Research Article

QoE-Aware Scheduling Algorithm for Adaptive HTTP Video Delivery in Wireless Networks

Frederico Rodrigues, Ivo Sousa, Maria Paula Queluz, and António Rodrigues

Instituto de Telecomunicações, Instituto Superior Técnico, University of Lisbon, Portugal

Correspondence should be addressed to Ivo Sousa; ivo.sousa@lx.it.pt

Received 13 April 2018; Revised 20 July 2018; Accepted 13 August 2018; Published 2 September 2018

Academic Editor: Mauro Femminella

Copyright © 2018 Frederico Rodrigues et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the last years, the video content consumed by mobile users has increased exponentially. Since mobile network capacity cannot be increased as fast as required, it is crucial to develop intelligent schedulers that allocate radio resources very efficiently and are able to provide a high Quality of Experience (QoE) to most of the users. This paper proposes a new and effective scheduling solution—the Maximum Buffer Filling (MBF) algorithm—which aims to increase the number of satisfied users in video streaming services provided by wireless networks. The MBF algorithm uses the current buffer level at the client side and the radio channel conditions, which are reported to the network by the client, as well as the bitrate of the requested video segment. The proposed scheduling strategy can also fulfill different satisfaction criteria, since it can be tuned to maximize the number of users with high QoE levels or to minimize the number of users with low QoE levels. A simulation framework was developed, considering a Long Term Evolution (LTE) scenario, in order to assess the performance of the proposed scheduling scheme and to compare it with other well-known scheduling solutions. The results show the superior performance achieved by the proposed technique, in terms of the number of satisfied and unsatisfied users.

1. Introduction

Nowadays, video streaming is the most popular service on the Internet and it is foreseen that three-fourths of the worldwide mobile data traffic in 2020 will be video [1]. At the same time, people are influenced by the Quality of Experience (QoE) they have when using a service: for instance, if this service does not provide them the QoE they are expecting, they might stop paying for it and search for a new service provider. Thus, it is of great interest for the operators to provide a service that satisfies a high number of users. However, a wireless network environment is characterized by a time-varying behavior of the received signal strength and/or interference level and, as a consequence, of the available throughput. This may severely impact the experienced quality in video streaming applications, even with the adoption of a buffer at the receiver side; in fact, an insufficient throughput may cause the need for rebuffering, a streaming state activated when the playback buffer becomes empty and that leads to a playout stall, which is usually very annoying for the users [2, 3]. Other undesired effects may also be potentiated by the variability of the radio channel (e.g., long initial playout delays), all of which degrade the QoE. Furthermore, the continuous growth of consumed mobile multimedia content may lead to network congestion. Accordingly, the design of intelligent and efficient scheduling algorithms becomes crucial, in order to maintain a high QoE level to all users.

This paper addresses QoE-oriented techniques to be adopted by wireless schedulers, in order to maximize the number of satisfied users. The main contribution is the proposal of a QoE-aware scheduling algorithm for adaptive HyperText Transfer Protocol (HTTP) video delivery in wireless networks, such as Long Term Evolution (LTE) and Worldwide Interoperability for Microwave Access (WiMAX) networks. The proposed strategy makes use of the clients’ buffer level, which is reported by the mobile clients that implement the Dynamic Adaptive Streaming over HTTP (DASH) specifications [4, 5], and also takes into account the predicted buffer filling of each client (i.e., how much the buffer level would increase if resources are allocated to the respective user), which is estimated from the requested video segments and radio channel conditions. Moreover, the
proposed scheduling scheme provides an adjustable trade-off between efficiency and fairness, i.e., it allocates more resources to users experiencing higher throughputs but tries to ensure, at the same time and to an extent based on the service provider preferences, that the remaining users do not experience rebuffering events. The performance of the proposed scheduling algorithm is assessed through simulation and using, as assessment metrics, the percentage of satisfied and unsatisfied users for different cell loads of an LTE network. However, it is worth mentioning that the new scheduling strategy can be used by other radio access networks, provided that the resource allocation architecture is similar to the LTE one (e.g., the WiMAX network). Some of the most well-known scheduling solutions are also considered, thus enabling not only to assess their performance in terms of the number of users that can be served with a certain QoE, but also to reveal the benefits of the proposed technique.

The rest of the paper is organized as follows. Section 2 reviews the related work. In Section 3, the details of the proposed QoE-aware scheduling algorithm are presented. Section 4 describes the features of the developed simulation framework. In Section 5, the assessment results are provided. Finally, Section 6 concludes the paper. To facilitate the readability, the Notations shown at the end of the paper summarize the notations used herein.

2. Related Work

This section addresses the related work by providing (i) an overview of the most relevant concepts associated with the DASH standard; (ii) the definition of QoE and its contextualization in video streaming scenarios; (iii) the fundamentals of LTE resource allocation; and (iv) a review on the related literature concerning scheduling algorithms.

2.1. DASH Overview. In order to cope with variable Internet receiving conditions (e.g., those caused by radio channel quality variations), the Motion Picture Experts Group (MPEG) developed the DASH standard [4], which allows the transmission bitrate to be adapted by the client, according to the current network conditions or mobile device characteristics. As part of their collaboration with MPEG, the 3rd Generation Partnership Project (3GPP) has adopted DASH, with specific codecs and operating modes, for use over wireless networks, resulting in the 3GP-DASH standard [5].

At the DASH server side, the same media content has several representations, each one with a different bitrate (and a different subjective quality), and each representation is divided into segments, which can be independently decoded by the client. The characteristics of each segment—such as server location (URL), the associated bitrate, or video resolution—and the temporal and structural relationships between segments are defined in a manifest file denoted as Media Presentation Description (MPD), which is sent at the start of the transmission.

The DASH client estimates the channel throughput, selects an appropriate encoded representation, and starts downloading the content by fetching the respective segments using the HTTP protocol; during the streaming session, the client can seamlessly switch between the different media representations, in order to cope with throughput fluctuations or according to mobile device characteristics; an example of DASH segment requests during a video streaming session is illustrated in Figure 1. It is worth mentioning that the DASH standard only defines how the media content is represented and how the respective metadata is handled. Hence, the delivery of the metadata file and media encoding formats containing the segments are not specified, nor the client’s action for fetching, adapting, and playing the content [6].

The DASH specifications also include the possibility of the server to inquire some client’s metrics, which are useful for QoE monitoring (cf. Section 2.2). More specifically, an MPD may contain the list of DASH metrics to be measured and reported by the client, the time interval, and the granularity for the measurements, as well as the scheme according to which the metric reporting is desired.

2.2. Understanding QoE. 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” [7]. Therefore, two different persons will probably perceive a video streaming experience in different ways, which is mainly due to the following influence factors:

(i) Human factors: user characteristics that affect QoE (e.g., gender, age, mood, and visual and auditory acuity);

(ii) System factors: factors related to the transmission system (e.g., codec, packet loss rate, delay, and screen resolution);

(iii) Context factors: external factors that affect QoE (e.g., time of the day, day of the week, and service type).

Clearly, QoE can be very subjective; accordingly, the most reliable and accurate method for assessing it is to perform subjective tests, where the perceived QoE is usually expressed in a Mean Opinion Score (MOS) scale ranging from 1 (“Bad”) to 5 (“Excellent”) [8] (cf. Table 1). However, obtaining MOS scores with subjective tests is very expensive and time-consuming; furthermore, it is impractical in certain types of services, such as live streaming. For this reason, objective measurement techniques have been developed, in order to obtain a valid QoE prediction regarding video streaming, while avoiding the need to perform subjective tests; related
surveys can be found in [9–12]. Most of the proposed objective assessment methods for adaptive video streaming (e.g., [13–15]) make use of part (or all) of the following quantitative metrics, which are obtained from information provided by the client (the metrics marked with an asterisk denote DASH standardized metrics [4, 5]):

(i) Representation switch event*: a list of representation switches that took place during the measurement interval;

(ii) Average throughput*: the average throughput that is observed by the client during the measurement interval;

(iii) Initial playout delay*: the period between the starting time of loading a video and playing it;

(iv) Buffer level*: a list of buffer occupancy levels carried out during the measurement interval;

(v) Play list*: a list of playback periods in the measurement interval, where each playback period is the time interval between a user action (new playout request, resume playout from pause, or user-requested quality change) and the end of playback or a failure that stops playback;

(vi) Mean rebuffering duration: the average time that the playback was paused during a rebuffering event;

(vii) Rebuffering frequency: the frequency of the rebuffering events;

(viii) Mean video quality: the average video quality served to a user during playout;

(ix) Video quality changes amplitude: the extent of each video quality change, specifying whether it is a change from a higher to a lower quality or vice versa;

(x) Video quality changes frequency: the frequency of the video quality changes during playout.

These quantitative metrics are usually incorporated into parametric models that establish a mapping between the objective quality metrics and a subjective score; the parameters of these models are tuned based on the results of subjective tests, e.g., using ordinary least squares or maximum likelihood estimation methods.

2.3. Background on LTE Resource Allocation. As previously mentioned, in order to assess the proposed QoE-aware scheduling technique, an LTE network was considered, so its resource allocation architecture is briefly described in this section.

<table>
<thead>
<tr>
<th>MOS</th>
<th>Perceived quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Excellent</td>
</tr>
<tr>
<td>4</td>
<td>Good</td>
</tr>
<tr>
<td>3</td>
<td>Fair</td>
</tr>
<tr>
<td>2</td>
<td>Poor</td>
</tr>
<tr>
<td>1</td>
<td>Bad</td>
</tr>
</tbody>
</table>

Table 1: Perceived quality score within MOS subjective tests.

<table>
<thead>
<tr>
<th>Bandwidth [MHz]</th>
<th>RBs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>15</td>
<td>75</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 2: Number of RBs in the frequency domain for each bandwidth.

Within the LTE downlink physical layer, radio resources are allocated into the time/frequency domain. As shown in Figure 2, the total time is split into 10 ms frames, each one composed of 10 consecutive 1 ms subframes denoted as Transmission Time Intervals (TTI). Furthermore, each TTI comprises 2 time slots, with a length of 0.5 ms, and each time slot corresponds to 7 Orthogonal Frequency-Division Multiplexing (OFDM) symbols (normal cyclic prefix) or 6 symbols (extended cyclic prefix) [16]. In the frequency domain, the total bandwidth is divided into subchannels of 180 kHz, each one comprising 12×15 kHz subcarriers (or, less often, 24×7.5 kHz subcarriers) [16]. A time/frequency radio resource spanning over one time slot and over one subchannel is denoted as Resource Block (RB). Therefore, each RB accounts for 12×7 OFDM symbols in the vast majority of cases. The bandwidths defined by the standard determine how many RBs are available in each situation regarding the frequency domain [16] (cf. Table 2).

Although RBs are defined over one subchannel and over one time slot, the basic scheduling unit in LTE corresponds to the intersection of one subchannel with one subframe/TTI (i.e., two consecutive time slots); this means that the scheduling unit comprises two consecutive (in the time domain) RBs and that the scheduler makes resource allocation decisions every 1 ms [17]. For instance, the 20 MHz bandwidth has 200 RB available per subframe; however, there are only 100 addressable locations of 2 RBs each. Accordingly, for each User Equipment (UE), a minimum of two consecutive RBs (one RB pair, denoted from now on as RBP) can be allocated per TTI, and a maximum of 100 UEs can be served per TTI.
It is also important to mention that, in a LTE network, all UEs report periodically a parameter to the base station—the Channel Quality Indicator (CQI)—which keeps the latter informed about the channel conditions of each UE. The efficiency associated with each CQI value is presented in Table 3 for reporting CQI based on QPSK, 16QAM, and 64QAM modulations [16]. In the frequency domain, the granularity of a CQI report ranges from wideband to per RB reporting, whereas its periodicity varies between 1 ms and 160 ms [16].

Finally, in resource allocation architectures like or similar to the LTE one, the resource allocation is usually performed by comparing metrics on a scheduling unit basis (in the LTE case, the scheduling unit corresponds to a RBP); in each scheduling period (or TTI), there are \( K \) scheduling units to be allocated; in the \( n^{th} \) scheduling period, the scheduling unit \( k \in \{1, \ldots, K\} \) is assigned to the \( j^{th} \) user if the respective metric \( m_{nk}[j] \) is the biggest one among all users, i.e.,

\[
j = \arg \max_i \{ m_{nk}[i] \}. \tag{1}\]

These metrics can be regarded as the priority of each user on a particular scheduling unit; according to the intended system performance, the computation of these metrics takes into account one or more factors, e.g., radio channel conditions, resource allocation history, or buffer state.

### 2.4. Related Work on Scheduling Algorithms

Wireless resources scheduling aims at establishing a trade-off between satisfying the customers’ expectations and handling the limited network capacity. According to the intended system performance, scheduling algorithms not only allocate physical radio resources among users, but also assign the users’ serving order, in order to fulfill some service requirements, e.g., fairness.

A large number of scheduling strategies have been proposed in the last years, but most of them do not take the QoE into account, in contrast to the solution proposed in this paper. Nevertheless, some of these strategies—as the Round-Robin (RR) [18], the Blind Equal Throughput (BET) [18], and the Proportional Fair (PF) [18] strategies—deserve to be mentioned because they are widely used as benchmarks in the literature. The RR approach is one of the simplest scheduling schemes, which allocates an equal share of resources to each user in a Round-Robin manner. Hence, RR is fair in terms of channel occupancy time and can be a good option either if the radio channel conditions are not known, or if these conditions are similar and do not vary much in time for all users; otherwise, some users may suffer from starvation. The BET scheduling algorithm can be regarded as the fairest in terms of throughput, as it prioritizes the user that has experienced the lowest average throughput. Thus, BET is able to provide similar QoE to all users, namely, if these are using the same service and in the same conditions (e.g., if the same rate adaptation algorithm is adopted by all of them). On the other hand, a small number of users with very bad channel conditions can greatly reduce the overall QoE (i.e., the average QoE of all connected users) if BET is enforced, as the respective scheduler will allocate a lot of resources to these users, thus not exploiting the higher throughput experienced by the remaining users. A scheduler that implements the PF strategy tries to avoid the BET’s problem previously mentioned, namely by providing a compromise between fairness and spectral efficiency. In this case, the goal is to maximize the total network throughput by allocating more resources to users experiencing better channel conditions (hence, they can achieve higher throughputs), while ensuring, at the same time, that users with poor conditions will always be served after some time. More elaborated strategies, such as the Modified Largest Weighted Delay First (M-LWDF) scheme [19] and the Exponential/PF (EXP/PF) approach [20], aim at providing a balanced trade-off among fairness, spectral efficiency, and service requirements, such as delay. For more details on these and other scheduling techniques that do not take the QoE into account, please refer to [18].

With the advent of the QoE concept, many authors have proposed scheduling schemes that take QoE metrics into account, in contrast to the solution proposed in this paper. Nevertheless, some of these strategies—as the Round-Robin (RR) [18], the Blind Equal Throughput (BET) [18], and the Proportional Fair (PF) [18] strategies—deserve to be mentioned because they are widely used as benchmarks in the literature. The RR approach is one of the simplest scheduling schemes, which allocates an equal share of resources to each user in a Round-Robin manner. Hence, RR is fair in terms of channel occupancy time and can be a good option either if the radio channel conditions are not known, or if these conditions are similar and do not vary much in time for all users; otherwise, some users may suffer from starvation. The BET scheduling algorithm can be regarded as the fairest in terms of throughput, as it prioritizes the user that has experienced the lowest average throughput. Thus, BET is able to provide similar QoE to all users, namely, if these are using the same service and in the same conditions (e.g., if the same rate adaptation algorithm is adopted by all of them).

\[
\begin{array}{|c|c|c|}
\hline
\text{CQI index} & \text{Modulation} & \text{Efficiency [bits/symbol]} \\
\hline
1 & \text{QPSK} & 0.1523 \\
2 & \text{QPSK} & 0.2344 \\
3 & \text{QPSK} & 0.3770 \\
4 & \text{QPSK} & 0.6016 \\
5 & \text{QPSK} & 0.8770 \\
6 & \text{QPSK} & 1.1758 \\
7 & \text{16QAM} & 1.4766 \\
8 & \text{16QAM} & 1.9141 \\
9 & \text{16QAM} & 2.4063 \\
10 & \text{64QAM} & 2.7305 \\
11 & \text{64QAM} & 3.3223 \\
12 & \text{64QAM} & 3.9023 \\
13 & \text{64QAM} & 4.5234 \\
14 & \text{64QAM} & 5.1152 \\
15 & \text{64QAM} & 5.5347 \\
\hline
\end{array}
\]
video segment) of not only some, but all users. This global approach is also followed by other authors; for instance, in [23], the rebuffering events are adopted as QoE inputs by the proposed scheduler, namely, by taking into account not only current playback interruptions, but also past service quality degradation. Subsequently, the users experiencing the lowest cumulative QoE are scheduled with the highest priority, thus seeking an equitable QoE. Some optimization problems and respective solutions have also been devised with respect to QoE-oriented scheduling techniques, such as the ones presented in [24, 25]. The solutions therein make use of the reported buffer level and either try to prevent the violation of a certain rebuffering constraint [24] or aim at maximizing the minimum buffered video playout time among all clients [25].

Most of the previous scheduling strategies do not exploit the higher throughputs that can be achieved by some users, namely those that are experiencing better channel conditions—by better serving these users, the overall QoE could be improved as long as users with poor channel conditions would still keep an acceptable QoE. For this reason, the scheduling technique devised in this work also attempts to take advantage of the better channel conditions experienced by some users, besides taking into account some QoE relevant metrics sent by the clients, with the final goal of maximizing both the amount of satisfied users connected to a base station (i.e., users that have, at least, a certain QoE level) and the overall QoE. Some authors presented scheduling strategies with similar purposes [26, 27], but corresponding to enhancements made to a PF scheduler (whereas the solution proposed in this paper is not based on the PF approach); in other words, these strategies act like a PF scheduler until some user's reported buffer level is below a certain threshold, thus forcing (albeit with a soft transition) the prioritization of this user. In addition, the Proportional Fair with Barrier for Frames (PFBF) algorithm introduced in [26] allocates resources taking into account not only the buffer level, but also the rebuffering time percentage (i.e., the percentage of the entire streaming time that is spent on rebuffering), with the intention of including fairness in terms of this last QoE metric. In [27], the users are prioritized not only according to the absolute amount of information on the player's buffer, but also considering the variation rate of the buffer level, in order to constrain the rebuffering probability without compromising the video quality.

The solutions proposed in [28–30] follow a different approach than the scheduling strategy introduced herein, namely by considering the use of a proxy that performs joint optimization of resource allocation and of the client streaming rate adaptation within DASH systems. In particular, the radio channel conditions, the multiple bitrate encodings available on the server, and the buffer status of the clients are taken into account by the proxy with two purposes: first, to prioritize a stream when the scheduler detects that the corresponding radio channel is not in good conditions; secondly, to address the possibility of modifying a client's request to a representation with a lower bitrate, in order to avoid a higher QoE deterioration due to an imminent stall. Although this proxy-based approach is able to decrease significantly the occurrence of rebuffering events and to improve the decoded video quality of all users, a client, after requesting a given DASH segment, may not be willing to accept other DASH segment(s) as an alternative—in fact, even though the DASH standard incorporates the Server and Network Assisted DASH (SAND) functionality, in which clients can inform the server that they are ready to accept other representations described in the MPD than the one they initially requested, a 3GP-DASH client shall not include alternative segments unless it is ready to receive them and be able to play them [5].

The scheduling techniques addressed so far consider the case where each base station represents an independent scheduler. Nevertheless, some authors have proposed solutions that adopt a centralized network controller, which is able to enhance QoE of users that can connect to more than one base station. More specifically, the authors of [31] designed a QoE-aware scheduling system along with call admission and handoff procedures, which provide a seamless multimedia streaming; this is accomplished by ensuring that previously connected users keep a satisfactory level of buffered video playout time among neighboring base stations (through scheduling and handoff operations), as well as these users' buffer levels are used to determine the admission of new users to the network. In [32], a system is proposed to support a large number of video streaming sessions in a dense wireless network; in this case, the buffer level is taken into account in order to not only maintain a desired video quality, but also to fairly allocate the available remaining capacity, thus providing a high utilization of the network and eventually allowing users to receive a video with a higher quality. In spite of the fact that these centralized approaches enable us to greatly improve the overall QoE in the network, they accomplish this goal by exploiting some specificities of the multicell scenario; e.g., if several different base stations are able to serve a specific user simultaneously, a handoff can be triggered from time to time as to select the base station that enhances this user's QoE. Therefore, a single base station scenario (which is the one addressed by the solution proposed in this paper) can somewhat be regarded as more challenging, since it may lead to a situation where the base station has to deal with a large number of users that were admitted and that must be served only by this base station.

### 3. Proposed QoE-Aware Scheduling Algorithm

The system model considered in this work is depicted in Figure 3. It consists in several DASH video streaming sessions, established between mobile terminals and a DASH based video server, over a wireless network (such as the LTE network), considering a single base station scenario; the segments requested by the DASH clients are delivered by the server to a scheduler, which then has to perform resource allocation based on a scheduling algorithm, in order to satisfy the users' expectations. Besides knowing some information about the video segment that is being streamed to each user, namely, the respective bitrate, the scheduler has also knowledge about the radio channel conditions that each user is experiencing (e.g., through the CQIs reported to the base station in the LTE case) and about the current buffer level
at the client side (recall that this is a QoE metric that clients implementing the DASH specification can report to the network). It is worth mentioning that, since wireless resources scheduling is one of the baseband processing functionalities of a base station, the scheduler’s location depends on the network architecture, i.e., the scheduler can be colocated, in one unit, with the radio functionalities, such as the case of traditional base stations, or it can be within a centralized Baseband Unit (BBU) pool, such as the case of Cloud Radio Access Networks (C-RAN) [33].

As mentioned in Section 1, a rebuffering event is usually very annoying for the users and its prevention should be the first priority for the scheduler; if a stall happens, then the rebuffering period must be short. Accordingly, a first solution for the proposed scheduler, denoted from now on as Maximum Buffer Filling (MBF), could be to simply allocate RBs to the users with the lowest buffer levels, i.e., this first approach is analogous to the BET one, but now, instead of the throughput, the buffer level is adopted as scheduling input. The respective resource allocation metric would then be given by

\[ m_{nk}^{MBF}[i](1^{st}\text{approach}) = \frac{1}{U_n[i]}, \]  

(2)

where \( U_n[i] \) stands for the amount of video playout time stored in the \( i^{th} \) user’s buffer at the beginning of the \( n^{th} \) scheduling period. However, metric (2) does not consider the current channel conditions experienced by the user, which is a waste of useful information. For instance, it would be inefficient to allocate resources to a user that is momentarily experiencing poor channel conditions, just because this user has the lowest buffer level among all connected users: several seconds of video may still be stored in the respective buffer, allowing the user to play the video smoothly during that time, even if the user is not served. Therefore, why not take advantage of the reported radio channel conditions and serve, for example, the user with the second lowest buffer but that can achieve a higher throughput? On the other hand, sending many bits to the user that is experiencing a higher throughput may not mean that the video playout time will increase greatly in the respective buffer; e.g., if this user is requesting a very good video quality segment, then the respective buffer level may increase very slowly. Accordingly, it would be more efficient to allocate resource to the users whose channel conditions and requested bitrates allow a higher amount of video playout time to be stored in their buffers, thus helping the prevention of playout stalls.

In order to take advantage of the reported channel conditions and of the known users’ requested video bitrates, the QoE-aware scheduler should not only consider the current buffer level of the users (denominator of (2)), but it should also estimate how much the buffer level would increase if the scheduling unit is allocated to the respective user, yielding the metric

\[ m_{nk}^{MBF}[i](2^{nd}\text{approach}) = \frac{\Delta U_{n,k}[i]}{U_n[i]}, \]  

(3)

where \( \Delta U_{n,k}[i] \) corresponds to the predicted buffer level variation of the \( i^{th} \) user if the \((n,k)^{th}\) scheduling unit is allocated to this user; the value of \( \Delta U_{n,k}[i] \) can be estimated by

\[ \Delta U_{n,k}[i] = \frac{\overline{R}_{nk}[i] \times T_{SU}}{V_n[i]}, \]  

(4)

where \( V_n[i] \) denotes the bitrate of the video that is being downloaded by the \( i^{th} \) user during the \( n^{th} \) scheduling period, \( \overline{R}_{nk}[i] \) represents the estimated achievable throughput for the \((n,k)^{th}\) scheduling unit given the channel conditions of the \( i^{th} \) user, and \( T_{SU} \) stands for the scheduling period length. Considering the LTE network, for which a scheduling unit comprises two consecutive RBs (hence, \( T_{SU} = 1 \text{ ms}, \) cf. Section 2.3), the achievable throughput per scheduling unit can be estimated as

\[ \overline{R}_{nk}[i] = 2 \times \Omega \times E_{nk}(CQI[i]) \times G \times (1 - \zeta), \]  

(5)

where \( \Omega \) corresponds to the number of OFDM symbols that each RB encompasses, \( E_{nk}(CQI[i]) \) denotes the \( k^{th} \) RBP efficiency with respect to the \( n^{th} \) scheduling period, which depends on the respective last CQI reported by the \( i^{th} \) user (according to the CQI frequency granularity and periodicity), \( CQI[i], \) \( G \) stands for the gain obtained by using the Multiple Input Multiple Output (MIMO) technique, and \( \zeta \) represents the percentage of overhead bits.

Usually, the time interval between each report of the client’s buffer level is higher than the scheduling period length; in this case, the buffer level can be estimated by the scheduler in the meantime (i.e., while the updated real buffer level is not reported) by

\[ U_{n+1}[i] = \max \left\{ U_n[i] + \sum_{j=1}^{K} \left( \Delta U_{n,j}^*[i] - T_{SU}; 0 \right) \right\}, \]  

(6)

where \( \Delta U_{n,j}^*[i] \) represents an estimate of how much the buffer filling of the \( i^{th} \) user will increase with respect to the \( j^{th} \) allocated scheduling unit in the \( n^{th} \) scheduling period, i.e.,
Wireless Communications and Mobile Computing

Figure 4: MBF scheduling algorithm flowchart.

\[ \Delta U^*_{n,j} [i] = \begin{cases} \Delta U_{n,j} [i], & \text{if scheduling unit } j \text{ of scheduling period } n \text{ was allocated to user } i \\ 0, & \text{otherwise.} \end{cases} \] (7)

By considering the term \( \Delta U_{n,k} [i] \) in (3), namely, the predicted amount of video playout time that will be possible to transmit to the each user, the network efficiency is improved due to a higher probability of serving the users experiencing higher throughputs. On the other hand, the use of the term \( V_n [i] \) in (4) also increases 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. Within the approach of (3), the term \( U_n [i] \) acts like a normalization factor regarding the amount of video playout time that is possible to transmit, hence not only providing fairness in terms of buffer occupancy, but also trying to avoid rebuffering events.

Nevertheless, metric (3) may lead to some users suffering from starvation (which originates playout stalls), mainly when they are experiencing a very low throughput for a considerable amount of time. For this reason, the final version of the proposed QoE-aware scheduling algorithm incorporates an emergency condition in the resource allocation metric:

\[ n^{MBF}_{n,k} [i] = \begin{cases} \frac{1}{U_n [i]}, & \text{if } \exists j: U_n [j] < \alpha \\ \frac{\Delta U_{n,k} [i]}{U_n [i]}, & \text{otherwise.} \end{cases} \] (8)

The parameter \( \alpha \) defines the emergency state threshold, i.e., if one or more users have a buffer level lower than \( \alpha \), then the scheduler serves the user which has the lowest buffer level, thus trying to avoid a rebuffering event. On the other hand, if all buffer levels are above the \( \alpha \) threshold, then the served user is the one which has the chance of having the respective buffer filled with the highest amount of video playout time, normalized by the current buffer level. The previously described procedure is summarized in the flowchart depicted in Figure 4. In addition, and provided that the buffer level thresholds typically implemented by DASH players are known by the service provider (namely, those which are related to the initial buffering and rebuffering events), the parameter \( \alpha \) may also be used to ensure that, after some time, a client will always (re)start playing the video: this fair behavior takes place when \( \alpha \) is greater than these buffer level thresholds, since in that case all users that are only loading the video will be prioritized by the MBF approach; on the contrary, if \( \alpha \) is lower than these buffer level thresholds, then some users (especially those that are experiencing poor channel conditions) might never fill their buffers up to the predefined point where the video is allowed to (re)start playing, i.e., some users might be sacrificed. Nevertheless, note that a lower \( \alpha \) value also enables that the nonsacrificed users not only have lower initial playout delays, but also they have higher possibilities of experiencing a stall-free experience; thus these users achieve a better QoE; on the other hand, although higher \( \alpha \) values ensure the previously mentioned (re)start playing of the video after some time for all users, it also may lead to a situation where a great amount of time is required for all users to fill the respective buffers, which means that this may deteriorate the QoE of all users, namely if these start experiencing frequent switching between stalls and playing the video just for a few seconds.

In a nutshell, the proposed MBF scheduling strategy is a noncomplex technique, easy to implement and suitable for the case where each base station represents an independent scheduler. It only requires information about the radio channel conditions, buffer status, and requested video segments. The MBF approach not only exploits the higher throughputs experienced by some users, thus aiming to improve the overall QoE of the network, but also has the goal of preventing playout stalls, in order to avoid sacrificing the users that are experiencing poor channel conditions. Accordingly, the
MBF algorithm establishes a trade-off between efficiency (regarding the increase of the overall QoE) and fairness (regarding the prevention of rebuffering events), which is not fixed and can be adjusted by simply varying the value of the emergency state threshold (parameter $\alpha$), thus allowing to meet different service provider preferences.

4. Simulation Setup

In order to assess the scheduling algorithm proposed in this work, a simulator was developed from scratch, using the Java programming language [34]; this program simulates a scenario comprising an LTE base station and several mobile users, which are accessing to a video server according to the DASH specifications (as illustrated in Figure 3). The program was designed in a modular form, in order to allow the incorporation of different scheduling techniques, rate adaptation algorithms (at the client side), and QoE assessment models. The general workflow of the simulator, which is depicted in Figure 5, is as follows: first, the simulation settings are read (such as the total simulation time, $T_{\text{max}}$, the number of connected users, $N_U$, and the number of RBPs available for allocation in each TTI, $K$); afterwards, in each iteration corresponding to a TTI, the behavior of each user's device is simulated; the client buffer level is updated, new video segments might be requested (according to the rate adaptation algorithm), as well as the buffer level and the CQI might be reported to the base station; subsequently (after simulating all users' devices), the behavior of the scheduler is simulated, i.e., the scheduling algorithm computes the respective metrics for all users and allocates the RBPs accordingly; next, the simulation moves to the next TTI and the previous process is repeated till the simulation reaches the predefined time interval $T_{\text{max}}$, after which the QoE of each user regarding the entire streaming session is assessed and the respective results are saved to a file.

4.1. User CQI and Throughput. The radio channel conditions are simulated through periodic wideband CQI reports sent by the users to the network side every 5 ms [35]. These CQIs were obtained through an independent simulation, using the SimuLTE [36] and the OMNeT++ [37] tools: an LTE urban macrocell scenario was considered, comprising a single base station and 200 mobile users moving around with a random trajectory; all the CQIs reported by the users to the base station, during a time interval of 3 minutes (the optimal length of a YouTube video [38]), were saved. The resulting CQI database was then used by the newly developed simulator; each simulated user adopts randomly (but without repetitions) one of the 200 radio channels conditions previously saved.

The throughput yielded by each allocated RBP is computed using (5); the most common case of $12 \times 7$ OFDM symbols per RB was used in the developed simulator, as well as the RB efficiency values presented in Table 3; moreover, a $1 \times 1$ MIMO configuration was considered, which yields a MIMO gain equal to one, and the percentage of overhead bits was set equal to 25% in the simulations [39]. It is also assumed an LTE configuration comprising a bandwidth of 20 MHz, resulting in 100 RBPs available for allocation in each TTI (cf. Table 2). A summary of the aforementioned LTE parameters adopted in the simulations is given in Table 4.
4.2. Rate Adaptation Algorithm. In order to perform rate adaptation at the client side, the QoE enhanced Adaptation Algorithm over DASH (QAAD) [40] was incorporated in the simulator; this algorithm is responsible for the video bitrate selection based on the available channel throughput estimated by the client.

4.2.1. Channel throughput Estimation at the Client Side. The available channel throughput at time $t$, denoted by $R_{est}(t)$, is estimated every $\theta$ seconds. Since the channel quality varies a long time, the estimated channel throughput is smoothed with a moving average:

$$R_{est}(t) = \omega \times R_{est}(t-\theta) + (1 - \omega) \times R_{sample},$$

(9)

where $\omega$ stands for the weight factor for the sampled channel throughput $R_{sample}$ ($0 < \omega < 1$); this $R_{sample}$ corresponds to the available channel throughput sampled in every $\theta$ seconds, which is computed as

$$R_{sample} = \frac{B}{\theta},$$

(10)

where $B$ denotes the amount of bits downloaded during $\theta$ seconds. For simulation purposes, the parameters $\omega$ and $\theta$ were set to 0.875 and 0.3 s, respectively, as suggested in [40].

4.2.2. Bitrate Selection. The quality of the next requested video segment, $l_{best}$, is selected by the client based on the estimated channel throughput, $R_{est}$, and current buffer occupancy, $U$; the respective algorithm pseudocode is given in Algorithm 1.

Initially, the algorithm tests if the current highest video segment quality that does not exceed the estimated channel throughput, $l_{best}$, is equal to the previously selected segment quality, $l_{prev}$; if this is true, then the quality of the requested segment is kept unchanged. On the other hand, if $l_{best} > l_{prev}$, then the quality of the next video segment is increased (by one quality level) iff the current buffer occupancy is larger than a certain marginal buffer length, $\mu$; otherwise, the quality of the requested segment does not change. Finally, if $l_{best} < l_{prev}$, then the quality of the next video segment may or may not decrease. In order to minimize the impact on the QoE, the algorithm tries to keep the next video quality comparable to the previous one. Consequently, the algorithm first determines the elapsed time, $t_{1,\sigma}$, to consume all the segments in the playback buffer and computes the expected number of segments, $n_t$, to be downloaded during this time; the parameters $\sigma$, $\tau$, and $b(l)$ represent the predefined minimal buffer length, the segment duration, and the bitrate of the quality level $l$, respectively; the deduction of $n_{1,\sigma}$ and $n_l$ can be found in [40]. Afterwards, the algorithm searches for the maximum quality level (equal to or lower than $l_{prev}$) that meets the condition $n_{prev,j} \geq 1$; if this feasible quality level is not achievable, then the lowest quality level is selected.

It is worth mentioning that the pseudocode presented in [40] would always lead to a decrease of at least one quality level if $l_{best} < l_{prev}$. However, this was not in agreement with the behavior of the fluctuations tests presented in [40] (and with additional tests performed during our QAAD implementation). Therefore, the lines (17) and (18) of Algorithm 1 were modified (the version presented herein already reflects this adjustment), so as to obtain the intended QAAD behavior. Moreover, for simulation purposes, the parameters $\mu$ and $\sigma$ were set to 10 s and 3 s, respectively, as suggested in [40].

4.3. Additional Video Application Considerations. It was assumed that the video to be downloaded by the users has 15 different representations/quality levels in the server, with the characteristics presented in Table 5 (these assumptions were based on a publicly available DASH dataset [41]).

The initial playout delay and rebuffering periods were set equal to the time that a client needs to download $5$ s of the lowest video quality. In the remaining situations, the video segment requests are made according to the QAAD algorithm (cf. Section 4.2). In addition, the buffer level is reported by the clients to the network in each segment request.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>System bandwidth</td>
<td>20 MHz</td>
</tr>
<tr>
<td>RBFs per TTI</td>
<td>100</td>
</tr>
<tr>
<td>OFDM symbols per RB</td>
<td>$12 \times 7$</td>
</tr>
<tr>
<td>MIMO configuration</td>
<td>$1 \times 1$</td>
</tr>
<tr>
<td>Overhead bits percentage</td>
<td>25%</td>
</tr>
<tr>
<td>CQI periodicity</td>
<td>5 ms</td>
</tr>
<tr>
<td>Channel model</td>
<td>Urban macrocell</td>
</tr>
</tbody>
</table>

**Algorithm 1: QAAD: bitrate selection.**

1. if $l_{best} = l_{prev}$ then
2. $l_{next} \leftarrow l_{prev}$
3. else
4. if $l_{best} > l_{prev}$ then
5. if $U > \mu$ then
6. $l_{next} \leftarrow l_{prev} + 1$
7. else
8. $l_{next} \leftarrow l_{prev}$
9. end if
10. else
11. if $l_{best} < l_{prev}$ then
12. $j \leftarrow 0$
13. do
14. $t_{prev,j} \leftarrow \frac{U - \sigma}{R_{est}[b(l_{prev} - j)]}$
15. $n_{prev,j} \leftarrow \frac{t_{1,\sigma} \times R_{est}}{t_{1,\sigma} \times b(l_{prev} - j)}$
16. $j \leftarrow j + 1$
17. while $n_{prev,j} < 1$ & $j < l_{prev}$
18. $l_{next} \leftarrow l_{prev} - (j - 1)$
19. end if
20. end if
21. end if
4.4. Other Implemented Scheduling Algorithms. In order to assess the relative performance of the proposed MBF scheduling algorithm, some of the scheduling techniques reviewed in Section 2.4 were also implemented in the developed simulator; the respective resource allocation metrics, which were adopted for simulations purposes, will now be detailed.

(i) Round-Robin (RR):
\[ m_{n,k}^{RR} [i] = K \times (n - n_0 [i]) + (k - k_0 [i]), \]  
where \( n_0 [i] \) and \( k_0 [i] \) correspond to the indices of the last scheduling unit allocated to the \( i^{th} \) user, i.e., this user was last served with scheduling unit \( k_0 [i] \) of scheduling period \( n_0 [i] \).

(ii) Blind Equal Throughput (BET):
\[ m_{n,k}^{BET} [i] = \frac{1}{\bar{R}_n [i]}, \]  
where \( \bar{R}_n [i] \) represents the past average throughput (prior to the \( n^{th} \) scheduling period) of the \( i^{th} \) user.

(iii) Proportional Fair (PF):
\[ m_{n,k}^{PF} [i] = \frac{\bar{R}_{n,k} [i]}{\bar{R}_n [i]}, \]  
(iv) Proportional Fair with Barrier for Frames (PFBF):
\[ m_{n,k}^{PFBE} [i] = \left( \frac{\lambda \times \bar{R}_{n,k} [i]}{F_n [i]} \right)^{\xi (f_{min} - f_s [i])} + \frac{\bar{R}_{n,k} [i]}{\bar{R}_n [i]} W [i], \]  
\[ W [i] = \begin{cases} 1 + \frac{N_U \times P_{rebufi}}{\sum_{j=1}^{N_U} P_{rebufj}}, & \text{if } \sum_{j=1}^{N_U} P_{rebufj} > 0 \\ 1, & \text{otherwise}, \end{cases} \]  
where \( F_n [i] \) denotes the size of the video frame that is being downloaded by the \( i^{th} \) user with respect to the \( n^{th} \) scheduling period, \( f_s [i] \) represents the number of video frames in the buffer of this user at the beginning of the \( n^{th} \) scheduling period, and \( f_{min} \), which is tunable, stands for the minimum number of frames that each client should have in the buffer; the variable \( P_{rebufi} \) corresponds to the percentage of the total streaming time spent by the \( i^{th} \) user in rebuffering, whereas \( \lambda \) and \( \xi \) represent other tunable parameters of this scheduling strategy. For simulation purposes, and as suggested in [26], \( f_{min} \) was set equal to 10 frames, while \( \lambda = \xi = 1 \).

4.5. QoE Model. In order to know how well the users are satisfied with the respective video streaming service, a QoE model based on the one adopted in [15] was incorporated in the developed simulator. This model, which yields outputs in the same range of MOS, takes into account the video quality and the rebuffering (or stall) events:
\[ \text{QoE} [i] = \max \left\{ 5.67 \bar{q} [i] - 6.65 \bar{q} [i] + 0.17 - 4.95 \right\}, \]  
where \( \bar{q} [i] \) and \( \bar{q} [i] \) stand for the average and standard deviation, respectively, of the video quality level served to the \( i^{th} \) user, \( q_{total} \) corresponds to the total number of different video quality levels available in the server, and \( \phi [i] \) represents a function that measures the impact of video freezes on the QoE of the \( i^{th} \) user, which is given by
\[ \phi [i] = \frac{7}{8} \times \max \left\{ \ln \left( \frac{\beta [i]}{6} + 1 \right) + \frac{1}{8} \right\} \times \frac{\phi [i]}{15}, \]  
where \( \beta [i] \) denotes the frequency of the rebuffering events experienced by the \( i^{th} \) user and \( \psi [i] \) represents the mean duration of these rebuffering events. Nevertheless, this model does not consider the initial playout delay, which may also impair the QoE. In the developed simulator, the initial playout delay is added to the duration of rebuffering events, thereby affecting the mean duration of these events.

As previously mentioned, this model yields outputs in the same range of MOS, which means that these outputs can be directly mapped onto the qualitative scale presented in Table 1, i.e., a user is assumed to have an excellent streaming video experience if the respective QoE is between 4 and 5, a bad experience if the QoE is between 0 and 1, etc.

5. Results

The results presented in this section were obtained through Monte Carlo simulations, with a confidence interval of 95% [42]. Each simulation corresponds to 3 minutes of resources scheduling. During this period, all the simulated users are connected to the base station and continuously receive and play the video delivered by the server. The number of users is kept constant during each simulation.

A user is considered to be satisfied if the respective QoE value is equal to, or greater than, a predefined threshold \( Y \), whereas the user is classified as unsatisfied if the respective
QoE is below a predefined threshold $Z$ (with $Z \leq Y$). For a given number of simulated users, the percentage of satisfied and unsatisfied users is denoted as $S$ and $NS$, respectively; accordingly

$$S(QoE \geq Y) = \frac{\text{Number of users with } QoE \geq Y}{\text{Number of connected users}} \times 100,$$

$$NS(QoE < Z) = \frac{\text{Number of users with } QoE < Z}{\text{Number of connected users}} \times 100.$$  

5.1. MBF’s $\alpha$ Parameter. As previously mentioned, the parameter $\alpha$ of the proposed MBF algorithm (cf. (8)) represents the emergency state threshold, allowing us to tune the behavior of this scheduling strategy between rebuffering events fairness and QoE efficiency. In order to evaluate the impact of $\alpha$ on the performance of the MBF approach, Figures 6 and 7 depict, respectively, $S(QoE \geq 3)$ and $NS(QoE < 2)$ as a function of the number of connected users and for different values of $\alpha$ (ranging from zero to infinity).

According to Figure 6, it can be observed that, for $S < 95\%$, the lower the value of $\alpha$, the higher the number of satisfied users that can be connected to the base station. On the other hand, and taking into account Figure 7, higher $\alpha$ values lead to a lower number of unsatisfied users. This behavior can be explained as follows: when the MBF algorithm enters in its emergency state, it will serve the users just based on their buffers, without considering their throughputs and requested segment video qualities; hence, by fulfilling the requirement to serve the user with the lowest buffer until its level is larger than $\alpha$, the remaining users will experience lower throughputs and will request lower video qualities according to the rate adaptation algorithm.

Therefore, the procedure of adjusting the $\alpha$ value can be regarded as setting a trade-off between having a higher number of connect users that are satisfied (lower $\alpha$ values) or having a lower number of users that are unsatisfied with the streaming session (higher $\alpha$ values). Accordingly, this enables us to meet different service provider preferences, such as the two modes suggested below:

(i) MBF–Mode 1: this approach aims, as first priority, to maximize the number of satisfied users ($QoE \geq Y$) and, secondly, at minimizing the number of unsatisfied users ($QoE < Z$). Thus, $\alpha = 5$ s can be regarded as an appropriate value for this mode, because it leads to a slightly less number of satisfied users than $\alpha = 0$ and $\alpha = 2.5$ s (cf. Figure 6), but it is better at maintaining a low number of unsatisfied users (cf. Figure 7).

(ii) MBF–Mode 2: this mode seeks, as first priority, to minimize the number of unsatisfied users and, secondly, to maximize the number of satisfied users. For this mode, an $\alpha$ of 11 s is proposed, as it leads to a low number of unsatisfied users (cf. Figure 7) but still maintains a high number of satisfied users (cf. Figure 6).

5.2. Satisfied Users. The performance of different scheduling strategies, namely, RR, BET, PF, and PFBF, is now compared to the performance of both MBF modes previously suggested, taking into account the percentage of satisfied users.

5.2.1. Satisfied Users: $QoE \geq 3$. Figure 8 shows the percentage of users with $QoE \geq 3$ as a function of the number of connected users and for the different scheduling solutions. As can be observed, the best performance for a low cell load is obtained by adopting the BET approach. However, as the number of connected users increases, the BET strategy leads to the worst performance; more specifically, when the amount
of available resources starts being scarce to serve all users as desired, the number of satisfied users starts drooping earlier when BET is enforced; henceforth, the scenario where the performance of the BET algorithm is lower than that of the remaining scheduling schemes will be denoted as the critical cell load situation.

On the other hand, the proposed MBF–Mode 1 strategy yields the best performance in this critical cell load situation. In particular, assuming that a service provider aims at having at least 90% of satisfied users (i.e., \( S \geq 90\% \)), the number of users that can be connected in this situation is maximized by using the MBF–Mode 1 scheme (cf. Figure 9), which depicts the maximum number of users that can be connected while assuring that 90% of them have \( QoE \geq 3 \).

With respect to the MBF–Mode 2 approach operating in the critical cell load scenario, its performance is surpassed in some situations by the PF and RR strategies (besides the MBF–Mode 1 scheme), namely, when the number of connected users is very high (cf. Figure 8). This is an expected behavior, as the proposed MBF scheduling scheme operating in Mode 2 is not optimized for achieving the highest possible number of satisfied users. Nevertheless, it still attains the second-best performance (after MBF–Mode 1) when considering the \( S \geq 90\% \) benchmark (cf. Figure 9).

5.2.2. Satisfied Users: \( QoE \geq 4 \). Figure 10 presents the percentage of users with \( QoE \geq 4 \)—which can be considered highly satisfied users—as a function of the number of connected users and for the different scheduling solutions. As can be seen, the results are similar to the ones obtained for \( QoE \geq 3 \): the BET algorithm has the best performance when the number of connected users is low but yields the worst results when the number of connected users exceeds 40; the proposed MBF strategy, on the other hand, is able to provide one of the best performances in the critical cell load scenario. In particular, if a network operator wants to provide a very high QoE to the respective users, then the proposed MBF scheme operating in Mode 1 is a good choice, as it maximizes the number of users that can be connected while assuring that at least 90% of them have \( QoE \geq 4 \) (cf. Figure 11).

Nevertheless, there is a slight qualitative difference of the results shown in Figure 10 with respect to the previous analysis for \( QoE \geq 3 \) (cf. Figure 8), namely, when the number of connected users is very high regarding the critical cell load situation: the performance of both MBF modes is now surpassed by the PF and RR scheduling techniques. This behavior is explained by the fact that an MBF-based scheduler tries to exploit the higher throughputs experienced by some users, in order to provide them higher QoE, but without sacrificing the remaining users. For the PF and RR strategies, even though in some situations more users can be served with \( QoE \geq 4 \) than the ones served by the MBF scheme, this comes at the cost of increasing the number of unsatisfied
users (i.e., with very low QoE values), as will be seen next; hence, the MBF algorithm adopts a fairer approach towards serving more users with an acceptable QoE.

5.3. Unsatisfied Users. A service provider must try also to minimize the number of users experiencing very low QoE values, as they may decide to drop out, which would eventually decrease the profits obtained by the service provider. In this context, Figure 12 depicts the percentage of users with QoE < 2 as a function of the number of connected users and for the different scheduling solutions. As expected, the MBF–Mode 2 yields better results than the MBF–Mode 1, because the former was specifically adjusted to minimize the number of users experiencing a very low QoE. Furthermore, the performance of both MBF modes is only surpassed by the performance of the BET technique. Nevertheless, it is important to recall that the MBF strategy yielded better results than

the BET scheme when considering the number of satisfied users in the critical cell load situation (cf. Section 5.2).

The results presented in Figure 12, in conjunction with the ones of Figures 8 and 10, also allow us to show the balance achieved by the PFBBF strategy: although it leads to a number of satisfied users that is lower than in the RR and PF approaches, a lower number of unsatisfied users is also present. This enables us to point out the major advantage of the proposed MBF algorithm regarding this trade-off between fairness and QoE efficiency: the MBF strategy always yields a better performance than the PFBBF approach, not only by lowering the number of unsatisfied users, but also by increasing the number of satisfied users; regarding this last assessment metric, the performance of the proposed MBF technique even exceeds, in most of the simulated cases, the one attained by the RR and PF schemes (cf. Section 5.2).

6. Conclusions

This paper focused on the role played by scheduling algorithms in the resulting users’ QoE for video streaming services. In particular, a new scheduling technique was proposed, the MBF scheme, which allocates the resources based on the clients’ buffer levels and predicted filling. This scheduling strategy not only has the goal of preventing playout stalls but also exploits the higher throughputs experienced by some users, thus aiming to improve the overall QoE efficiency without sacrificing some users. Furthermore, the proposed technique incorporates a tuning parameter, which enables a service provider to easily adjust the trade-off between minimizing the number of users experiencing a very low QoE (fairness) and maximizing the overall QoE of the network (efficiency).

The proposed algorithm was assessed by simulating a scenario comprising an LTE base station and several mobile users, which are accessing to a video server according to the 3GP-DASH specifications. Two operating modes were considered for the proposed MBF strategy: while the MBF–Mode 1 seeks to maximize the number of users with good or excellent QoE, the MBF–Mode 2 puts more emphasis on minimizing the number of users with poor or bad QoE. The performances of both MBF modes were compared to the performances yielded by other scheduling techniques, namely, RR, BET, PF, and PFBBF, with respect to the number of satisfied and unsatisfied users, i.e., users with high and low QoE levels, respectively.

The results showed that both MBF modes attained a superior performance than the other scheduling algorithms in terms of the number of users with a good or excellent QoE. In particular, the MBF-based scheduler is capable of providing almost a higher number of admissible users such that 90% of them are satisfied (i.e., with high QoE levels). Additionally, and regardless of the considered operating modes, the proposed MBF strategy also leads to a percentage of users with poor or bad QoE that is lower than in the RR, PF, and PFBBF algorithms. Accordingly, it was demonstrated that the proposed MBF scheduling algorithm is able to accomplish its mission of serving more users with high QoE, without compromising the QoE of the remaining users.
Notations

\( K \): Number of scheduling units per scheduling period
\( T_{SG} \): Scheduling period length
\( m_{n,k}[i] \): Resource allocation metric of the \( i^{th} \) user with respect to the \( k^{th} \) scheduling unit of the \( n^{th} \) scheduling period
\( U_n[i] \): Amount of video playout time stored in the \( i^{th} \) user's buffer at the beginning of the \( n^{th} \) scheduling period
\( \Delta U_{n,k}[i] \): Predicted buffer level variation of the \( i^{th} \) user if the \((n,k)^{th}\) scheduling unit is allocated to this user
\( \overline{R}_{n,k}[i] \): Estimated achievable throughput for the \((n,k)^{th}\) scheduling unit given the channel conditions of the \( i^{th} \) user
\( V_n[i] \): Bitrate of the video that is being downloaded by the \( i^{th} \) user during the \( n^{th} \) scheduling period
\( E_{nk}(CQI[i]) \): Efficiency of the \( k^{th} \) RBP with respect to the \( n^{th} \) scheduling period when considering \( CQI[i] \)
\( CQI[i] \): Last CQI reported by the \( i^{th} \) user
\( \Omega \): Number of OFDM symbols that each RB encompasses
\( G \): Gain obtained by using the MIMO technique
\( \xi \): Percentage of overhead bits
\( \Delta U_{n,j}^*[i] \): Estimate of how much the buffer filling of the \( j^{th} \) user will increase with respect to the \( f^{th} \) allocated scheduling unit in the \( n^{th} \) scheduling period
\( \alpha \): Emergency state threshold of the MBF scheduling algorithm
\( T_{max} \): Total simulation time
\( N_U \): Number of connected users
\( R_{est} \): Estimated channel throughput by the rate adaptation algorithm
\( R_{sample} \): Sampled channel throughput by the rate adaptation algorithm
\( \theta \): Period concerning \( R_{est} \) estimation and \( R_{sample} \) sampling
\( B \): Amount of bits downloaded during \( \theta \)
\( b(l) \): Bitrate of the quality level \( l \)
\( l_{next} \): Next requested video segment
\( l_{prev} \): Previously selected segment
\( l_{best} \): Video segment which has the highest bitrate that does not exceed \( R_{est} \)
\( \mu \): Marginal buffer length (rate adaptation algorithm)
\( t_{l,o} \): Elapsed time to consume all the segments in the playback buffer
\( n_l \): Expected number of segments to be downloaded during \( t_{l,o} \)
\( \sigma \): Predefined minimal buffer length (rate adaptation algorithm)
\( \tau \): Segment duration

\( \omega \): Weight factor for \( R_{sample} \) (rate adaptation algorithm)
\( n_{o}[i] \): Scheduling period index of the last scheduling unit allocated to the \( i^{th} \) user
\( k_{o}[i] \): Index (within \( n_{o}[i] \)) of the last scheduling unit allocated to the \( i^{th} \) user
\( \overline{R}_n[i] \): Past average throughput (prior to the \( n^{th} \) scheduling period) of the \( i^{th} \) user
\( F_n[i] \): Size of the video frame that is being downloaded by the \( i^{th} \) user with respect to the \( n^{th} \) scheduling period
\( f_n[i] \): Number of video frames in the buffer of the \( i^{th} \) user at the beginning of the \( n^{th} \) scheduling period
\( f_{min} \): Minimum number of frames that each client should have in the buffer (PPBF scheduling algorithm)
\( \lambda \): Percentage of the total streaming time spent by the \( i^{th} \) user in rebuffering

\( \xi \): Tunable parameter of the PPBF scheduling algorithm
\( QoE[i] \): QoE value (within the MOS scale) with respect to the \( i^{th} \) user (output of the QoE model)
\( \overline{q}[i] \): Average video quality level served to the \( i^{th} \) user
\( \varepsilon[i] \): Standard deviation of the video quality level served to the \( i^{th} \) user
\( q_{total} \): Total number of different video quality levels available in the server
\( \phi[i] \): Function that measures the impact of video freezes on the QoE of the \( i^{th} \) user (QoE model)
\( \beta[i] \): Frequency of the rebuffering events experienced by the \( i^{th} \) user
\( \psi[i] \): Mean duration of the rebuffering events experienced by the \( i^{th} \) user
\( S(QoE \geq Y) \): Percentage of satisfied users (with QoE values greater or equal than \( Y \))
\( NS(QoE < Z) \): Percentage of unsatisfied users (with QoE values less than \( Z \)).

Data Availability

The simulator data used to support the findings of this study have been deposited in the GitHub repository (https://github.com/ivo-sousa/scheduling_LTE).

Conflicts of Interest

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

This work was funded by Instituto de Telecomunicações and Fundação para a Ciência e a Tecnologia (FCT) under Project UID/EIA/50008/2013 and by Project MESMOQoE (no. 023110–16/SI/2016) supported by Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (ERDF).

References


