Chapter 5

Simulator implementation

5.1 Introduction

In order to test the new scheduling algorithm proposed in this thesis, a Java program was developed using an integrated development environment, Eclipse. This program simulates a LTE network scenario where several mobile users which implement the MPEG-DASH specification are streaming video. The program was designed in a modular form, so as to facilitate the process of implementing additional scheduling algorithms, QoE models and rate adaptation algorithms for requesting video segments. The program has multiple classes of objects: *User*, *Video*, *Throughput*, *CQI*, *Scenario, Scheduler, QoEModel, RateAdaptationAlgorithm* and the *Main*. The Users are stored in an ArrayList that belongs to the *Scenario*. Each *User* is streaming a certain *Video*, have a certain *Throughput* according to the number of resource blocks allocated and current *CQI* and implement a *RateAdaptationAlgorithm*. The *Main* class is responsible for running the program simulating a possible scenario and determine the users' QoE using a certain QoEModel. It has a "for loop" where the number of iterations is equal to the number of milliseconds that one wants to simulate. In each iteration the allocation algorithm calculates the metric for all users and allocates the resource blocks to one single user, the throughput and QoE until the moment of each user are calculated, the buffer is updated and it is verified if any user requested for a new segment. If the user with the highest metric does not need all the available resource blocks to finish receiving the entire requested video segment, then the user with the second highest metric receives the remaining RB as long as he needs them and so on.

In section 5.2 it is explained how users report their CQIs, how these values influence their throughputs, and how user's buffer level is updated. In section 5.3, it is presented and explained one QoE adaptation algorithm over DASH that simulates the requests of the video segments performed by DASH clients. In section 5.4 it is described the QoE model that is used in the simulations to predict the quality of experience of the users. Finally, in section 5.5 are pointed the algorithms that were implemented in the simulator to be able to compare the performance of MBF.

5.2 User CQI, throughput and buffer

It is considered that there are no losses during the transmission of data to the users. However, the channel quality must vary according to its conditions. The variability of the channel conditions affects the number of bits that can be transmitted per resource block. This variability makes it essential to have a good scheduler capable of providing a good level of QoE to all users while maintaining a high bandwidth efficiency. The base station has access to this information through the periodic CQI reports sent by users. As it was stated in Section 4.1, the minimum periodicity could be 2 ms for FDD and 1 ms for TDD. In the simulator, all the users report their CQIs to the base station every 5 milliseconds. A smaller or higher value could be also used, as it is used in the literature. For example, in [15] it is used a periodicity of 5 milliseconds too. The CQIs of the users were obtained through a simulation that was

made using OMNeT++ [45, 46]. OMNeT++ is open source, not-proprietary and supports a variety of simulated traffic engineering protocols. INET-Framework [45] must be added to OMNeT++ as the extension. INET-Framework is an open-source extension that allows OMNeT++ to simulate network communication in TCP/IP architecture. INET-Framework consists of protocol e.g. IPv4, IPv6, TCP, SCTP, UDP, PPP, Ethernet, and 802.11. For traffic engineering simulation, INET-Framework supports various protocols e.g. IntServ, DifServ, MPLS, RSVP-TE and LDP Signalling. In addition, SimuLTE [47, 48] tool was used which is an open source project that is built on top of OMNeT++ and INET Framework. It is a simulation tool enabling complex system level performance-evaluation of LTE and LTE Advanced networks (3GPP Release 8 and beyond) for the OMNeT++ framework. A scenario with a single base station and 200 mobile users moving around with a random trajectory was simulated, for three minutes. The CQIs reported by the users to the base station were saved. There are users that report always CQI=15 during all the entire simulation but there are also users whose CQI varies between 2 and 15, so there is heterogeneity. The users created in the simulator simulate randomly the channel of one of the 200 users simulated in OMNET++, by importing the file saved from OMNET++.

A higher CQI will result in a higher resource block efficiency, according to Table 4.2 and thus in a higher throughput. In each TTI the scheduler assigns an amount of resource blocks to the user who has the highest metric, and according to the number of resource blocks allocated and user's CQI, the user has a certain throughput that is calculated using the equation (5.1):

$$U^{i}_{throughput}[bits/s] = NbrRB_i \times 1000 \times 12 \times 7 \times 2 \times eff[CQI_i] \times NbrA \times 0.75 \quad (5.1)$$

where $NbrRB_i$ is the number of resource blocks allocated to i^{th} user, $NbrA$ is the number of antennas and $eff[CQI_i]$ is the resource block efficiency which depends on the i^{th} user reported CQI, CQI_i. These efficiencies are presented in Table 4.2. Each resource block consists of 2 time slots of 0.5ms and each time slot corresponds to 7 symbols in the time domain. In the frequency domain one resource block consists of 12 subcarriers. It is considered 25% of overhead used for controlling and signaling like Reference Signal, Primary Synchronization Signal (PSS), Secondary Synchronization Channel (SSS) and Physical Broadcast Channel (PBCH), etc., according to [49].

User's buffer is measured in milliseconds. It could be also measured in bits but it would also be necessary to store the video quality of the segment to which these bits belong. Furthermore, it would be more difficult to subtract the number of bits that are leaked from buffer each millisecond. In fact, one has a better idea of the buffer level if it is measured in milliseconds rather than in number of received bits. So, users are receiving bits and these bits correspond to a certain number of milliseconds in the buffer. Every 1 millisecond the buffer is filled with a certain number of milliseconds of video, according to the representation requested by the user and its throughput, and leaks out 1 millisecond of video, except when the user is requesting the initial segments (initial buffering) or recovering from a stall (re-buffering event). In these cases, it is not subtracted 1 ms every millisecond spent because the video is not playing. Except for these cases, the buffer variation is then calculated every millisecond with equation (5.2):

$$\Delta U^{i}_{buffer}[ms] = \frac{U^{i}_{throughput}}{U^{i}_{RequestedVideo}} - 1 \quad (5.2)$$

where $U^i_{RequestedVideo}$ is the bitrate of the segment that the user requested and that is being downloaded. Thus, the buffer level at the beginning of millisecond t is given by the equation (5.3):

$$U^i_{buffer}(t) = U^i_{buffer}(t-1) + \Delta U^i_{buffer} \tag{5.3}$$

where $U^i_{buffer}(t-1)$ is the buffer level at the beginning of millisecond $t-1$.

5.3 QoE adaptation algorithm

The video exists in the server with the following bitrates: {0.2, 0.25, 0.3, 0.4, 0.5, 0.7, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0, 6.0, and 8.0} Mbps and the user has the possibility of choosing the segment length to download: {1, 2, 4, 6, 8 or 15} seconds. This information was obtained based on a DASH dataset [50]. It is considered a constant bitrate for all video qualities. During the simulation, users download always segments with the same length.

At the beginning of the simulation, users start by requesting a certain number of segments with the lowest quality. The time spent between this request and the time they have 5 seconds of video in their buffers is the initial delay. The buffer is not leaking out during this period. After that, each millisecond of the video buffered seen by the user is subtracted from the buffer amount. DASH client behavior is not standardized. Anyway, a rate adaptation algorithm needs to be implemented in order to simulate client segment requests. This behavior is how client adapts to the available bandwidth. Naturally if the bandwidth is very low, client should request video with a lower quality. The same way, if bandwidth is large, client should ask for a better quality.

The implemented QoE adaptation algorithm is QoE-enhanced Adaptation Algorithm over DASH (QAAD), presented in [51]. In [51], the authors propose a QoE adaptation algorithm that has shown to improve user's QoE by gradually decreasing the quality level when the available bandwidth network decreases drastically. Furthermore, QAAD stabilizes the quality level when the available network bandwidth fluctuates without playback interruption, as it will be shown. The adaptation algorithm in the DASH client consists of two parts:

1. **Bandwidth estimation part**: the available bandwidth at time t, denoted by $bw_{estimated}(t)$, is estimated using equation (5.4), every θ seconds. It uses the previously calculated bandwidth estimated, $bw_{estimated}(t-\theta)$. Since the channel quality is time-varying, the estimated bandwidth is smoothed by means of a weight moving average:

$$bw_{estimated}(t) = w \times bw_{estimated}(t-\theta) + (1-w) \times bw_{sample} \tag{5.4}$$

where the parameter w is the weight factor for sampled bandwidth, bw_{sample} $(0 < w < 1)$. The sampled bandwidth, bw_{sample}, is the available network bandwidth, sampled in every θ seconds and computed using equation (5.5):

$$bw_{sample} = \frac{K}{\theta} \tag{5.5}$$

where K is the amount of data downloaded during θ seconds.

2. **Bitrate selection part**: once the estimated bandwidth has been calculated at time t, the video quality of the next segment is selected, denoted by l_{next}, based on the available estimated bandwidth and the buffer length, $B(t)$. Furthermore, l_{best} represents the highest video quality that does not exceed the estimated bandwidth.

   The QAAD pseudocode is the following [51]:

1: if $l_{best} == l_{prev}$ **then**
2: $l_{next} = l_{prev}$
3: else
4: **if** $l_{best} > l_{prev}$ **then**
5: **if** $B(t) > \mu$ **then**
6: $l_{next} = l_{prev} + 1$
7: **else**
8: $l_{next} = l_{prev}$
9: **end if**
10: else
11: **if** $l_{best} < l_{prev}$ **then**
12: $k = 0$
13: **do**
14: $t_{l_{prev}-k,\sigma} = \frac{B(t)-\sigma}{1-bw_{estimated} / b(l_{prev}-k)}$
15: $n_{l_{prev}-k} = \frac{t_{l,\sigma} \times bw_{estimated}}{\tau \times b(l_{prev}-k)}$
16: $k = k + 1$
17: **while** $\mathrm{n_{l_{prev}-k}} < 1$ && $\mathrm{k} < \mathrm{l_{prev}} - 1$
18: **end while**
19: $l_{next} = l_{prev} - k$
20: end if

Initially the algorithm tests if the current best quality reachable is equal with the previous one ($l_{best} == l_{prev}$, line 1). If it is, no change needs to be made ($l_{next} = l_{prev}$, line 2). Then, it is tested the possibility of increasing the bitrate for the next segments. If $l_{best} > l_{prev}$ (line 4), it is possible to increase the bitrate. However, the current buffer length is considered due to possible unpredictable throughput fluctuation. Thus, the video quality requested is only incremented ($l_{next} = l_{prev} + 1$, line 6) if the current buffer available is larger than a certain marginal buffer length μ ($\mathrm{B} > \mu$, line 5). Otherwise, the quality of the requested segments does not change ($l_{next} = l_{prev}$, line 8). Finally, if $l_{best} < l_{prev}$, the previous video quality cannot be kept. To avoid lowering more than two levels and this way avoiding a QoE reduction, the algorithm tries to keep the next video quality comparable to the previous one. To this end, the elapsed time to deplete the segments in the playback buffer, $t_{l,\sigma}$ and the expected number of segments downloaded during the time, n_l, should be first determined. The deduction of $t_{l,\sigma}$ and n_l can be found in [51]. In lines 13-18, the algorithm searches for the maximum feasible quality level lower than l_{prev} by iterating the index k. The loop terminates when it founds a quality level $l_{prev} - k$ that is feasible, i.e., $n_{l_{prev}-k} \geq 1$.

The authors of [51] made two different tests to prove the behavior of QAAD: a step-down test and a fluctuation test. In this thesis, the tests have been also done and the results are presented. Analyzing the algorithm QAAD it was found that if $\mathrm{l}_{\mathrm{best}} < \mathrm{l}_{\mathrm{prev}}$ (line 11), once inside the loop $(13-17)$ the resulting k is greater than or equal to 1 because k is always incremented at least once (line 16). This means that the video quality will always decrease at least 1 level (line 19, $\mathrm{l}_{\mathrm{next}} = \mathrm{l}_{\mathrm{prev}} - k$). So, it is impossible to maintain the video level quality if $\mathrm{l}_{\mathrm{best}} < \mathrm{l}_{\mathrm{prev}}$ or $\mathrm{bw}_{estimated} < \mathrm{l}_{\mathrm{prev}}$. Then, a modification was made in order to make possible the behavior of the fluctuation test presented in Figure 5.7 and Figure 5.8. Namely, the line 19 was modified to $\mathrm{l}_{\mathrm{next}} = \mathrm{l}_{\mathrm{prev}} - (k-1)$.

- **Step-down test**

The first test consists of a step-down test where the available bandwidth is reduced from 2200 kbps to 800 kbps at $t = 30s$ The segment size used is 2 seconds, and the following parameters are used: $\theta = 0.3s, w = 0.875, \mu = 10s\,, \sigma = 3s$. The available video bitrates are {400, 500, 600, 800, 1000, 1200, 1500 and 2000} kbps, as used by the authors in [51]. In Figure 5.1, it is presented the throughput experienced by the user.

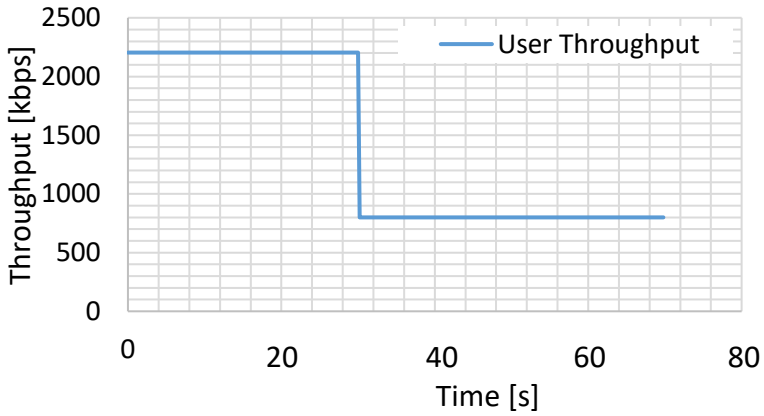

Figure 5.1 - User throughput for the step-down test.

In Figure 5.2, it is presented the estimated bandwidth by the client which is made every 0.3 seconds and the requested segment bitrates. The estimated bandwidth does not decrease abruptly because it is smoothed by means of a weight moving average, according to equation (5.4).

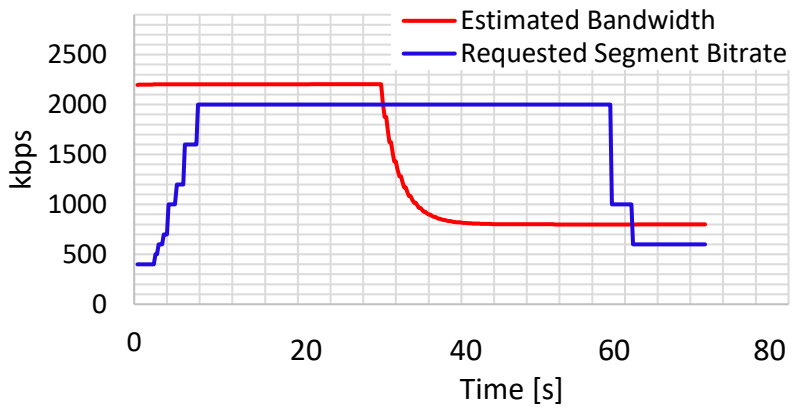

Figure 5.2 - Estimated bandwidth and requested segment bitrate for the step-down test, achieved using the developed simulator.

Initially user requests the lower segments at least until the buffer length is equal to 10 seconds. Figure 5.3 presents the user buffer length along time, in milliseconds, obtained in this thesis. Initially buffer length increases quickly because the requested segment bitrate is much lower than the estimated bandwidth. Approximately at 60 seconds, the buffer level increases again because the video bitrate is lower than the throughput. Then the quality level increases smoothly according to QAAD, because the estimated bandwidth is much greater than the segment bitrate.

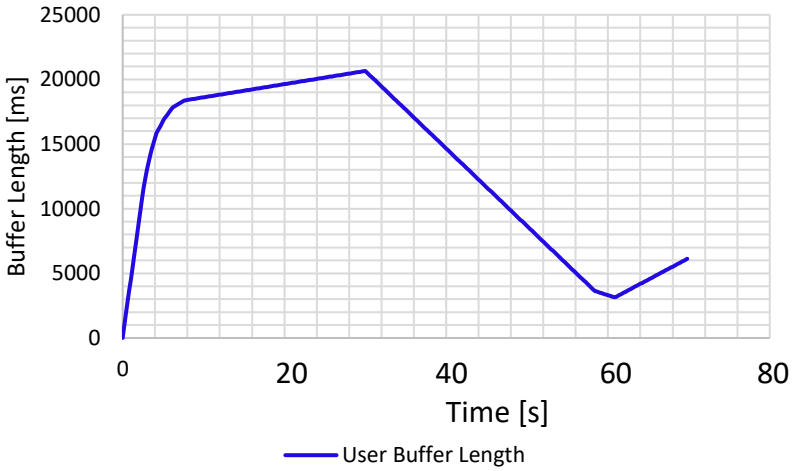

Figure 5.3 - User buffer length for the step-down test, achieved using the developed simulator.

In Figure 5.2, the representation of the requested segment bitrates after the 30 seconds when the throughput is abruptly decreased is not exactly the same presented in [51]. According to the results presented in [51], the requested bandwidth decreases at approximately at 49 and 59 seconds, as it can be seen in Figure 5.4. However, $n_{l_{prev-k}}$ is not greater than 1 if the buffer length at 49 seconds (approximately 10 seconds) is used, presented in [51], as it can be seen in Figure 5.5 which presents the user buffer length along time, in seconds, obtained in [51]. So, it is not understood how the client asks for a lower quality level at 49 seconds, according to the QAAD algorithm presented in [51].

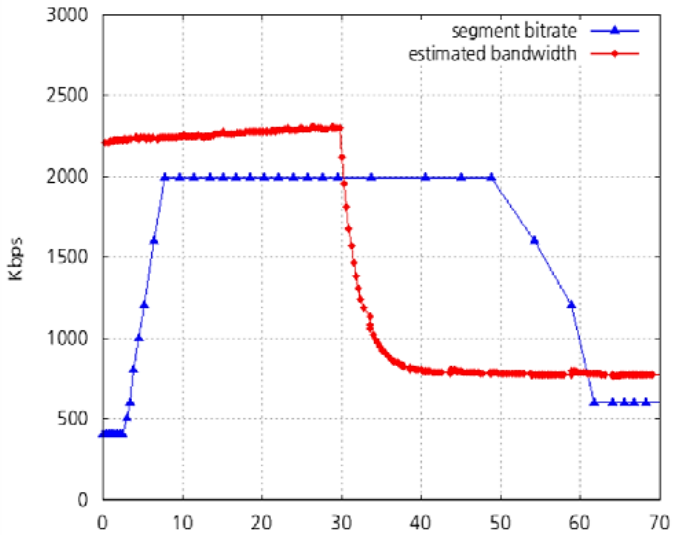

Figure 5.4 - Estimated bandwidth and requested segment bitrate for the step-down test, presented in [51].

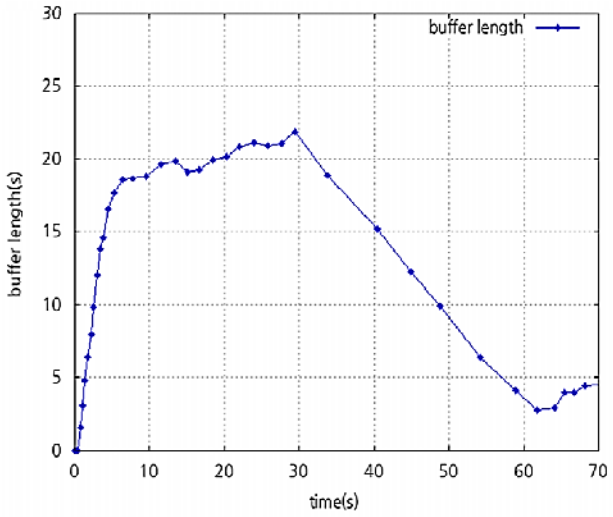

Figure 5.5 - User buffer length for the step-down test, presented in [51].

- **Fluctuation Test**

In the second test, the available bandwidth is fluctuated repeatedly every 4 seconds from 2100 kbps to 800 kbps and again from 800 kbps to 2100 kbps, according to Figure 5.6. The segments have 2 seconds and the following parameters are used: $\theta = 0.3s, w = 0.875, \mu = 10s\,, \sigma = 3s$. The available video bitrates are {400, 500, 600, 800, 1000, 1200, 1500 and 2000} kbps, as used by the authors in [51].

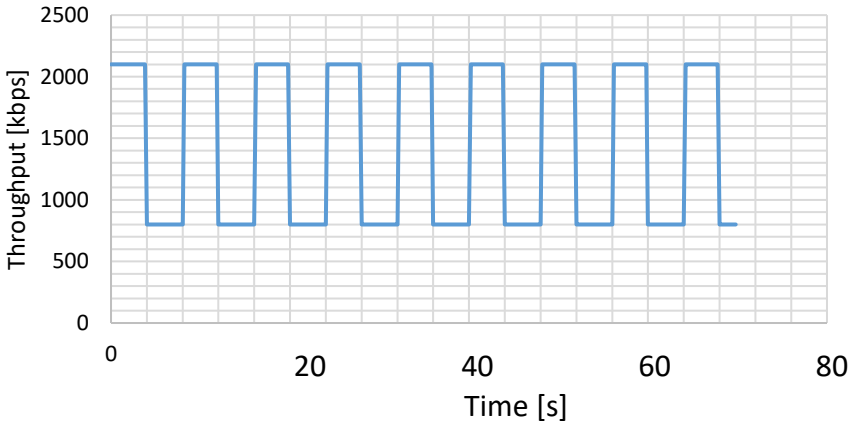

Figure 5.6 - User throughput for the fluctuation test.

In Figure 5.7, it is presented the bandwidth estimation performed by the client every 0.3 seconds and the requested segment bitrates. QAAD increases the bitrate in a conservative manner even though the network bandwidth increases and thus, more segments can be accumulated. The requested segment bitrate can be stabilized to 1600 kbps along the time when the estimated bandwidth is fluctuating. The results presented in Figure 5.7, achieved using the developed simulator, are very similar to the ones presented in Figure 5.8 which have been obtained in [51].

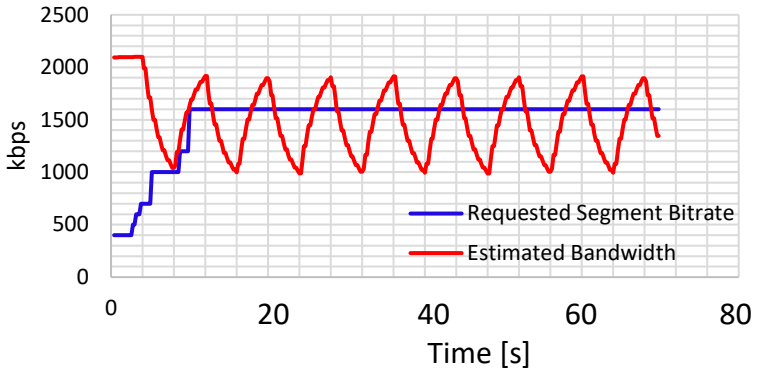

Figure 5.7 - Estimated bandwidth and requested segment bitrate for the step-down test, achieved using the developed simulator.

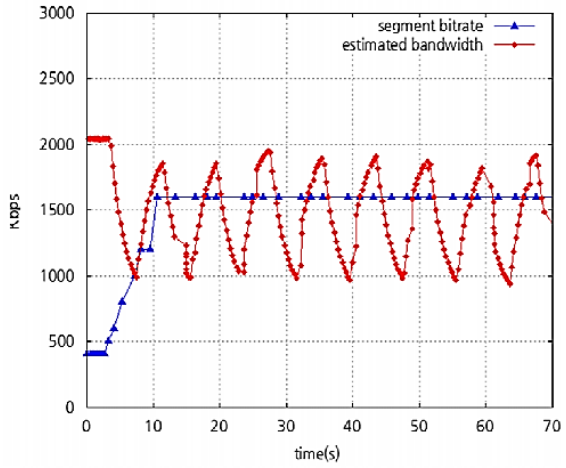

Figure 5.8 - Estimated bandwidth and requested segment bitrate for the step-down test, presented in [51].

Figure 5.9 presents the user buffer length along time, in milliseconds. A lot of segments are accumulated during the first 10 seconds because QAAD increases the bitrate conservatively, thus avoiding some re-buffering events. Then the buffer level is also fluctuating because the estimated bandwidth sometimes is bigger than the requested segment bitrate, sometimes is smaller. In Figure 5.10, it is presented the user buffer length that is achieved by the authors of [51].

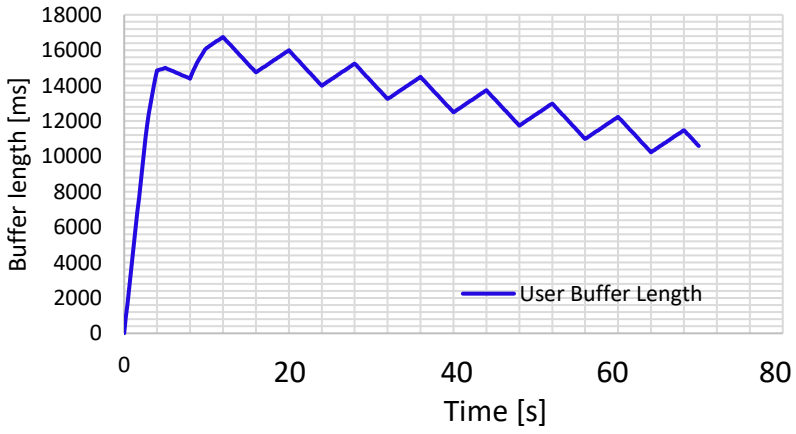

Figure 5.9 - User buffer length for the fluctuation test, achieved using the developed simulator.

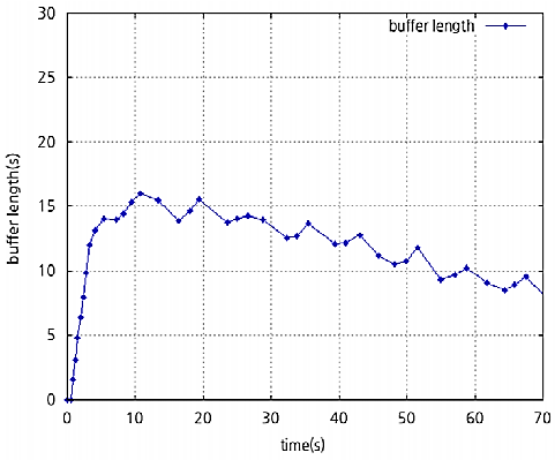

Figure 5.10 - User buffer length for the fluctuation test, presented in [51].

5.4 QoE model

In order to know if the connected users streaming video are satisfied with the service or which ones are satisfied and which ones are not, it is necessary to use a model capable of estimating the QoE of each user who is streaming video. In a video streaming session there are some parameters that affect clearly the QoE. Intuitively, the most important ones are the quality served and the re-buffering events. To be more accurate, the average served quality, the standard deviation of the served quality, the frequency of changes of video served quality, the frequency of re-buffering events, the mean duration of re-buffering events and the initial delay affect QoE and are the most used parameters to calculate the QoE in the literature. In [52], a QoE model is proposed and it is calculated using equations (5.6) and (5.7):

$$QoE_i = 5.67 \times \frac{\bar{q}_i}{q_{max}} - 6.72 \times \frac{\hat{q}_i}{q_{max}} + 0.17 - 4.95 \times F_i \tag{5.6}$$

$$F_i = \frac{7}{8} \times \max\left(\frac{\ln(\phi_i)}{6} + 1\,, 0\right) + \frac{1}{8} \times \frac{\min(\varphi_i\,, 15)}{15} \tag{5.7}$$

where $\bar{q}_i$ is the average video served quality to the i^{th} user, q_{max} is the highest video quality level available in the server (or the number of available quality levels), $\hat{q}_i$ is the standard deviation of the video served qualities to the i^{th} user, ϕ_i and φ_i are the frequency and mean duration of re-buffering events respectively. $\bar{q}_i$ and $\hat{q}_i$ can be calculated with equations (5.8) and (5.9), respectively:

$$\bar{q}_i = \frac{\sum_{k=1}^{K} QL_k}{K} \tag{5.8}$$

$$\hat{q}_i = \sqrt{\frac{\sum_{k=1}^{K}(QL_k - \bar{q}_i)}{K}} \tag{5.9}$$

where K is the total number of downloaded segments and QL_k is the quality level of k^{th} downloaded video segment. The re-buffering frequency ϕ_i is calculated dividing the number of stalls by the duration of streaming session (duration of video downloaded plus initial buffering delay and re-buffering). All the coefficients presented in equations (5.6) and (5.7), proposed in [52], have been fixed according to the

work [53], whose authors have made a small subjective test in order to tune the parameters of the proposed model. However, this QoE model presented does not consider the initial buffering delay. To have into account this parameter, it is considered that the initial delay is not a re-buffering event and thus does not affect the re-buffering frequency. Nevertheless, the period of initial buffering is added to the duration of re-buffering events, affecting this way the mean duration of re-buffering events. The reason is that initial delay is not an event that interrupts the video play. It does not annoy as much as a re-buffering event with the same duration does, because when a re-buffering event happens, the user was already engaged with the video.

With this model it is possible to analyze the QoE of users each TTI along the time of the streaming session (achieved QoE until a certain moment) and global QoE - the overall QoE of the entire streaming session. It is possible to know what was the mean global QoE experienced by the users and the standard deviation of their QoEs which can be interesting results to make a comparison between different scheduling algorithms. Table 5.1 classifies the quality of experience the i^{th} user had streaming the video.

Table 5.1 - Perceived quality by user that experienced a global QoE equal to QoE_i.

QoE_i	Perceived quality
$\mathbf{4 \leq QoE_i \leq 5}$	Excellent
$\mathbf{3 \leq QoE_i < 4}$	Good
$\mathbf{2 \leq QoE_i < 3}$	Fair
$\mathbf{1 \leq QoE_i < 2}$	Poor
$\mathbf{0 \leq QoE_i < 1}$	Bad

Using this model, it is considered that the i^{th} user had an excellent experience streaming video if his QoE is between 4 and 5. He had a bad experience if his QoE is between 0 and 1, and so on.

5.5 Implemented scheduling algorithms

In order to be able to compare the proposed allocation algorithm in this thesis, other existing algorithms were implemented, namely Round Robin, Blind Equal Throughput, Proportional Fair and PFBF. These algorithms have been described in section 4.3.1 and 4.3.2. These schedulers were implemented for the reasons explained in section 4.5.

- **Round Robin**

The simplest implemented algorithm is the Round Robin which consists of allocating the same number of resource blocks to all users. In order to prove that the Round Robin algorithm is well implemented a test is presented. Three users are streaming video for 3 minutes. Their reported CQIs are presented in Figure 5.11.

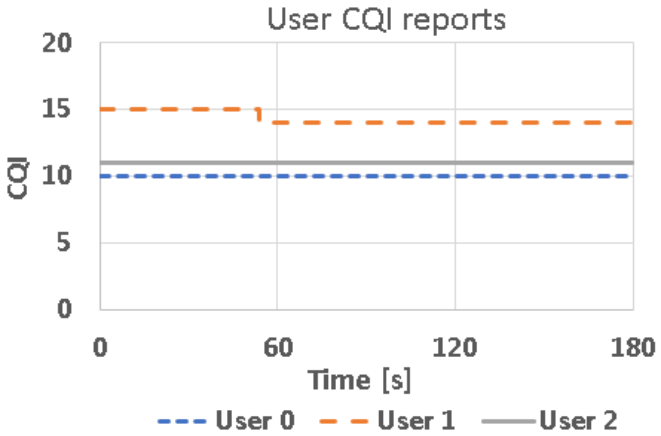

Figure 5.11 - CQIs reported by users along time for the RR test.

Although they have different CQIs, meaning that they receive a different number of bits per resource block allocated to them, RR ensure that all the users receive the same number of RB along time, as it can be confirmed in Figure 5.12.

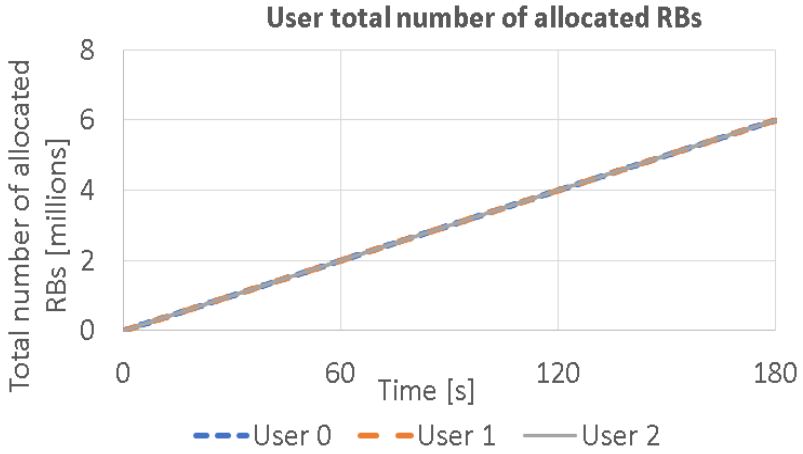

Figure 5.12 - Total number of allocated RB along time for the RR test.

- **Blind Equal Throughput**

Blind Equal Throughput is also implemented. This algorithm allocates in each TTI resource blocks to the user who has the lowest average throughput so far. It is fair in terms of users' throughputs. In order to prove that the BET algorithm is well implemented a test is presented. Three users are streaming video for 3 minutes. Their reported CQIs are presented in Figure 5.13.

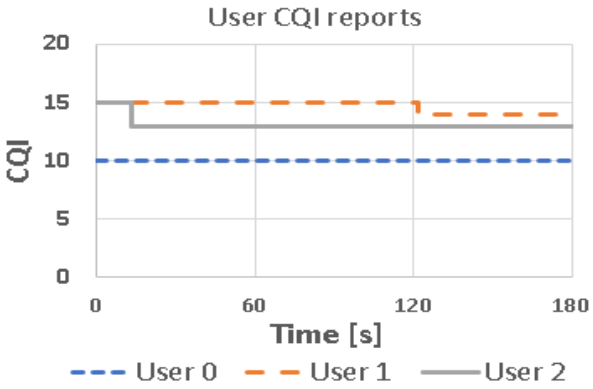

Figure 5.13 - CQIs reported by users along time for the BET test.

Although they have different CQIs, meaning that they receive a different number of bits per resource block allocated to them, BET ensures that all the users receive the same average throughput along time, as it can be verified in Figure 5.14.

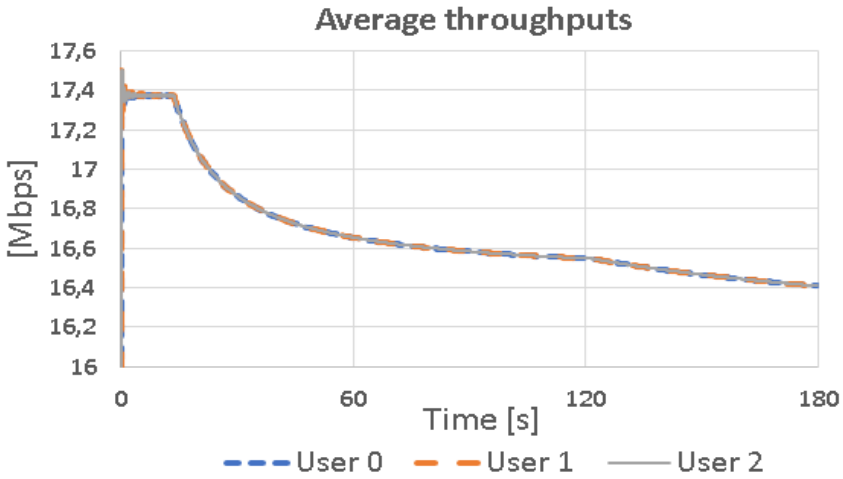

Figure 5.14 - Users' average throughputs for the BET test.

- **Proportional Fair**

The Proportional Fair algorithm was also implemented. This algorithm takes into account not only the average throughput but also the current channel conditions providing higher network efficiency. For example, if two users have the same average throughput, the resources are going to be allocated to the one with higher CQI.

To prove that the Proportional Fair algorithm is well implemented a test is presented. In this simulation two users are streaming video during 100 milliseconds. User 0 has a constant CQI equal to 10 and User 1 has a constant CQI equal to 15, according to Figure 5.15. Therefore, the achievable throughput of user 1 is always greater than the achievable throughput of user 0. If user 0 and user 1 have the same average throughput, user 1 is going to be served because his achievable throughput is higher. That is what happens at millisecond 0.

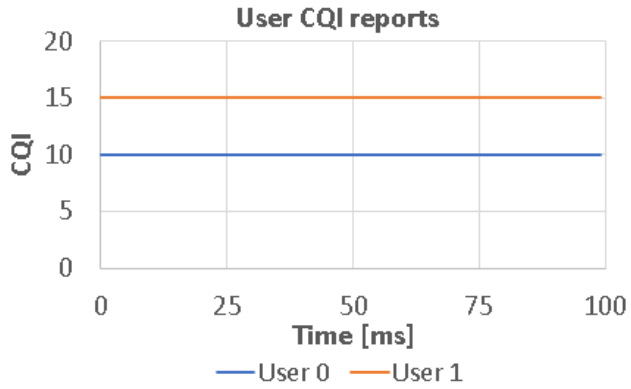

Figure 5.15 - CQIs reported by users along time for the PF test.

The average throughputs along time are presented in Figure 5.16 and the metrics in each TTI are presented in Figure 5.17. The user 1 has a higher average throughput. Although user 0 has always lower achievable throughput, he is served when his metric is greater than user 1's metric. If the metrics

are equal, the longest served user is served. As one can see in Figure 5.16, Proportional Fair algorithm avoids the starvation of the users with bad channel conditions.

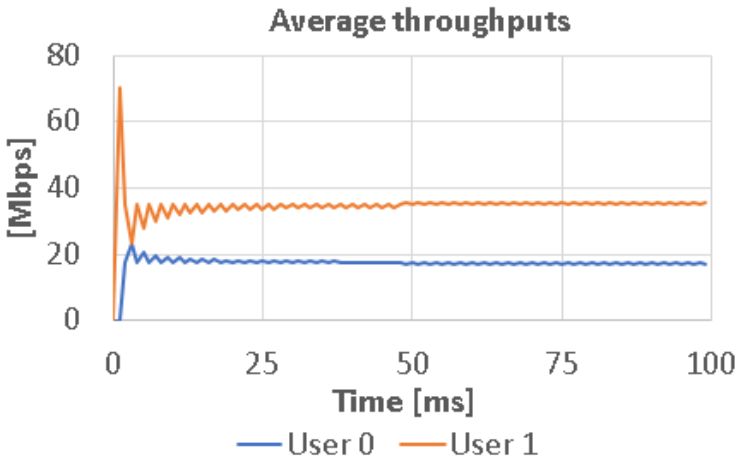

Figure 5.16 - Users' average throughputs for the PF test.

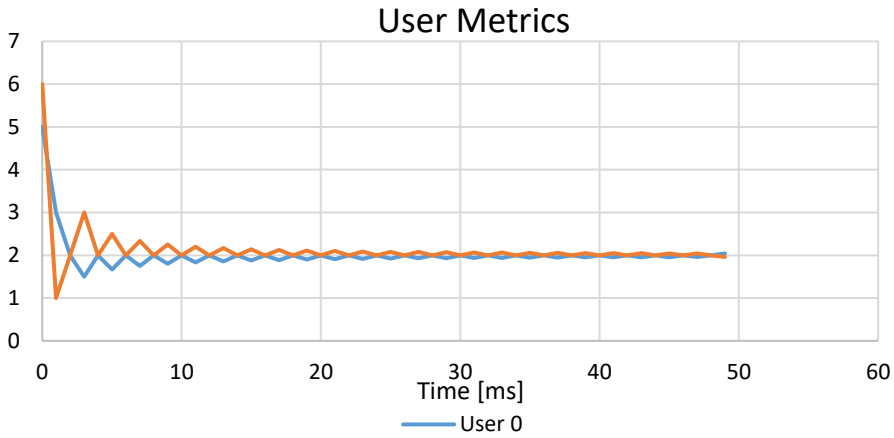

Figure 5.17 - User metrics for the PF test.

- **Proportional Fair with Barriers for Frames**

The before mentioned algorithms are not using the information the DASH clients report to the base station. They are not QoE-aware. In [40] it is presented an algorithm, namely PFBF, which is described in section 4.4.2. IT is QoE oriented by taking into account the re-buffering percentage and the buffer amount of data in the user-device. This algorithm aims at decreasing the re-buffering frequency which affects greatly the QoE. The frame size being downloaded by the i^{th} user, $S_{frame,i}$, is the number of bits of one second of video divided by the number of frames per second of the video, considering a constant bitrate. $S_{frame,i}$ is calculated using:

$$S_{frame,i} = \frac{U^{i}_{RequestedVideo}}{Video_{FrameRate}} \tag{5.10}$$

In the simulator, the buffer is measured in milliseconds and therefore f_{min} frames correspond to $\frac{f_{min}}{Video_{FrameRate}} * 1000$ milliseconds. Thus, f_j represents the number of milliseconds stored in the buffer. It is considered that all videos have a common frame rate of 30 frames per second.

The authors in [40] prove that PFBF is better than PF in terms of re-buffering percentage that users have when streaming video. A test is presented and the conclusion is the same. For 3 minutes, 80 users are streaming video, with the following bitrates {0.5, 0.6, 0.7, 1, 1.2 and 2} Mbps, implementing QAAD.

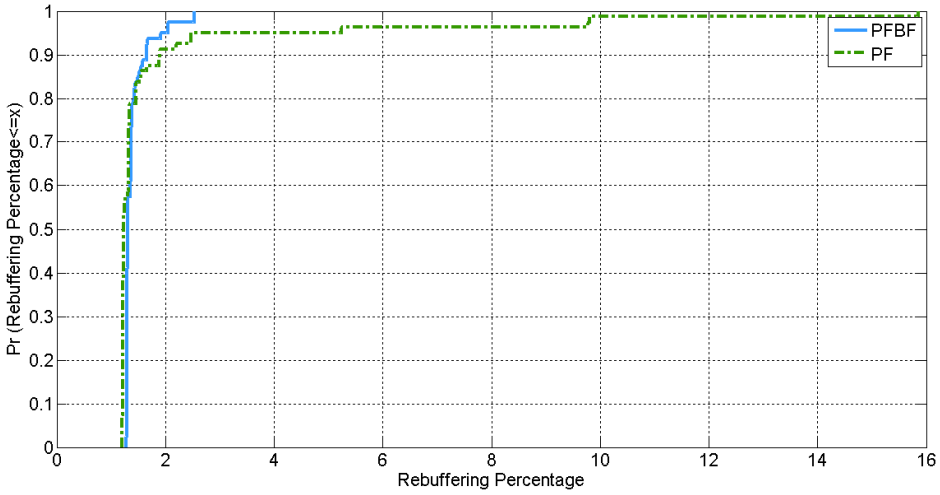

Figure 5.18 - CDF of re-buffering percentage for PFBF and PF algorithms.

PFBF ensure that all the 80 users have less than 3% of re-buffering percentage while PF have some users that have almost 16% of re-buffering percentage.

Chapter 6

Simulations and results

6.1 Introduction

This chapter presents the performance assessment methodology and the simulation parameters. Also, the assessment results are presented and discussed.

In section 6.2, it is described the Monte Carlo method that is used to guarantee with 95% of certainty that the real solutions do not differ by more than 1% from the presented values, because there are random variables that could greatly affect the results if a single simulation was made. In section 6.3, the values of the parameters used for simulating are presented. In section 6.4, it is presented a demonstration of the Method of Monte Carlo. In section 6.5, it is described the methodologies that are used to assess the performance of MBF and the already existing. The results are also presented and discussed. Finally, in section 6.6 it is presented a summary of relevant results and observations about this work.

6.2 Monte Carlo method

As it was stated in section 5.2, the reported channel quality indicators were simulated for 200 users using OMNET++. In the simulations, a single channel quality indicator from the 200 generated, is randomly assigned to each user. Given that this is a random process, the simulations are also random, especially in simulations with many users. The reason is that if there are only a few connected users, the algorithm can satisfy all of them very well because there are a lot of resources available. Consequently, to obtain statistically relevant results, the Monte Carlo Method is used, by running the simulation a given number of times to derive a final result that includes a mean value and a confidence interval. One single simulation would result in an unknown accuracy. Thus, to obtain a statistical representation of the real result, many simulations need to be runt. The calculation of the result is based on the Central Limit Theorem [54]. Consider a random variable X with a finite mean μ and variance σ^2. The theorem states that the arithmetic mean of a sufficiently large set of samples of the random variable $X = (X_1, X_2, \dots, X_n)$ of size n samples, defined by equation (6.1):

$$\hat{X} = \frac{1}{n}\sum_{i=1}^{s} X_i \tag{6.1}$$

approaches the random variable's mean as the number of samples, n, approaches infinity:

$$n \to \infty \Longrightarrow \hat{X} \to \mu \tag{6.2}$$

The samples' mean is the most probable solution and the standard deviation defines the confidence interval. The higher the number of simulations runs, the smaller the confidence intervals or the higher the accuracy of the solution. The minimum number of simulations, S_{min}, that need to be runt to achieve a certain target accuracy can be calculated using equation (6.3) [55]:

$$S_{min} = (\frac{z_{\alpha/2}}{w}\frac{\sigma}{\mu}) \quad (6.3)$$

where $z_{\alpha/2}$ is the value of the normal probability distribution function for the half distance $\alpha/2$ and w is the size of the confidence interval, normalized to the mean.

In this study, a maximum confidence interval size of 1% of the mean ($w = 0.01$), with a probability of 95% are assumed ($\frac{\alpha}{2} = 2.5\%$, corresponding to $z_{\alpha/2}$=1.96 as it can be seen in Figure 6.1). Thus, it can be affirmed with 95% of certainty that the real solution μ is in the interval [$\hat{x} - w\hat{x}, \hat{x} - w\hat{x}$], if the program is run at least S_{min} times. To calculate S_{min}, the values of σ and μ are needed. But these values are unknown too. So, this problem has to be solved iteratively. Firstly, an arbitrary large number of simulations is used for each number of connected users. After the simulations have been run for each number of users, the resulting μ and σ values are used to compute the minimum number of simulations necessary, S_{min}, using Equation (4.3). If S_{min} is larger than the number of simulations performed, S, the program must be run again for S_{min} simulations; if not, the confidence interval requirements are met and no further simulations are needed. The S_{min} computations are performed for all output variables and the confidence interval must be met by all of them. If $S < S_{min}$ for at least one variable, the simulation needs to be re-run.

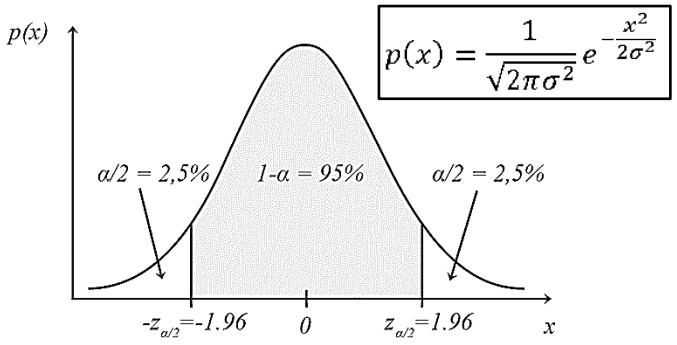

Figure 6.1 - Illustration of confidence intervals in the normalized normal distribution [56].

6.3 Simulation parameters

The simulation consists of three minutes of resources scheduling. During this period, 180 000 milliseconds, users are streaming video, being continuously downloading and playing the video sent by the base station. The number of connected users streaming video is constant. The video exists in the server in the following bitrates: {0.2, 0.25, 0.3, 0.4, 0.5, 0.7, 0.9, 1.2, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0, and 6.0} Mbps [50]. Users request always 1 second video segments. It is considered that there are not delays between the server and the base station and between the base station and the user. The data packets are sent instantaneously.

There are 100 resource blocks (bandwidth of 20 MHz) available. The resource blocks efficiencies used to calculate the throughputs depend on the CQI reported by the user every 5 ms, according to Table 4.2. Other tables similar to Table 4.1 are presented in [18], where different modulations are used. The number of bits transmitted per symbol (efficiency) would differ. However, this would affect all users independently of the allocation algorithm. The best algorithm would continue to have the best results as the worst algorithm would have the worst results and so on.

The DASH clients request segments using the algorithm QAAD, explained in the section 5.3, with the parameters $\theta = 0.3s, w = 0.875, \mu = 10s$, $\sigma = 3s$, as suggested in [51]. The initial delay and re-buffering periods are equal to the time that a user needs to download 5 seconds of the lowest video quality. During these periods, the buffers do not leak out 1 ms of video every millisecond spent in the simulation. When the users start the streaming or re-buffering after a stall, the users request 5 segments at the lowest quality level. Then, the requests are made according to QAAD.

The PF algorithm uses $\alpha = \beta = 1$, so user's throughput and average throughput influence equally the metric. For PFBF scheduler, it is used $\alpha = \beta = 1$ as suggested in [40]. The frame size being downloaded by the i^{th} user, $S_{frame,i}$, is the number of bits of 1 second of video divided by the number of frames per second. In this thesis, it is considered that all videos have a constant bitrate and a frame rate equal to 30 fps and thus the frame size, $S_{frame,i}$, is given by equation (6.4).

$$S_{frame,i} = \frac{U^{i}_{RequestedVideo}}{30} \tag{6.4}$$

It is used $f_{min} = 10$ frames, as suggested in [40]. In the program, the buffer is measured in milliseconds and thus $f_{min} = 10$ frames are $\frac{1}{3}$ of a second, approximately 333 milliseconds.

6.4 Monte Carlo method demonstration

Using the simulation parameters mentioned in section 6.3 and using the RR scheduling algorithm, it is plotted in the Figure 6.2 the percentage of connected users that have a global QoE ≥ 3 as a function of the number of connected users in the network for one single simulation and for at least the S_{min}simulations calculated using the Monte Carlo method to guarantee with 95% of certainty that the real solutions do not differ by more than 1% from the presented values.

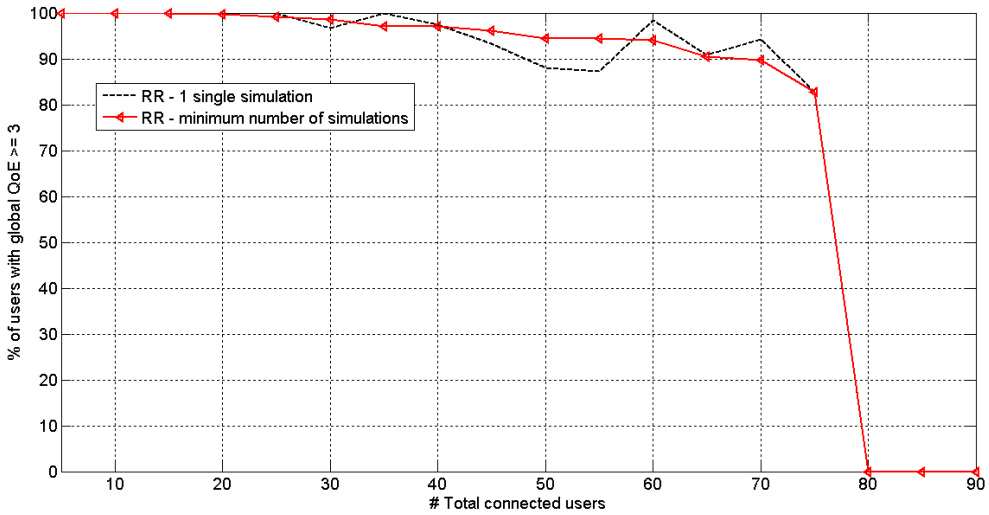

Figure 6.2 - Monte Carlo method demonstration using one single simulation and S_{min} simulations.

As it can be seen in Figure 6.2, performing a significant number of simulations greatly increases the results' accuracy, smoothing the curve. For each plotted point of the curve "RR – minimum number of simulations", the percentage of users with QoE ≥ 3 is calculated by averaging the values obtained in the simulations.

6.5 Algorithms comparison methodology

In this section, MBF will be compared with the RR, BET, PF and PFBF for the reasons explained in section 4.5. To assess the schedulers' performances, it is important to know both the number of satisfied and non-satisfied users at the end of their streaming sessions (duration of video downloaded plus initial delay and re-buffering periods). The term "global QoE" refers to the overall QoE that a user has with all the streaming session. Let's admit that a user is considered satisfied if his global QoE is above a certain limit U and non-satisfied if his global QoE is below a limit Z (with $Z < U$). For a given number of connected users, the percentage of satisfied and non-satisfied users is Y and W %, respectively. Note that the total number of users that can be streaming video in order to achieve a certain percentage of satisfied or non-satisfied ones, depends heavily on the number of video qualities and the range of bitrates stored in the server, according to the QoE model presented in section 5.4. However, the relative schedulers' performances do not depend on this.

In section 6.5.1, it is studied the algorithms' performances regarding the number of satisfied users. It is plotted the percentage of satisfied users as the total number of connected users increases. This way, an operator can adopt the scheduler that supports more users with QoE ≥ U for a given percentage Y % of satisfied users. However, it is also relevant to know how good or bad the remaining users are served. In this context, in section 6.5.2 it is studied the algorithms' performance regarding the number of non-satisfied users. This may help an operator to adopt a scheduler based on the supported number of satisfied users and/or level of fairness.

Before studying the schedulers' performances, the parameter α used by MBF must be chosen because it impacts both the results presented in sections 6.5.1 and 6.5.2. To do so, it is plotted the percentage of users with global QoE ≥ U, for U = 3 and U = 4, as a function of the number of connected users in the network streaming video for α[s] = {0, 2.5, 5, 7.5, 10, 11, 12.5, 15 and Infinity} (see Figure 6.3 and Figure 6.4). Only these values of α were tested for time simulation reasons. Then, it is also studied the percentage of users with global QoE < 2.

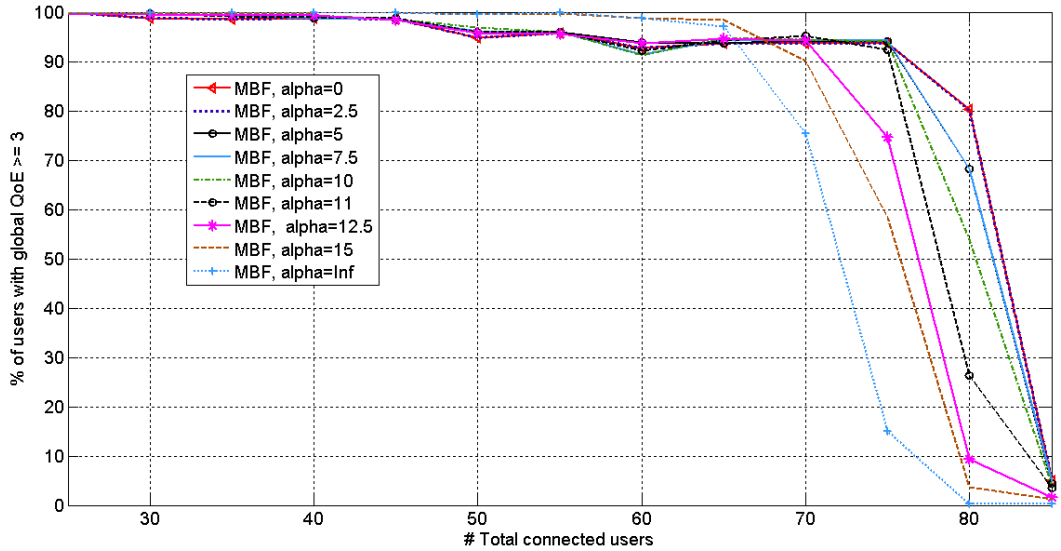

Figure 6.3 - Influence of the parameter α [s] in the percentage of users with global QoE ≥ 3.

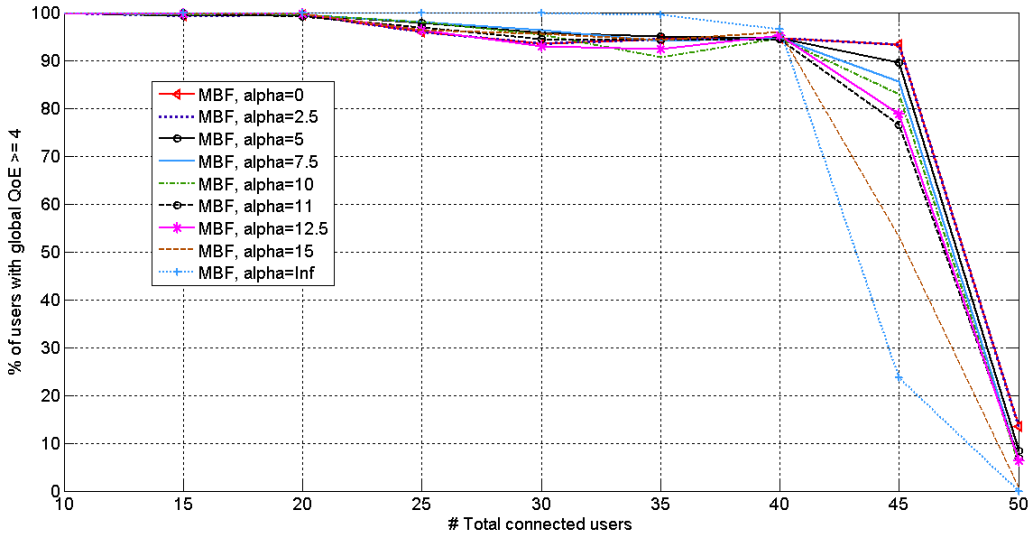

Figure 6.4 - Influence of the parameter α [s] in the percentage of users with global QoE ≥ 4.

In Figure 6.3 and Figure 6.4, for each plotted point of each curve, at least the S_{min}simulations calculated using the Monte Carlo method to guarantee with 95% of certainty that the real solutions do not differ by more than 1% from the presented values, were made. The percentages of the users with global QoE ≥ 3 and QoE ≥ 4 are calculated by averaging the values obtained in the simulations.

According to Figure 6.3 and Figure 6.4, for Y = 90, 80 and 70 %, the higher the value of α the higher the total number of users that can be connected streaming video. The values of α that maximize the number of satisfied users are 0 and 2.5 seconds. However, it is important to study also how good or bad the remaining users are served. In this context, in Figure 6.5, it is plotted the percentage of users with global QoE < 2 as a function of the total number of connected users.

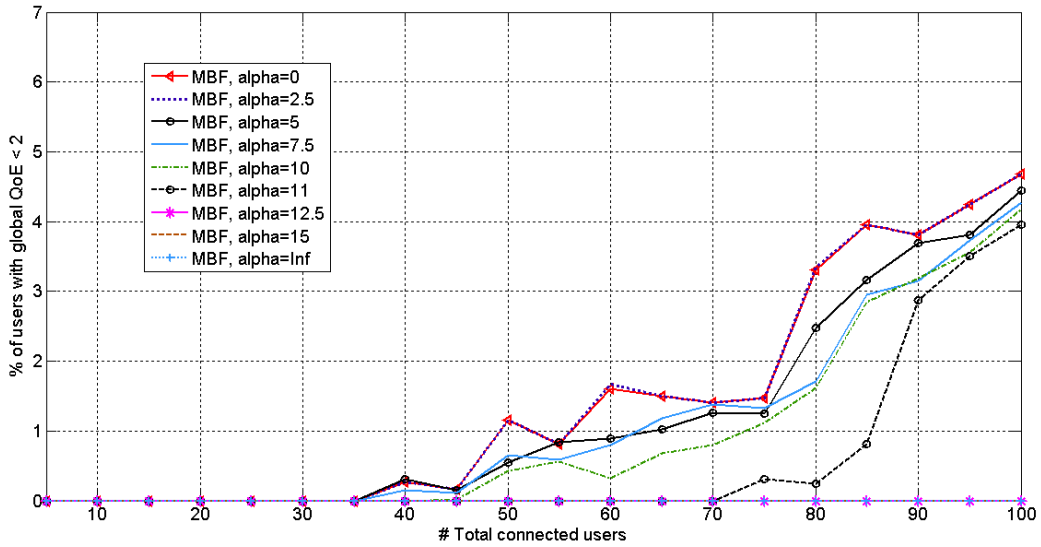

Figure 6.5 - Influence of the parameter α [s] in the percentage of users with global QoE < 2.

In Figure 6.5, the number of simulations S_{min} required were not made because it would not be possible to do so in a reasonable period. Still, it was made a sufficient high number of simulations to be possible to have an idea of the related position of the curves. The plotted points' x-coordinates for the three figures presented above, are multiples of 5 users because it is only required to know which α should be chosen, thus it is not necessary a high accuracy. According to Figure 6.5, the higher the value

of α, the lower the number of users with QoE < 2. As expected, the α values that showed to lead to a higher total number of connected users (α = 0s and α = 2.5s) have also a higher number of non-satisfied users, with QoE < 2, as it can be seen in Figure 6.5. This happens because when the algorithm enters in its emergency state it is going to serve the users just based on their buffers, without considering their throughputs and requested segment video qualities. By fulfilling the lowest buffer by serving the user until its buffer level is larger than α, the rest of the users experience lower throughputs and request lower video qualities according to the QoE adaptation algorithm the DASH clients implement, the QAAD. So α is a parameter whose value implies a trade-off between the number of satisfied users with QoE ≥ U and the non-satisfied users, with QoE < Z.

Table 6.1 presents in the second column the maximum number of users that can be streaming video, for which 90% of them have a global QoE ≥ 3, for each value of α. The column presents the maximum numbers of connected users which ensure that all users have global QoE ≥ 2. Finally, the capacities presented in the last column are the least of the values presented in the second and third columns. They are the maximum numbers of users that can be connected which ensure that 0% of them have global QoE < 2 and at least 90% have global QoE ≥ 3. The same information for U = 4 is not presented because the capacities differences between the α values are not so significant.

Table 6.1 - Capacity of MBF for different values of α for Z = 2, Y = 90%, U = 3 and W = 0%.

α [s]	Max. number of connected users for which 90% have global QoE ≥3 ($U = 3, Y = 90\%$)	Max. number of connected users for which 0% of them have global QoE <2 ($Z = 2, W = 0\%$)	Capacity
0	76.39	35	35
2.5	76.36	35	35
5	75.77	35	35
7.5	75.81	35	35
10	75.51	45	45
11	75.18	70	75.18
12.5	71.1	>100	71.1
15	70.02	>100	70.02
∞	71.63	>100	71.63

According to Table 6.1, α = 11 seconds is the tested values that leads to the highest capacity. However, to ensure 0% of the connected users have global QoE ≥ 2 may be too much severe, leading to small capacities. Thus, Table 6.2 presents the same information that Table 6.1 does, but for W=5%. The values of both tables are the average of the values obtained in the simulations.

Table 6.2 - Capacity of MBF for different values of α for W = 5%, Z = 2, Y = 90% and U = 3.

α [s]	Max. number of connected users for which 5% of them have global QoE <2 ($U = 3, Y = 90\%$)	Max. number of connected users for which 90% have global QoE >3 ($Z = 2, W = 0\%$)	Capacity
0	76.36	>100	76.36
2.5	76.36	>100	76.36
5	75.77	>100	75.78
7.5	75.81	>100	75.80
10	75.51	>100	75.51
11	75.18	>100	75.18
12.5	71.1	>100	71.1
15	70.02	>100	70.02
∞	71.63	>100	71.63

According to Table 6.1, 11 seconds is the tested α values that leads to the highest capacity. For W = 5%, 11 seconds is not the value that leads to the highest capacity. Depending on the value of W the most adequate value of α changes as it was shown. One can say that a higher value of α leads to a smaller number of users with global QoE<2 but also in a smaller number with global QoE≥3. This allows the algorithm to operate in two modes, according to the operator's intention:

- **Mode 1:** for $Y\%$ of users with global QoE ≥ U, this mode aims at maximizing the number of users with global QoE ≥ U (first priority), minimizing at the same time the number of users with global QoE < Z. For this mode, an appropriate α value is **5 s**, according to the study done previously. It leads to a slightly less number of satisfied users for U = 3 and U = 4 when compared to α = 0 s and α = 2.5 s (see Figure 6.3 and Figure 6.4) but it is clearly better at maintaining a low number of non-satisfied users (see Figure 6.5). For α = 7.5 s, the number of satisfied users is smaller than for α = 5 s (for U =4) and it is not clearly better at maintaining a lower number of non-satisfied users.
- **Mode 2:** this mode seeks to minimize the percentage of users with global QoE < Z (first priority), maximizing at the same time the number of users with global QoE ≥ U, for a given percentage $Y\%$ of users with global QoE ≥ U. For this mode, it is chosen α = **11 s** because it leads to a low number of non-satisfied users (see Figure 6.5) and still maintains a high number of satisfied users (see Figure 6.3 and Figure 6.4).

The parameter α is a tunable parameter that can be tuned to decrease the percentage of non-satisfied users for a given number of connected users streaming video, but decreasing also the percentage of satisfied users. In the context of this study, in sections 6.5.1 and 6.5.2 the analyzes are made for MBF operating in both modes to understand the trade-off regarding the mode choice.

6.5.1 Analysis of the number of satisfied users

In this section, different algorithms are compared regarding the number of satisfied users. For a given percentage $Y\%$ (90, 80 or 70%) of satisfied users, the algorithms under test support different numbers of users with a global QoE ≥ U. Particularly, it is plotted the percentage of satisfied users as the number of connected users increases, for U = 3, 3.5 and 4. In this analysis, MBF operating in mode 1 is expected to have better results than mode 2 because the first priority of the mode 1 is the number of satisfied users (global QoE ≥ U) whereas mode 2 is fairer, seeking for a smaller number of non-satisfied users (with global QoE < Z).

- **U = 3**

Figure 6.6 plots the percentage of users with QoE ≥ 3 as a function of the total number of users streaming video for the scheduling algorithms: RR, BET, PF, PFBF and both modes of MBF. As expected, the percentage of satisfied users decreases when the total number of connected users increases because the available radio resources are limited. From Figure 6.6, one can stablish a certain percentage of satisfied users (axis yy, "% of users with global QoE>=3") and check the total number of connected users that can be streaming video (axis xx, "# Total connected users"). By multiplying the

established percentage by the total number of connected users, it is obtained the number of satisfied users. In the same way, it is possible to check which algorithm has the highest percentage of satisfied users, for a certain total number of connected ones.

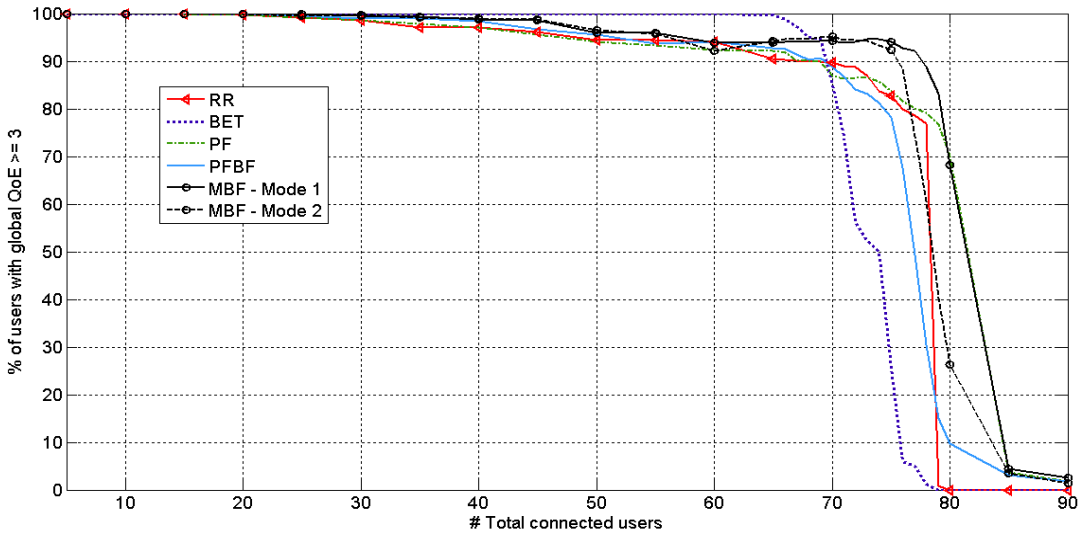

Figure 6.6 - Percentage of satisfied users as a function of connected users streaming video (U = 3).

For each curve presented in Figure 6.6, at least the S_{min} simulations, calculated using the Monte Carlo method 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 percentage of users with global QoE ≥ 3 is the average of the values obtained in the simulations. The plotted points whose x-coordinate is between 65 and 80 (inclusive) are distanced from 1 by 1 to achieve a high accuracy in the calculation of the number of satisfied users for Y= 90, 80 or 70%. The remaining plotted points' x-coordinates are multiples of 5 users from 5 to 60 and 85 to 90 (inclusive).

Figure 6.7 presents the number of satisfied users (U = 3) for all the algorithms for some percentages of satisfied users that may be interesting (Y = 90, 80 and 70%). These values were calculated doing a linear interpolation between two points.

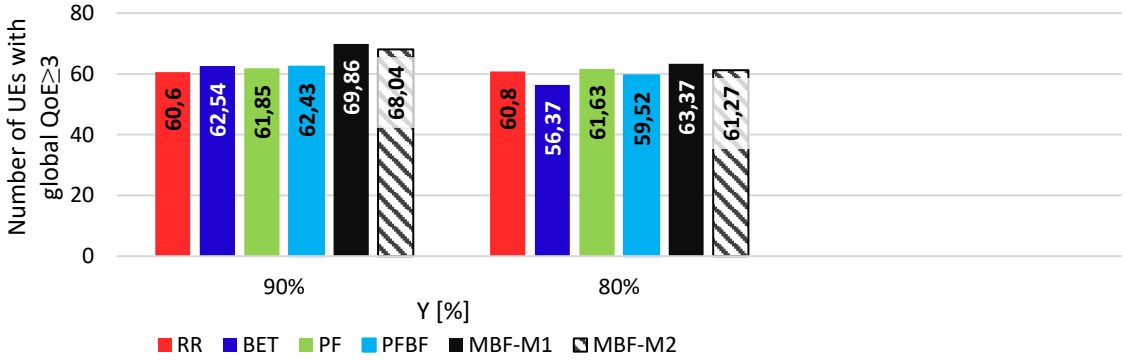

Figure 6.7 - Number of users with global QoE ≥ 3 for Y = 90, 80 and 70% for each algorithm.

According to Figure 6.7, in order to satisfy 90 % of the connected users (when the criterion of satisfaction is to have a global QoE ≥ 3) the scheduling algorithm that supports the largest number of connected users between those that are being tested, is the MBF scheduler operating in mode 1, with

the possibility of having on average 69.86 satisfied users, followed by MBF operating in mode 2, supporting on average 68.04 satisfied users.

For 80%, the allocation algorithm that leads to the highest capacity is MBF operating in mode 1, supporting on average 63.37 satisfied users. The second one is MBF operating in mode 2 and this scheduler supports on 61.27 satisfied users.

For 70%, MBF operating in mode 1 is also the scheduler that leads to the highest capacity, supporting an average of 55.92 satisfied users. The second one is the RR supporting on average 54.66 satisfied users.

Table 6.3 presents for each algorithm the number of satisfied users, the total number of connected users and the capacity gains from to the RR, for U = 3 and Y = 90, 80 and 70%.

Table 6.3 - Algorithms' capacities and gains with respect to Round Robin for U = 3 and Y = 90, 80 and 70%.

# Satisfied UEs # Total connected UES (Gain [%])		**Algorithm**					
		RR	BET	PF	PFBF	MBF Mode 1	MBF Mode 2
Y [%]	90	**60.60** 67.33 (0%)	**62.54** 69.49 (+3.20%)	**61.85** 68.72 (+2.06%)	**62.43** 69.36 (+3.02%)	**69.86** 77.62 (+15.28%)	**68.04** 75.60 (+12.28%)
	80	**60.80** 75.99 (0%)	**56.37** 70.46 (-7.86%)	**61.63** 77.03 (+1.37%)	**59.52** 74.40 (-2.15%)	**63.37** 79.21 (+4.23%)	**61.27** 76.59 (+0.77%)
	70	**54.66** 78.09 (0%)	**49.87** 71.24 (-9.60%)	**56.01** 80.01 (+2.47%)	**53.04** 75.77 (-3.05%)	**55.92** 79.88 (+2.31%)	**54.12** 77.31 (-1.00%)

According to Table 6.3, MBF operating in mode 1 has the highest gains for the Y percentages under test, supporting 15.28% more satisfied users than RR does for Y = 90%. It is also the only algorithm with positive gains for Y = 90, 80 and 70%. Although MBF operating in mode 2 is not optimized for achieving the highest possible number of satisfied users, it is still better than RR, BET and PF for Y = 90%. The BET scheduler has the lowest values for approximately Y > 95 % but leads clearly to the highest capacities for approximately Y < 90 % (see Figure 6.10) because it serves the users with the same global QoE. PFBF whose metric is based on PF's metric, is slightly better than PF for Y > 90 %. RR's performance is better than expected. Its performance would be worse in a scenario where the users' channel conditions varied more significantly along time. It is also important to mention that the algorithms are very sensitive because serving a slightly larger number of users may imply a large percentage loss of satisfied ones.

- **U = 3.5**

Figure 6.8 plots the percentage of users with QoE ≥ 3.5 as a function of the total number of users streaming video. For each curve presented, at least the S_{min}simulations, calculated using the Monte Carlo method 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 percentage of users with global QoE ≥ 3.5 is calculated by averaging the values obtained in the simulations. The plotted points whose x-coordinate is between 46 and 59 (inclusive) are distanced from 1 by 1 to achieve a high accuracy in the calculation of the number of satisfied users for Y = 90, 80 or 70%. The remaining plotted points' x-coordinates are multiples of 5 users from 5 to 45 and 60 to 70.

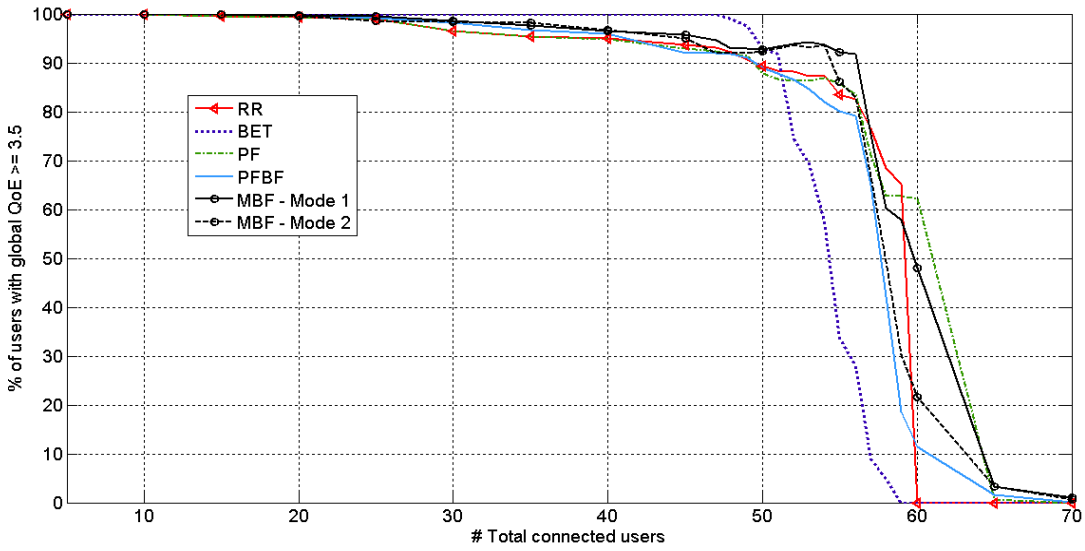

Figure 6.8 - Percentage of satisfied users as a function of connected users streaming video (U = 3.5).

Figure 6.9 presents the number of satisfied users (for U = 3.5) for all the algorithms for Y = 90, 80 and 70%. These values were calculated doing a linear interpolation between two points.

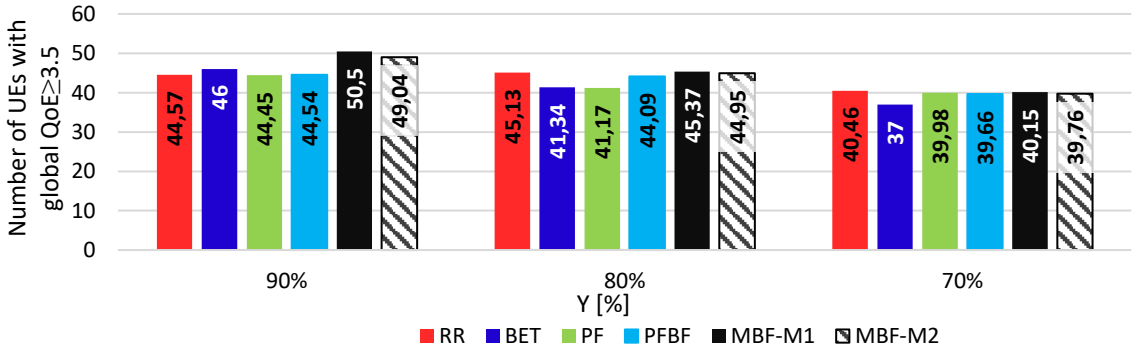

Figure 6.9 - Number of users with global QoE ≥ 3.5 for Y = 90, 80 and 70% for each algorithm.

According to Figure 6.9, in order to satisfy 90 % of the connected users (when the criterion of satisfaction is to have a global QoE ≥ 3.5) the scheduling algorithm that supports the largest number of connected users between those that are being tested, is the MBF scheduler operating in mode 1, with the possibility of having on average 50.5 satisfied users, followed by MBF operating in mode 2, supporting on average 49.04 satisfied users.

For 80%, the scheduler that leads to the highest capacity is MBF operating in mode 1, supporting on average 45.37 satisfied users. The second one is RR, supporting on average 45.13 satisfied users.

For 70%, RR is the scheduler that leads to the highest capacity, supporting on average 40.46 satisfied users, followed by MBF operating in mode 1, supporting on average 40.15 satisfied users.

Table 6.4 presents for each algorithm the number of satisfied users, the total number of connected users and the capacity gains with respect to the RR, for U = 3.5 and Y = 90, 80 and 70%.

Table 6.4 - Algorithms' capacities and gains with respect to Round Robin for U = 3.5 and Y=90, 80 and 70%.

# Satisfied UEs # Total connected UES (Gain [%])		**Algorithm**					
		RR	BET	PF	PFBF	MBF Mode 1	MBF Mode 2
Y [%]	90	**44.57** 49.52 (0%)	**46.00** 51.11 (+3.21%)	**44.45** 49.39 (-0.27%)	**44.54** 49.49 (-0.07%)	**50.50** 56.11 (+13.3%)	**49.04** 54.49 (+10.03%)
	80	**45.13** 56.41 (0%)	**41.34** 51.68 (-9.17%)	**45.02** 56.28 (-0.24%)	**44.09** 55.12 (-2.36%)	**45.37** 56.72 (+0.53%)	**44.95** 56.18 (-0.4%)
	70	**40.46** 57.80 (0%)	**37.00** 52.86 (-9.35%)	**39.98** 57.12 (-2.74%)	**39.66** 56.65 (-2.02%)	**40.15** 57.36 (-0.77%)	**39.76** 56.80 (-1.76%)

According to Table 6.3, MBF operating in mode 1 is the scheduler with the highest gain for the Y percentages under test, allowing 13.3% more satisfied users than RR does, for Y = 90%.

- **U=4**

Figure 6.10 presents the percentage of connected users with global QoE ≥ 4 as the total number of connected users streaming video increases.

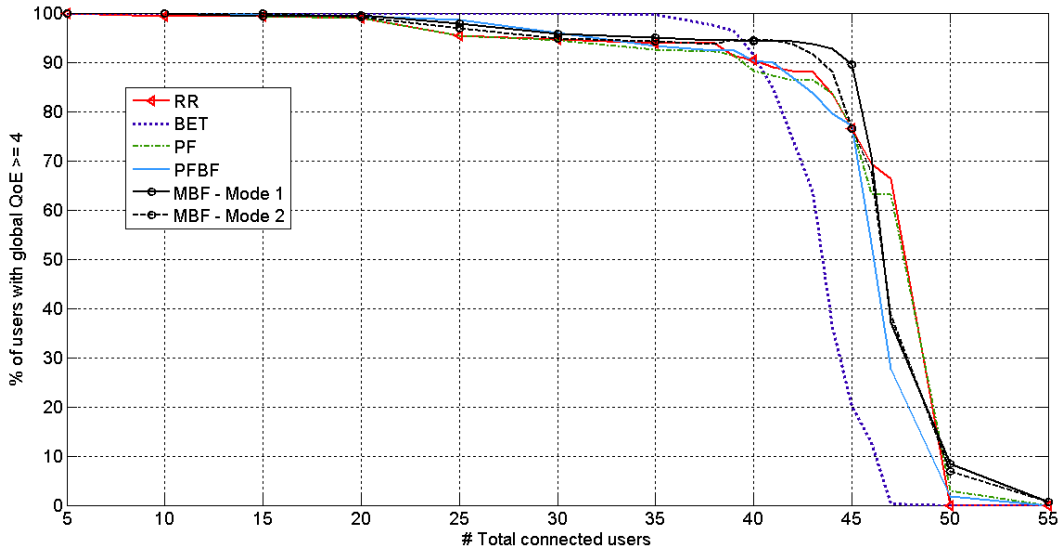

Figure 6.10 - Percentage of satisfied users as a function of connected users streaming video (U = 4).

For each curve presented in Figure 6.10, at least the S_{min} simulations, calculated using the Monte Carlo method 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 whose x-coordinate is between 38 and 47 (inclusive) are distanced from 1 by 1 to achieve a high accuracy in the calculation of the number of satisfied users for Y= 90, 80 or 70%. The remaining

plotted points' x-coordinates are multiples of 5 users from 5 to 35 and 50 to 55 (inclusive). For each plotted point, the percentage of users with global QoE ≥ 4 is calculated by averaging the values obtained in the simulations.

Figure 6.11 presents the number of satisfied users for all the algorithms for Y = 90, 80 and 70%. The presented values were calculated doing a linear interpolation between two points.

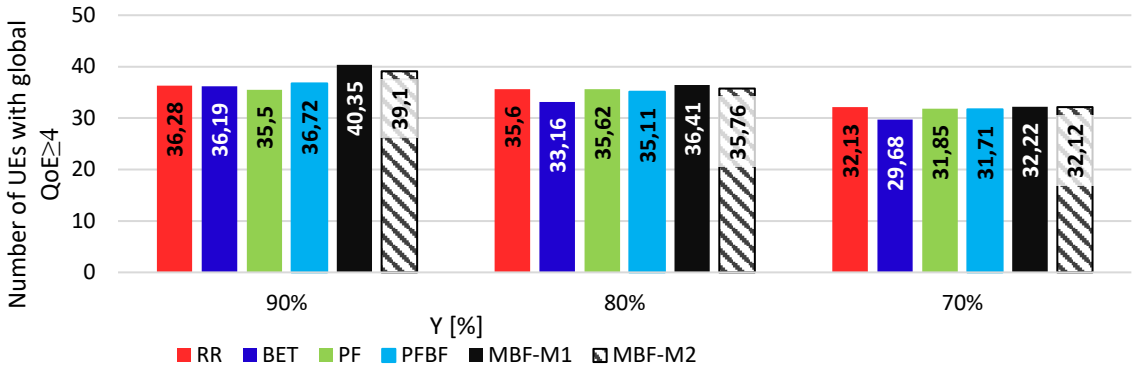

Figure 6.11 - Number of users with global QoE ≥ 4 for Y = 90, 80 and 70% for each algorithm.

As it is possible to see in Figure 6.11, in order to satisfy 90 % of the connected users (when the criterion of satisfaction is to have a global QoE ≥ 4), the scheduling algorithm that supports the largest number of connected users between those that are being tested, is the MBF scheduler operating in mode 1, with the possibility of having on average 40.35 satisfied users, followed by MBF operating in mode 2, supporting on average 39.10 satisfied users.

For 80%, the allocation algorithm that leads to the highest capacity is MBF operating in mode 1, supporting on average 36.41 satisfied users. The second one is MBF operating in mode 2 and this scheduler supports on average 35.76 satisfied users.

For 70%, MBF operating in mode 1 is the scheduler that has the highest capacity, supporting on average 32.22 satisfied users followed by RR supporting on average 32.13 satisfied users.

Table 6.5 presents for each algorithm the number of satisfied users, the total number of connected users and the capacity gains with respect to the RR, for U = 4 and Y = 90, 80 and 70%.

Table 6.5 - Algorithms' capacities and gains with respect to Round Robin for U = 4 and Y = 90, 80 and 70%.

# Satisfied UEs # Total connected UES (Gain [%])		**Algorithm**					
		RR	BET	PF	PFBF	MBF Mode 1	MBF Mode 2
Y [%]	90	**36.28** 40.31 (0%)	**36.19** 40.21 (-0.25%)	**35.50** 39.45 (-2.20%)	**36.72** 40.80 (+1.21%)	**40.35** 44.83 (+11.22%)	**39.10** 43.44 (+7.77%)
	80	**35.60** 44.51 (0%)	**33.16** 41.46 (-7.36%)	**35.62** 44.53 (+0.06%)	**35.11** 43.89 (-1.40%)	**36.41** 45.51 (+2.28%)	**35.76** 44.70 (+0.44%)
	70	**32.13** 45.90 (0%)	**29.68** 42.41 (-8.25%)	**31.85** 45.50 (-0.88%)	**31.71** 45.30 (-1.32%)	**32.22** 46.03 (+0.28%)	**32.12** 45.89 (-0.03%)

According to Table 6.5, MBF operating in mode 1 is the scheduler algorithm with the highest gains with respect to RR and the only one with positive gains for Y = 90, 80 and 70%. Although MBF operating in mode 2 is not optimized for achieving the highest possible number of satisfied users, it is still better than RR, BET and PF for Y = 90, 80 and 70%, allowing 7.77 % more satisfied users than RR does for Y = 90%.

BET has the lowest values for approximately $Y < 85$ % but leads clearly to the highest capacities for approximately $Y > 95\%$ (see Figure 6.10) because it is the fairest scheduler by providing the same QoE to all users, regardless their radio channel conditions.

6.5.2 Analysis of the percentage of users with global QoE < 2

In section 6.5.1, different schedulers were compared regarding the number of satisfied users who have global QoE ≥ U. However, an operator may be also interested in knowing how well or bad the remaining $(100 - Y)$ % users (the non-satisfied ones, with QoE below the threshold Z) are served. In this context, this section analyzes the percentage of users with a global QoE < 2 (poor and bad QoE) as a function of the total number of connected users streaming video. Figure 6.12 compares MBF operating in its both modes, RR, BET, PF and PFBF regarding the percentage of users with QoE < 2 as the total number of connected users increases.

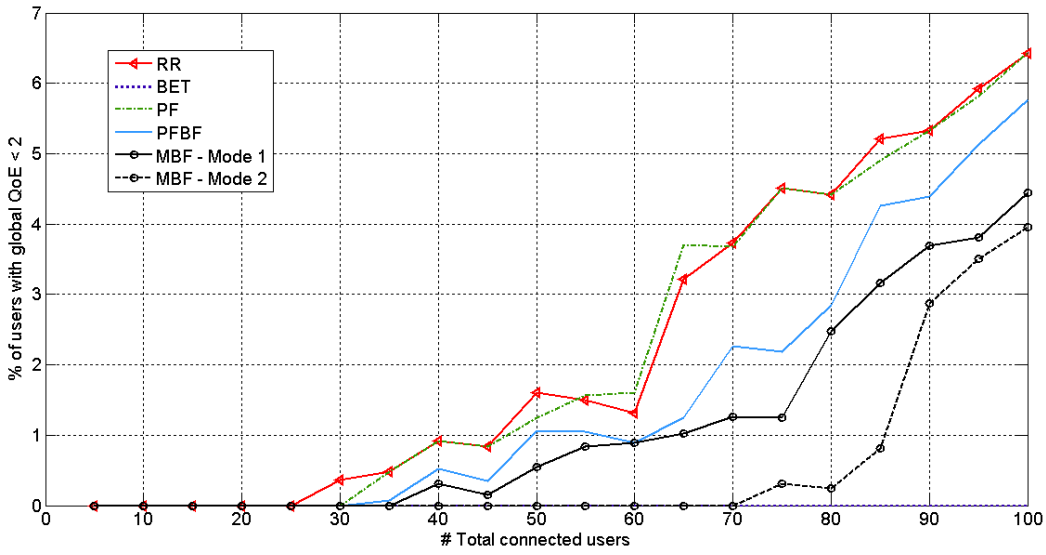

Figure 6.12 - Percentage of users with global QoE < 2 as a function of connected users streaming video.

In Figure 6.12, the number of simulations required was very high so it would not be possible to do so in a reasonable period. Still, it was made a sufficient high number of simulations to be possible to know the relative position between the curves. The plotted points' x-coordinates are multiples of 5 users from 5 to 100. According to Figure 6.12, as expected BET is the algorithm that, for a given number of connected users between 5 and 100, leads to the lowest number of users with QoE < 2. BET support more than 100 connected users streaming video with QoE ≥ 2, because it serves all users with the same global QoE by providing them with the same average throughput along time. The second one is the MBF operating in mode 2, the mode that tries indeed to minimize the number of users with a bad or poor quality of experience streaming video, by using the adjustable parameter $\alpha = 11$ s. MBF operating in

mode 2 allows a percentage of non-satisfied users smaller than 1% until there are 85 users streaming video which is a higher number of users than the capacities calculated in section 6.5.1 (see Table 6.3, Table 6.4 and Table 6.5). This positive property of this mode is just possible by having a lower number of users with QoE ≥ U when compared to the same algorithm operating in mode 1, as it was seen in section 6.5.1. Still, even with very low percentages of non-satisfied users (less than 3 %), regardless MBF's mode, it supports a higher number of satisfied users than BET for U = 3, 3.5 and 4 and Y = 90, 80 and 70% (see Table 6.3, Table 6.4 and Table 6.5). MBF, regardless of the mode in which it operates, guarantees clearly a lower number of users with QoE < 2 than RR, PF and PFBF and supports always more satisfied users for Y = 90%.

Table 6.6 presents the maximums capacities that are imposed by W%, for Z=2 and W = 0%, 3% and 5%. To guarantee that less than W % of users have global QoE < 2, the total number of connected users streaming video cannot exceed the values presented in Table 6.6. For instance, if the operator wants to guarantee that at least Y % of the connected users have a global QoE ≥ 3 and no more than W % of them have global QoE < 2, the maximum capacity is given by the smaller of the numbers presented in Table 6.3 and Table 6.6. The values presented in Table 6.6 were calculated doing a linear interpolation between two points.

Table 6.6 - Maximum capacities imposed by the value of W %.

Maximum capacity imposed by W, for Z=2		Algorithm					
		RR	BET	PF	PFBF	MBF Mode 1	MBF Mode 2
W [%]	0	25	>100	30	30	35	70
	3	64.43	>100	63.32	80.55	83.73	91.02

According to Table 6.6, if an operator wants to ensure that any of the connected users have a global QoE < 2 (W = 0%), RR and PF maximum capacites are approximately 25 and 30 users, respectively, whereas MBF operating in mode 2 allows a maximum capacity of 70 users.

The MBF, PFBF and BET maximum capacities that ensure that a maximum of 3 % of the connected users have a global QoE < 2 and at least Y % (for Y = 90, 80 and 70%) of them have have a global QoE > U (for U = 3, 3.5 and 4) are the capacities presented in the Table 6.3, Table 6.4 and Table 6.5. The RR and PF maximum capacities for U = 3 and Y = 90, 80 or 70 % are approximately 64.43 and 63.32 users, respectively. For U = 3.5 and U = 4, RR and PF capacities are also the ones presented in the Table 6.4 and Table 6.5.

For a percentage of users with global QoE < 2 greater than 5 %, all algorithm's capacities are given by Table 6.3, Table 6.4 and Table 6.5, for Y ≥ 70%.

BET supports more than 100 connected users providing to all them at least a fair global QoE in their streaming sessions.

6.6 Results summary

In this section, it is presented the most important results and some relevant and interesting observations from the study made in this thesis:

- Blind Equal Throughput is the fairest scheduling algorithm in terms of users' QoE. It allows the users to have the same QoE regardless their radio channel conditions, as long as they implement the same rate adaptation algorithm for requesting video segments.
- Blind Equal Throughput is the algorithm that allows more users to be streaming video until there is one or more users non-satisfied, as long as they implement the same rate adaptation algorithm for requesting video segments.
- MBF operating in mode 1 supports a larger number of connected users than RR, BET, PF and PFBF, if one wants to have 90% of the users with a good or excellent global QoE. It has a gain of approximately 15 % with respect to RR scheduler.
- MBF regardless the mode is still better than RR, PF and PFBF at guaranteeing a minimum number of users with a poor or bad QoE.
- MBF has the same computational complexity than RR, BET, PF and PFBF. MBF have access to the reported buffer levels by the users implementing the MPEG-DASH specification.
- RR's performance is all better in terms of QoE level fairness, the more similar are the channel qualities of the users, as long as they implement the same rate adaptation algorithm for requesting video segments.
- Although it is not part of the main study of this thesis, users have higher QoEs if they request segments with lower lengths because if the channel conditions change, they can request faster for other video segments with lower or higher quality.

Chapter 7

Conclusions

In today's world, people are getting more and more demanding in terms of the quality of experience they have using a service they pay for. If the service does not provide them the quality of experience they are expecting for, they simply stop paying for it and search for a new service provider. Furthermore, the exponential growth of consumed mobile multimedia content may congest the network. This way, it is of great interest of the operators to provide a service that satisfies the highest possible number of users. To do so, the development of intelligent and efficient scheduling algorithms may play an important role. A lot of research has been done in this field, however very few schedulers are analyzed regarding the provided network capacity in terms of the number of satisfied and non-satisfied users. The objective of the present dissertation was to study the MPEG-DASH specification, the already existing scheduling algorithms in the literature and to develop a scheduler that used the reported metrics by the clients which implement the MPEG-DASH specification to maximize the number of satisfied users while maintaining also a high QoE fairness level between all the connected users.

The first challenge was to implement the scenario, particularly to decide what QoE model to use to measure the QoE of the users and the QoE adaptation algorithm, responsible for the users' video segment requests. After that, there was a difficulty in deciding what would be the best methodology to compare the scheduling algorithms because this would define the strategy that should be followed to develop the scheduling algorithm. The algorithms are compared regarding the number of satisfied users streaming video while an analysis of the non-satisfied users with a poor or bad QoE is also done. This way, not only the maximum capacities of the algorithms are studied but also their QoE fairness levels. To assess the performances of the algorithms, a simulator was developed implementing a LTE network scenario where several mobile users are streaming video, reporting their time-varying radio channel conditions and their buffer levels QoE metric. They request video segments using a QoE adaptation algorithm (QAAD) and their satisfactions is measured using a model that predicts the QoE, ranging from 0 to 5.

The proposed algorithm, Maximum Buffer Filling, makes use of the buffer level, a QoE metric reported by the users who implement the MPEG-DASH specification. It allocates the resources based on the users' buffer levels and their achievable buffer fillings. The MBF scheduler can operate in two modes, according to an adjustable parameter that tunes the QoE level fairness between the users. One of them has as first priority to maximize the number of satisfied users with a good or excellent QoE. The second one minimizes the number of non-satisfied users with a poor or bad QoE.

MBF regardless the mode in which it operates has shown positive results by providing higher capacities than RR, PF, PFBF and BET, for a 90% of satisfied users. It has a gain between 7.77% and 15.28% with respect to RR, regarding the number of users with a good or excellent QoE. Furthermore, it leads to a lower percentage of users with a bad and poor QoE than RR, PF and PFBF. For a 90 % of users with a good or excellent QoE, it leads to a percentage less than 1% of non-satisfied users. As expected, BET has shown to be better at guaranteeing a minimum number of non-satisfied users.

For future work it would be interesting to study the sensibility of the capacities of the algorithms to the QoE model and to the number of video qualities stored in the server. Furthermore, it would be interesting to analyze the QoE along the streaming session, analyzing the possibility of having users giving up from their streaming sessions.

References

[1] "Rohde & Schwarz Mobile Network Testing", [Online] Available: http://www.mobile-network-testing.com/en/expertise/testing-qos--qoe/overview-qos-and-qoe/ . [Accessed: 20 April 2017].

[2] Qualinet, "Qualinet White Paper on Definitions of Quality of Experience", 2013.

[3] R. Mok, E. Chan, R. Chang, "Measuring the Quality of Experience of HTTP Video Streaming", IEEE, Hong Kong, China, 2011.

[4] C. Mueller, S. Lederer, C. Timmerer, ITEC - Dynamic Adaptive Streaming over HTTP, VLC Plugin, DASHEncoder and Dataset.

[5] A. Salo, MPEG DASH: A Technical Deep Dive and Look at What's Next, Chicago, USA: NCTA, 2012.

[6] R. Pereira, E. Pereira, "Dynamic Adaptive Streaming over HTTP and Progressive Download: Comparative Considerations", 28th IEEE International Conference on Advanced Information Networking and Applications, AINA 14, pg. 905-909, Victoria, Canada, May 2014.

[7] I. Sodogar , "The MPEG-DASH Standard for Multimedia Streaming Over the Internet", IEEE Multimedia, vol.18, nº4 , pp 62-67 , 2011.

[8] ISO/IEC, "Information technology — Dynamic adaptive streaming over HTTP (DASH) —Part 1:Media presentation description and segment formats", 2014.

[9] 3rd Generation Partnership Project, "3GPP TS 26.247 V14.1.0." , 2017.

[10] IETF RFC 2616, Hypertext Transfer Protocol – HTTP/1.1, June 1999.

[11] "Transmission Control Protocol", [Online] Available: https://pt.wikipedia.org/wiki/Transmission_Control_Protocol, [Accessed 4 March 2017].

[12] International Telecommunication Union, "Methodology for the Subjective Assessment of the Quality of Television Pictures" , Geneva, Switzerland, 2003.

[13] International Telecommunication Union, "Subjective video quality assessments methods for multimedia application", ITU-T Recommendation P.910, 1999.

[14] H. Pascal, "An Innovative Tool for Measuring Video Streaming QoE", [Online] Available: https://insight.nokia.com/innovative-tool-measuring-video-streaming-qoe , [Accessed: 16 3 2017].

[15] O. Oyman, S. Singh , "Quality of experience for HTTP adaptive streaming services", IEEE Communications Magazine, vol. 50, nº 4, 2012..

[16] ISO/IEC JTC 1, ISO/IEC CD 23001-10 Information technology — MPEG systems technologies — Part 10: Carriage of Timed Metadata Metrics of Media in ISO Base Media File Format, 2014.

[17] Erik Dahlman, Stefan Parkvall, and Johan Sköld, 4G LTE / LTE Advanced for Mobile Broadband, Academic Press is an imprint of Elsevier, 2011.

[18] 3rd Generation Partnership Project, 3GPP TS 36.213 V14.3.0 (2017-06-23) : Physical Layer Procedures (Release 14).

[19] M. Cardei, I. Cardei, D. Du, "Resource Management in Wireless Networking", 2005.

[20] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, “Downlink Packet Scheduling in LTE Cellular Networks: Key Design Issues and a Survey,” IEEE Communications Surveys & Tutorials, vol. 15, no. 2, pp. 678–700, Second Quarter 2013.

[21] S. Mumtaz and J. Rodriguez (eds.), Smart Device to Smart Device Communication, DOI:10.1007/978-3-319-04963-2_1,Springer International Publishing Switzerland 2014.

[22] S. Sudheep, B. Rebekka, "Proportional Equal Throughput Scheduler- A Very Fair Scheduling Approach in LTE Downlink", International Conference on Information Communication and Embedded Systems, Tamil Nadu, India, 2014.

[23] G. Monghal, K. I. Pedersen, I. Z. Kovacs, and P. E. Mogensen, “QoS oriented time and frequency domain packet schedulers for the UTRAN long term evolution,” in Proc. of IEEE Veh. Tech. Conf., VTC-Spring , Marina Bay, Singapore, May 2008.

[24] R. Basukala, H. Mohd Ramli, and K. Sandrasegaran, “Performance analysis of EXP/PF and M-LWDF in downlink 3GPP, LTE system,” in Proc. of First Asian Himalayas International Conf. on Internet, AH-ICI, Kathmundu, Nepal, Nov. 2009,pp. 1 –5.

[25] B. Sadiq, R. Madan, and A. Sampath, “Downlink scheduling for multiclass traffic in lte,”EURASIP J. Wirel. Commun. Netw., vol. 2009, pp. 9–9, 2009..

[26] I. Sousa, M. Queluz and A. Rodrigues, "A Survey on QoE-oriented Wireless Resources Scheduling", IEEE Communications Surveys and Tutorials (submitted).

[27] M. Shehada, S. Thakolsri, Z. Despotovic, W. Kellerer, “QoE-based Cross-Layer Optimization for video delivery in Long Term Evolution mobile networks,” in 14th International Symposium on Wireless Personal Multimedia Communications (WPMC 2011), October 2011.

[28] S. Thakolsri, S. Cokbulan, D. Jurca, Z. Despotovic, and W. Kellerer, “QoE-driven cross-layer optimization in wireless networks addressing system efficiency and utility fairness,” in 2011 IEEE GLOBECOM Workshops, December 2011, pp. 12–17.

[29] T. Wirth, Y. Sanchez, B. Holfeld, and T. Schierl, “Advanced Downlink LTE Radio Resource Management for HTTP-streaming,” in 20th ACM International Conference on Multimedia, October/Nobember 2012, pp.1037–1040.

[30] M. Seyedebrahimi, X. H. Peng, and R. Harrison, “Adaptive Resource Allocation for QoE-Aware Mobile Communication Networks,” in 17th IEEE International Conference on Computational Science and Engineering (CSE 2014), December 2014, pp. 868–875.

[31] M. Nasimi, M. Kousha, and F. Hashim, “QoE-oriented cross-layer downlink scheduling for heterogeneous traffics in LTE networks,” in IEEE Malaysia International Conference on Communications (MICC 2013), November 2013, pp. 292–297.

[32] K. Piamrat, K. D. Singh, A. Ksentini, C. Viho, and J. M. Bonnin, “QoE-Aware Scheduling for Video-Streaming in High Speed Downlink Packet Access,” in 2010 IEEE Wireless Communication & Networking Conference (WCNC 2010), April 2010.

[33] S. Mohamed and G. Rubino, “A Study of Real–time Packet Video Quality Using Random Neural Networks,” in IEEE Trans. On Circuits and Systems for Video Tech., vol. 12, no. 12, pp. 1071–1083, Dec. 2002.

[34] J. Navarro-Ortiz, P. Ameigeiras, J. M. Lopez-Soler, J. Lorca-Hernando,Q. Perez-Tarrero, and R. Garcia-Perez, “A QoE-Aware Scheduler for HTTP Progressive Video in OFDMA Systems,” IEEE Communications Letters, vol. 17, no. 4, pp. 677–680, April 2013.

[35] P. Chandur and K. M. Sivalingam, “Quality of experience aware video scheduling in LTE networks”, in 20th National Conference on Communications (NCC 2014), February 2014.

[36] P. C. Hsieh and I. H. Hou, “Heavy-Traffic Analysis of QoE Optimality for On-Demand Video Streams Over Fading Channels,” in 35th IEEE International Conference on Computer Communications (INFOCOM 2016), April 2016.

[37] Y. Ju, Z. Lu, W. Zheng, X. Wen, and D. Ling, “A cross-layer designfor video applications based on QoE prediction,” in 15th InternationalSymposium on Wireless Personal Multimedia Communications (WPMC 2012), September 2012, pp. 534–538.

[38] F. Wamser, D. Staehle, J. Prokopec, A. Maeder, and P. Tran-Gia,“Utilizing buffered YouTube playtime for QoE-oriented scheduling inOFDMA networks,” in 24th International Teletraffic Congress (ITC 24), September 2012.

[39] K. Miller, D. Bethanabhotla, G. Caire, and A. Wolisz, “A Control-Theoretic Approach to Adaptive Video Streaming in Dense Wireless Networks,” IEEE Transactions on Multimedia, vol. 17, no. 8, pp. 1309–1322, August 2015.

[40] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, and J. G. Andrews, “Video capacity and QoE enhancements over LTE,” in 2012 IEEE International Conference on Communications (ICC 2012), June 2012, pp. 7071–7076.

[41] V. Ramamurthi and O. Oyman, “Video-QoE aware radio resourceallocation for HTTP adaptive streaming,” in 2014 IEEE International Conference on Communications (ICC 2014), June 2014, pp. 1076–1081.

[42] G. Aristomenopoulos, T. Kastrinogiannis, V. Kaldanis, G. Karantonis, and S. Papavassiliou, “A Novel Framework for Dynamic Utility-Based QoE Provisioning in Wireless Networks,” in 53rd IEEE Global Telecommunications Conference (GLOBECOM 2010).

[43] G. Lee, H. Kim, Y. Cho, and S. H. Lee, “QoE-Aware Scheduling for Sigmoid Optimization in Wireless Networks,” IEEE Communications Letters , vol. 18, no. 11, pp. 1995–1998, 2014.

[44] Quantifying the Influence of Rebuffering Interruptions on the User's Quality of Experience During Mobile Video Watching [J]. IEEE Transactions on Broadcasting, 2013, 59(1):47–61.

[45] “OMNET INET Framework,” [Online]. Available: https://omnetpp.org/. [Accessed 13 7 2017].

[46] Varga, A. and Hornig, R. (2008), "An overview of the OMNeT++ simulation environment", in Proc. SIMUTools '08, Marseille, France, March 2008.

[47] A. Virdis, G. Stea, and G. Nardini, “SimuLTE - A Modular System-level Simulator for LTE/LTE-A Networks based on OMNeT++,” in 4thInternational Conference on Simulation and Modeling Methodologies,Technologies and Applications (SIMULTECH 2014).

[48] A. Virdis and G. Nardini, “SimuLTE LTE User Plane Simulation Model for INET & OMNeT++,” SimuLTE, 2015. [Online]. Available: http://simulte.com/. [Accessed 13 7 2017].

[49] P. K. Rekhi, M. Luthra, S. Malik and R. Atri, White Paper, "Trhoughput Calculation for LTE TDD and FDD Systems", December 2012.

[50] WordPress, “ITEC – Dynamic Adaptive Streaming over HTTP,” [Online]. Available: http://www-itec.uni-klu.ac.at/dash/?page_id=6. [Accessed 24 7 2017].

[51] Dongeun Suh, Insun Jang, and Sangheon Pack,”QoE-enhanced Adaptation Algorithm over DASH for Multimedia Streaming”, Proceeding of the International Conference on InformationNetworking 2014 (ICOIN2014), pp. 497 - 501, 2014.

[52] Stefano Petrangeli, Jeroen Famaey, Maxim Claeys, Steven Latré, Filip De Turck: QoE-Driven Rate Adaptation Heuristic for Fair Adaptive Video Streaming. TOMCCAP 12(2): 28:1-28:24 (2016).

[53] M. Claeys, S. Latre, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck. 2014. Design and Optimization of a (FA)Q-Learning-based HTTP Adaptive Streaming Client. Connection Science 26, 01 (2014), 27–45.

[54] UAH College of Science, "The Central Limit Theorem" , [Online]. Available: http://www.math.uah.edu/stat/sample/CLT.html. [Accessed 10 10 2017].

[55] Michael D Byrne. How many times should a stochastic model be run? An approach based on confidence intervals. In Proceedings of the 12th International conference on cognitive modeling, Ottawa, 2013.

[56] Rita Bandeira, "Analysis of Spectrum Sharing Techniques for Unlicensed Frequency Bands", IST Master Thesis, April 2017.

Appendix A

Scheduling algorithms pseudocodes

This appendix presents the pseudocode of the implemented scheduling algorithms, used to make the comparison with the proposed scheduler. The notation used in the pseudocodes can be found in sections 4.3 and 4.4, where each scheduler is described.

- **Round Robin:**

1: for each TTI **do**
2: **for each** k^{th} RB **do**
3: $j^* = 0$
4: $m_{j^*,k} = 0$
5: **for each** i^{th} user **do**
6: $m_{i,k} = t - T_i$
7: **if** $m_{i,k} > m_{j^*,k}$ **then**
8: $j^* = i$
9: **end if**
10: **end for**
11: $NbrRB_{j^*} = NbrRB_j + 1$
12: **end for**
13: end for

- **Blind Equal Throughput:**

1: for each TTI **do**
2: **for each** k^{th} RB **do**
3: $j^* = 0$
4: $m_{j^*,k} = 0$
5: **for each** i^{th} user **do**
6: $m_{i,k}^{BET} = \frac{1}{R_i(t)}$
7: **if** $m_{i,k} > m_{j^*,k}$ **then**
8: $j^* = i$
9: **end if**
10: **end for**
11: $NbrRB_{j^*} = NbrRB_j + 1$
12: **end for**

13: end for

- **Proportional Fair:**

1: for each TTI **do**
2: **for each** k^{th} RB **do**
3: $j^* = 0$
4: $m_{j^*,k} = 0$
5: **for each** i^{th} user **do**
6: $m_{i,k}^{PF} = \frac{d_k^i(t)^\alpha}{R_\iota(t)^\beta}$
7: **if** $m_{i,k} > m_{j^*,k}$ **then**
8: $j^* = i$
9: **end if**
10: **end for**
11: $NbrRB_{j^*} = NbrRB_j + 1$
12: **end for**
13: end for

- **Proportional Fair with Barrier for Frames:**

1: for each TTI **do**
2: **for each** k^{th} RB **do**
3: $j^* = 0$
4: ${m_{j^*,k}}^{PFBF} = 0$
5: **for each** i^{th} user **do**
6: **if** $\sum_{i=1}^{k} p_{rebuf,i} > 0$ then
7: $V_i = 1 + \frac{k \times p_{rebuf,i}}{\sum_{i=1}^{k} p_{rebuf,i}}$
8: **else**
9: $V_i = 1$
10: **end if**
11: ${m_{i,k}}^{PFBF} = V_i(\frac{\alpha \times d_k^i(t)}{S_{frame,i}} \times exp(\beta(f_{min} - f_i)) + \frac{d_k^i(t)}{R_\iota(t)})\}$
12: **if** ${m_{i,k}}^{PFBF} > {m_{j^*,k}}^{PFBF}$ **then**
13: $j^* = i$
14: **end if**
15: **end for**
16: $NbrRB_{j^*} = NbrRB_j + 1$
17: **end for**
18: end for