quality of the service by accessing the playout interruptions probability. In particular, the proposed scheduler takes into account information saved within the MPD regarding the available content stored on the DASH server such as the number of available video versions for each content and/or the corresponding video characteristics, which could be the required bitrate for the requested video-stream.

In [30], a real-time adaptive resource allocation algorithm considering the end user's QoE in the context of video streaming service is presented. The scheduler is based on a metric that takes account of user's video code rate (required data rate) and network performance (throughput). The proposed scheme aims at providing the capability of adjustment of system efficiency, fairness and correlation between the required and allocated data rates. The proposed algorithms are examined in the context of 3GPP-LTE [5] for both adaptive and non-adaptive video streaming scenarios.

In [31], a QoE oriented cross-layer downlink scheduling for heterogeneous traffics in LTE Networks is presented. The objective of such cross layer is to maximize user-perceived quality and maintain a certain level of fairness among users. For video streaming, once a user makes a request, the video distortion is calculated based on the packet loss rate. The channel distortions are also calculated based on the CQI reports of the user. Using the video distortion and channel distortion, the corresponding MOS is calculated. Based on MOS and HOL packet delay, the allocation decision is made. Meanwhile for non-real-time traffic, the proposed framework utilizes proportional fair scheduling scheme to ensure high throughput and fairness among the non-real-time service users.

In [32], a QoE-aware scheduling algorithm for video streaming that takes QoE into account when making allocation decisions is proposed. In order to get QoE feedback in real-time, the authors use a technique called Pseudo-Subjective Quality Assessment or PSQA [33], which is based on statistic learning using Random Neural Network (RNN). The idea is to train the RNN to learn the mapping between MOS and technical parameters, namely loss rate (LR) and mean lost burst size (MLBS). The values of MOS are computed using average score obtained by a panel of human observers. Using this QoE based strategy it is possible to allocate resources giving higher priority to the users who have higher constraints in terms of QoE.

In [34], a QoE-aware scheduler for HTTP Progressive Video in Orthogonal Frequency-Division Multiple Access (OFDMA) is presented. The authors propose the use of a metric that directly affects the end-user experience, namely an estimation of the amount of video data stored in the player buffer, for resource allocation. It is important to note that if the buffer level is reported from the client device using a framework such as the proposed in [9], the accuracy of the estimation could be improved but in that case, it would not be a passive end-user device strategy anymore.

In [35], a QoE-aware video scheduling algorithm in LTE networks is presented, referred as "Delay-Constrained Proportional Fairness (DCPF)". The objective of the algorithm is to meet the QoE requirements of real-time and non-real-time applications, specifically, streaming video applications by avoiding re-buffering events. To this aim the algorithm needs packet delay and average packet delay values which are measured at the base station from the periodic acknowledgments received from each user.

In [36], the authors propose online scheduling policies to optimize QoE for video-on-demand applications in wireless networks. The QoE of each flow is measured by its duration of video playback interruption. Furthermore, the playback buffer status is estimated at the wireless access point. The authors study in particular the heavy-traffic regime in order to address not only the long-term average performance, but also short-term QoE performance.

4.4.2 Active end-user device / passive user strategies

These scheduling algorithms consider information that is feedbacked from the users' device. The user does not have an active role. In this type of schedulers are inserted the DASH-based ones. For example, clients implementing DASH specification can send some metrics to the base station such as the buffer level, initial playout delay, average throughput and others (see section 3.4.2) and they can be used to allocate resources in such a way that optimizes the overall quality experience of the users.

In [37], a QoE-based cross-layer design scheme for resource allocation of video applications is presented. The authors establish a mapping model between PSNR and MOS to present the relationship between objective parameters and human visual perceiving model. Based on this model, they propose a QoE prediction function to maximize QoE of users while guaranteeing fairness. The calculation of PSNR is made based on video distortion which is estimated and reported by the users' devices according to channel quality and packet loss rate.

In [38], QoE-oriented scheduling for YouTube is described and evaluated in a OFDMA network. The proposed scheduler dynamically prioritizes YouTube users against other users if a QoE degradation is imminent. The prioritization is done in a proactive way according to the buffered playtime of the YouTube video player. The scheduler incorporates client-based feedback in the scheduling decisions at the base station. A user who watches a YouTube video triggers a feedback event if the playback buffer is exceeding β or falling below α threshold. In the scheduler, a flow is tagged as being in a critical state if feedback is received indicating that the buffered playtime is below the threshold α. A flow is tagged as normal if feedback is received indicating that the threshold β is exceeded.

In [39], an approach for joint transmission scheduling (capacity allocation) and video quality selection in small networks is presented. The work aims at designing an efficient system that supports a high number of unicast HTTP adaptive video streaming sessions in a dense wireless access network. The authors use QoE-related performance metrics reported by users such as total re-buffering time, initial buffering delay, average requested video bitrates and video bitrate fluctuations. With this information, it is possible to provide a desired video quality to the users and to allocate the remaining capacity, if available, in a fair manner in order to serve a larger number of users or to eventually enable users to switch to a higher video quality if requested.

In [40], the authors propose a QoE-aware radio resource management (RRM) framework which works in conjunction with adaptive streaming framework. They propose the Proportional Fair with Barrier for Frames (PFBF) algorithm that acts like a proportional fair until some user's buffer is below a certain threshold, forcing the scheduler to serve this user first. The priorities among users for resource allocation

are adjusted based on the dynamic feedback of the QoE metric of re-buffering percentage. The first user to be served by the scheduler is the one who has the highest metric $m_{i,k}$:

$$m_{i,k}{}^{PFBF} = V_i(\frac{\alpha \times d_k^i(t)}{S_{frame,i}} \times exp(\beta(f_{min} - f_i)) + \frac{d_k^i(t)}{\overline{R_i(t)}})\} \quad (4.19)$$

$$V_i = \begin{cases} 1 + \frac{n \times p_{rebuf,i}}{\sum_{j=1}^{n} p_{rebuf,j}} & if \sum_{j=1}^{n} p_{rebuf,j} > 0 \\ 1 & otherwise \end{cases} \quad (4.20)$$

where $d_k^i(t)$ expected achievable throughput, $S_{frame,i}$ is the size of the frame in transmission, f_i is the number of frames in the playback buffer, $\overline{R_i(t)}$ average of delivered throughput of the i^{th} user and f_{min} is tunable, representing minimum number of frames in the playback buffer. Higher value of f_{min} implies higher fairness being guaranteed by the network to users in terms of their playback buffer. As long as the video frames in the playback buffer f_i exceed the minimum value of f_{min} PFBF allocates approximately like the proportional fair (PF). When f_i drops below the minimum value then the user metric increases, increasing the probability of being served. The scalar V_i includes fairness in terms of re-buffering percentage. $p_{rebuf,i}$ is the percentage of the total streaming time spent re-buffering and n is the total number of connected users. The DASH clients feedback this parameter to the base station. α and β are tunable parameters.

In [41], the authors propose a cross-layer media-buffer aware optimization framework for wireless resource allocation that constraints re-buffering probability for adaptive streaming users. They present the Re-buffering Aware Gradient Algorithm (RAGA) which is based on periodic feedback of the buffer status of video clients standardized in the DASH standard, where priorities are given not only considering the buffer levels, but also considering the rate of change of these buffer levels. The authors prove that RAGA has better results than PFBF in terms of re-buffering percentage. However, a lot of values of parameters needed to calculate the user's metrics are missing in [41] and for that reason it is not possible to assess the results presented in the paper.

4.4.3 Active end-user device / active user strategies

The active end-user device / active user scheduling strategies are the ones that receive both inputs from the user's device and from the user himself. The users have an active role and can influence the way the scheduler allocates the resources through their actions. For example, for video streaming application the users may feedback their QoE at the end of the video. The user interaction with the video (e.g.: the number of times the user pauses the video or changes the resolution) may be also traced.

In [42], the authors propose a QoE framework that allows users to dynamically and asynchronously express their (dis)satisfaction with respect to the instantaneous experience of their service quality at the overall network QoS-aware resource allocation process [42]. The users use a graphical user interface (GUI) to express his preference. Its preference is simply the chose for a better or worse video quality. Each user has a utility function which is being adapted according to the user feedback. After all the utility functions have been adapted, an optimization problem is set and solved. In the case of a non-feasible solution the user is informed via its graphical interface.

In [43], the authors propose a model to derive the individual user's QoE function from users' online feedback upon users' participation. Each user sends a single bit feedback that indicates the satisfaction or dissatisfaction upon the end of the service. Hence users are in the control loop in optimizing QoE. By imposing fairness constraints that guarantee a minimum QoE the authors convexify a non-convex sigmoid optimization problem. They finally prove that the users' participation improves the average QoE as well as edge users' QoE

4.5 Proposed QoE-aware scheduling algorithm

In this section, it is explained the development process of the proposed scheduling algorithm in this thesis in section 4.5.1 and the final algorithm in section 4.5.2. Note that a lot of unsuccessful experiments were made. However, for space reasons only the relevant ones with reasonable results are presented. The simulations presented were done using the simulator developed and presented in section 5.

4.5.1 Algorithm development aspects

Among the algorithms studied in sections 4.3 and 4.4, some of them are fair in terms of the number of RB allocated (RR) or regarding the users' QoE (BET) or even in terms of re-buffering percentage (PFBF). Others look for high network efficiencies in bits/s/Hz (MT and PF) or seek to guarantee the delay or data-rate requirements of the data packets and flows. However, from the point of view of a mobile operator, a good scheduler may not be the fairest in terms of QoE neither the most efficient one in bits/s/Hz. An operator may be interested in a scheduling algorithm that allows to have the maximum number of satisfied users with a certain level of QoE, ensuring that the lowest possible number of users have a very low QoE, achieving this way some fairness regarding the users' QoE too. With such scheduler, it is possible to have a high network capacity regarding the number of satisfied users streaming video and guarantee that the clients will not stop paying for the service and look for another service provider. From this point of view, a lot of research needs to be made in order to support the current mobile video streaming traffic growth. The proposed algorithm in this thesis, Maximum Buffer Filling (MBF) tries to achieve these targets, using a QoE metric which is reported by the clients that implement the MPEG-DASH specification (the buffer level), the reported status of their channel conditions (CQIs) and the users' requested video segment bitrate. It actually supports more satisfied users than RR, BET, PF and PFBF and less non-satisfied users than RR, PF and PFBF schedulers (as it will be seen in section 6.5). Furthermore, MBF has a low computational complexity because it does not know which is the rate adaptation algorithm the users are using to request the video segments. It just uses a metric as simple as RR, BET and PF and simpler than PFBF. Among all the studied scheduling algorithms described in sections 4.3 and 4.4, RR, BET, PF and PFBF were chosen to make the comparison with MBF in section 6.5. The pseudocode of these algorithms is presented in Appendix A. Each of these algorithms has a reason to have been chosen:

- RR serves as a reference in section 6.5 because it is the simplest scheduling algorithm. It allocates the same number of RB to all users and has a very low computational complexity. It is a good scheduling algorithm as long as the users 'channel conditions are similar to each other and do not vary much along the time. However, in a scenario where the channel quality varies

significantly and users report different CQIs, the RBs keep being allocated equitably to the connected users and some may suffer from starvation and have a poor QoE.

- BET is the QoE fairest algorithm providing the same QoE to all the users, if they implement the same rate adaptation algorithm for requesting video segments. This is achieved by serving always the user with the lowest average throughput. This way, BET ensures that all users receive the same number of bits, request the same segment qualities and finally have the same buffer levels. However, BET is very sensitive. A few users with very bad channel conditions need a lot of RB and this can greatly affect the users' QoE. Once MBF is a QoE-aware scheduler and it seeks to increase the number of satisfied and have few non-satisfied users as the total number of connected users increases, BET is very relevant.
- PF tries to take advantage of the users' channel conditions. The PF metric multiplies the BET's metric by the user achievable data-rate. This avoids the BET's problem stated before because not only the average throughput is considered but also the current achievable throughput. Furthermore, PF is largely used in the literature to make comparisons.
- PFBF modifies PF including fairness in terms of re-buffering percentage [40]. The authors affirm that scheduler designs that are based on average rates utility functions (like PF) may correlate poorly with playback buffer status. PFBF leads to smaller re-buffering periods and therefore it is used in the study of section 6.5. Note also that PFBF requires the feedback of each user's playback status and re-buffering percentage. Such as PFBF, MBF makes use of buffer level in its metric trying to avoid stalls and decrease the re-buffering periods.

In section 6.5.1, it is analyzed the number of satisfied users that the scheduler support for a given high percentage of satisfied ones. If the percentage is 100%, BET is the best scheduler because for this percentage of satisfied users, it supports the highest number of connected users. However, 100% may be too severe. For percentages like 80 or 70%, BET is not the best algorithm anymore. In fact, it has a bad performance, supporting a smaller number of connected users than RR, PF and PFBF (as it can be seen in section 6.5.1). In section 6.5.2, it is analyzed the percentages of non-satisfied users as the total number of connected users streaming video increases. In this study, BET is clearly better than RR, PF and PFBF. Using the BET scheduler, only if there is a lot of connected users, there will be one or more non-satisfied users. So, MBF aims at maximizing the number of satisfied users for high percentages of satisfied users (trying to be better than RR, PF and PFBF) while maintaining the lowest possible number of non-satisfied users (similar to BET).

In section 3.3.2, some QoE metrics like the spatial resolution of the video, the frame rate and the re-buffering durations and frequencies were described. In [44], it is shown using subjective evaluations that the users of the experiment preferred a fluent playback of the video above a higher resolution, frame rate, and bitrate. It is a well-known fact that re-buffering events can be very annoying and greatly impact the users' QoE. Even if the client's implemented rate adaptation algorithm is QoE-aware, like the one presented in section 5.3, a re-buffering event can happen due to abruptly channel condition variations. Let's see an example of the impact of the re-buffering events using the QoE model presented in section 5.4. Two different users downloaded the same amount of video (120 seconds), with an average quality of 7 (assuming there are 13 different quality levels available on the server) and a standard

deviation of 0.1. User A did not have any stall whereas user B had one followed by a re-buffering event lasting 3 seconds. Using the QoE model presented in section 5.4, user A had a QoE equal to 3.17 whereas user B had a QoE equal to 2.17. From this example, the idea was to develop a scheduler based on the users' buffer levels, a QoE metric that clients implementing the MPEG-DASH specification may report. Using this metric, the scheduler should firstly try to avoid a re-buffering event. If a stall happens, the re-buffering period must be as short as possible. So, the first idea was to simply allocate the RBs to the users with lower buffer levels. From this point of view the first tested metric was the one presented in equation (4.21).

$$m_{i,k}^{MBF-1^{st}Attempt} = \frac{1}{Buffer_i(t)} \tag{4.21}$$

According to this metric, a simulation using the developed simulator described in Chapter 5 and the simulation parameters presented in section 6.3 was made. The term "global QoE" refers to the overall QoE that a user has with all the streaming session (duration of video downloaded plus initial buffering delay and re-buffering). The result in terms of the number of satisfied users with global QoE ≥ 3 is presented in Figure 4.3. For each presented curve, at least the S_{min} simulations calculated using the Monte Carlo method presented in section 6.2, to guarantee with 95% of certainty that the real solutions do not differ by more than 1% from the presented values, were made for the points whose x-coordinate (# Total connected users) is between 5 and the x-coordinate of the first point with a percentage less than 70% (inclusive). The plotted points' x-coordinates are multiples of 5 users from 5 to 100. The percentages of users with global QoE ≥ 3 is the average of the values obtained in the simulations.

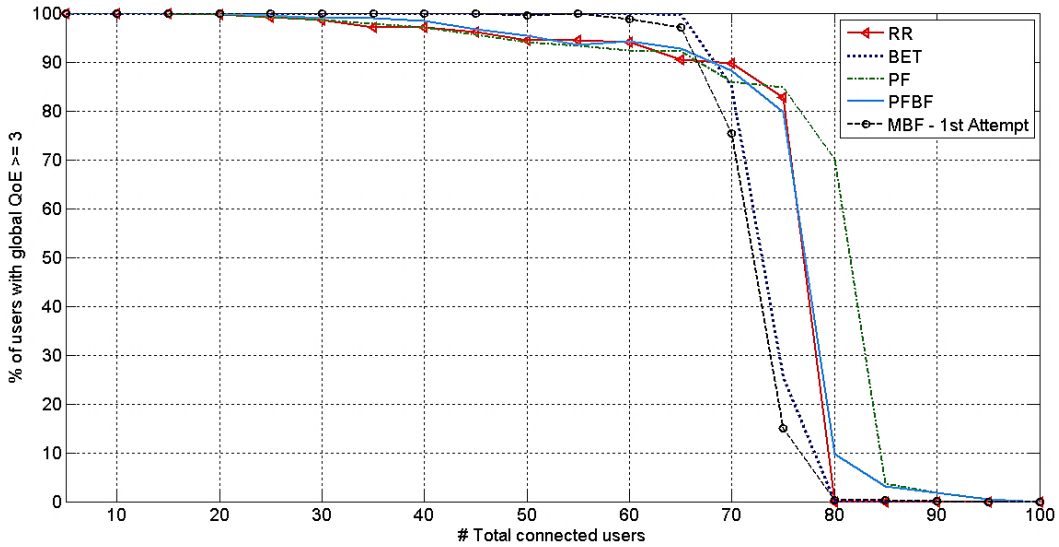

Figure 4.3 - 1^{st} Attempt - Percentage of satisfied users as a function of connected users streaming video.

The performance of the algorithm with a very simple metric had already a positive aspect. For a total number of connected users smaller than 65, MBF – 1^{st}Attempt guarantees a global QoE > 3 for a higher percentage of users than RR, PF and PFBF.

However, this metric does not consider directly the current channel condition of the user, which is a waste of useful information. For example, it would be inefficient to allocate resources to a user who

has momentarily a poor channel condition just because he has the lowest buffer level among the connected users. Although he has the lowest buffer level, he may have several seconds of video stored in the buffer, allowing the user to play the video smoothly during some seconds even if he is not served while he has a poor channel condition. Why not take advantage of the reported CQIs and serve the user with the second lowest buffer but that has a great CQI? So, the idea would be to have a numerator that somehow weighted the channel conditions. The better the channel conditions, the higher the metric. That could be the achievable throughput, the numerator that Proportional Fair uses in its metric or simply the CQI. However, a metric that divides two variables that are totally different does not work, normally. The impact of the numerator and the denominator on the metric would be totally different. Furthermore, sending many bits (by having the highest throughput) may not mean to greatly increase the buffer level. If the user is requesting for a very good video quality segment, the buffer level increases very slowly. For instance, consider several users who have the same buffer level, a low one. To avoid stalls, it is more efficient to allocate the resource blocks to the users whose channel conditions and requested bitrates allow them to store a higher number of milliseconds in his buffer, avoiding at least these users. Therefore, to have into account both the first idea (take advantage of the user channel conditions) and the second one (take advantage of knowing the users' requested video bitrates), it came intuitively that maybe it would be a good idea to not only consider the current buffer level (denominator) but also to measure how much the buffer is increasing if the resources blocks are allocated to the user. Hence, the numerator would be the quotient between achievable throughput and requested video bitrate. This quotient is in fact the number of milliseconds that is possible to send to the user, given his channel condition and requested video bitrate. The metric would be the one presented in equation (4.22):

$$m_{i,k}^{MBF-2^{nd}Attempt} = \frac{\frac{U_{throughput}^{i}(t)}{U_{RequestedVideo}^{ji}(t)}}{Buffer_i(t)} \tag{4.22}$$

where $U_{RequestedVideo}^{i}(t)$ is the bitrate of the video that is being downloaded at time t by user i and $U_{throughput}^{i}(t)$ is the achievable throughput calculated using the equation (5.1). The same simulation described above was run again and the results are presented in Figure 4.4.

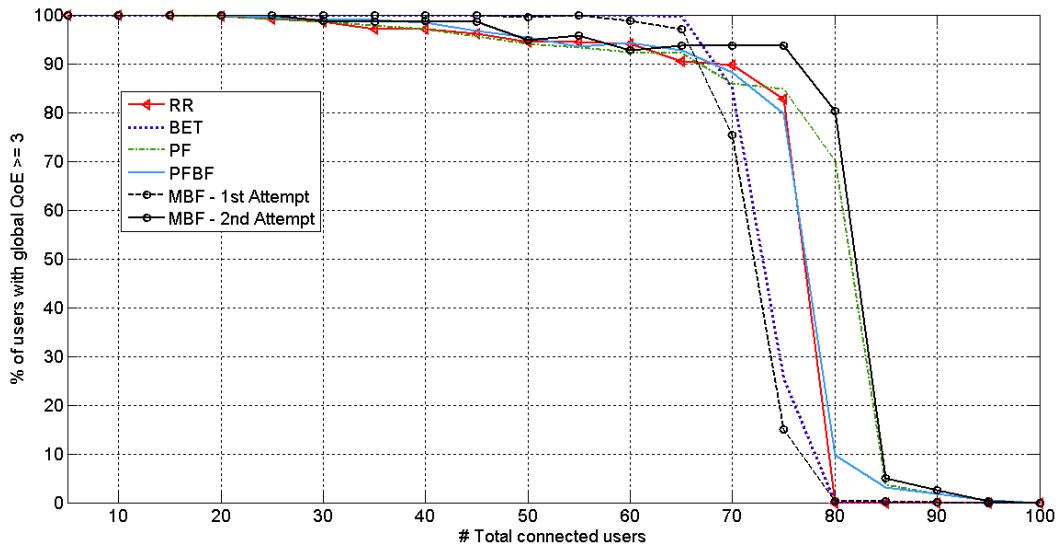

Figure 4.4 - 2^{nd} Attempt - Percentage of satisfied users as a function of connected users streaming video.

The result was already satisfying because for example to achieve a percentage of users with global QoE ≥ 3 equal to 90 %, MBF - 2^{nd}Attempt supports clearly a higher total number of connected users. However, it is necessary to understand what are the negative consequences of using MBF - 2^{nd}Attempt with respect to the MBF - 1^{st}Attempt, because some users may suffer from starvation and have a very low global QoE. In this context, the percentage of users with global QoE < 2 as a function of the total number of users streaming video is presented in Figure 4.5. According to Figure 4.4 and Figure 4.5, the MBF - 2^{nd}Attempt's gains with respect to the MBF - 1^{st}Attempt are achieved by slightly increasing the percentage of users with global QoE < 2. MBF – 1^{st}Attempt has thus a high level of fairness regarding the users' QoE.

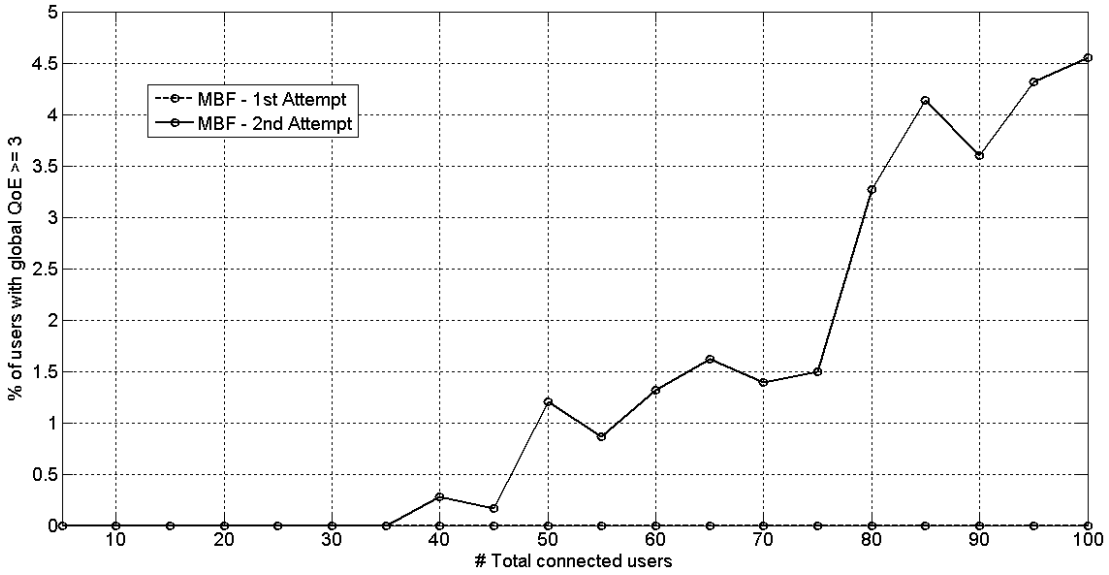

Figure 4.5 - Difference between the two MBF attempts regarding the number of users with global QoE < 2.

After a small number of experiments, the final MBF metric is given in equation (4.23) and its assessment performance is presented in section 5.5. In fact, the adjustable parameter α weights the algorithm behavior between scheduling resources according to the metric of MBF - 1^{st}Attempt and the metric of MBF - 2^{nd}Attempt.

4.5.2 A new algorithm: Maximum Buffer Filling

The proposed allocation algorithm in this thesis takes into account the following metrics reported by DASH clients: buffer level and the bitrate of the requested video. It considers also the channel conditions of the users through the CQIs reported which are used to calculate the achievable throughputs. At time t, it is served the user who has the highest metric:

$$m_{i,k}^{MBF} = \begin{cases} \dfrac{\Delta U_{buffer}^{i}(t)}{Buffer_i(t)}, & if\ \forall i\ Buffer_i(t) > \alpha \\ \dfrac{1}{Buffer_i(t)}, & otherwise \end{cases} \tag{4.23}$$

$$\Delta U_{buffer}^{i}(t) = \frac{U_{throughput}^{i}(t)}{U_{RequestedVideo}^{i}(t)} - 1 \tag{4.24}$$

where $Buffer_j(t)$ is the number of milliseconds of video stored in the i^{th} user's buffer and $U^i_{RequestedVideo}(t)$ is the bitrate of the video that is being downloaded at time t by the i^{th} user and $U^i_{throughput}(t)$ is the achievable throughput given their channel conditions, calculated using the equation (5.1). $\frac{U^i_{throughput}(t)}{U^i_{RequestedVideo}(t)}$ is the predicted number of milliseconds that will be possible to transmit to the i^{th} user if the resource blocks are allocated to him. $\Delta U^i_{buffer}(t)$ is the buffer variation in an interval of 1 ms. It is equal to the number of milliseconds sent to the user less 1 ms of video that the user plays every 1 millisecond spent. The term "-1" does not influence the way the resources are scheduled. However, for further modifications in the metric the term may be important, for instance if multiplied by a term that is different for the connected users.

The term $\Delta U^i_{buffer}(t)$ is responsible for increasing the network efficiency by increasing the probability of serving the user with better channel capacity. The user with better CQI has a higher achievable throughput, $U^i_{throughput}(t)$. If this term was used solely without the denominator $U^i_{RequestedVideo}(t)$, it would be maximizing the number of bits transmitted but not the number of milliseconds transmitted. By dividing the achievable throughput $U^i_{throughput}(t)$ by $U^i_{RequestedVideo}(t)$, the term $\Delta U^i_{buffer}(t)$ maximizes the sum of the number of milliseconds of video stored in the users' buffers, each TTI. The term $U^i_{RequestedVideo}(t)$ increases also the fairness among the users in terms of video quality served. The lower the quality that is being served to the i^{th} user, the higher the probability of being served. The term $Buffer_i(t)$ normalizes the number of milliseconds possible to transmit, using the number of milliseconds that are stored in the client's buffer, providing fairness in terms of buffer occupancy and trying also to avoid re-buffering events or buffer overflows. The lower the number of milliseconds in the buffer, the higher the metric, and thus the probability of the user of being served. If buffer is almost full, the metric value is small.

If there are one or more users with a buffer level $Buffer_i(t) < \alpha$ (and if the user is not in the initial buffering state), the scheduler serves the user who has the lowest buffer level, using a metric equal to $\frac{1}{Buffer_i(t)}$. This is an emergency allocation to avoid a stall. When the user buffer level drops bellow α, he has the lowest buffer and must be served. If all buffer levels are above the threshold α, the served user is the one who has the chance to fill his buffer with a bigger quantity of milliseconds, normalized to his current buffer level. The MBF pseudocode is the following:

```
1: for each TTI do
2:     emergency_state = false
3:     for each k^th RB do
4:         j* = 0
5:         m_{j*,k}^{MBF} = 0
3:         for each i^th user do
4:             if Buffer_i(t) < α then
6:                 emergency_state = true
```

7: **end if**

8: **end for**

9: **for each** i^{th} user **do**

10: **if** emergency_state == false **then**

11: $m_{i,k}^{MBF} = \frac{\Delta U_{buffer}^{i}(t)}{Buffer_i(t)}$

12: **else**

13: $m_{i,k}^{MBF} = \frac{1}{Buffer_i(t)}$

14: **end if**

15: **if** $m_{i,k}^{MBF} > m_{j^*,k}^{MBF}$ **then**

16: $j^* = i$

17: **end if**

18: **end for**

19: $NbrRB_{j^*} = NbrRB_{j^*} + 1$

20: **end for**

21: end for