Utilizing Buffered YouTube Playtime for QoE-Oriented Scheduling in OFDMA Networks

Florian Wamser*, Dirk Staehle*, Jan Prokopec†, Andreas Maeder‡, Phuoc Tran-Gia*
*University of Würzburg, Germany
Email: {wamser,staehle,trangia}@informatik.uni-wuerzburg.de
†Brno University of Technology, Czech Republic
Email: prokopec@feec.vutbr.cz
‡NEC Laboratories Europe, Network Research Division, Heidelberg, Germany
Email: andreas.maeder@neclab.eu

***Abstract*—With the introduction of 4th generation mobile networks, applications such as high-quality video streaming to the end user becomes possible. However, the expected demand for such services outpaces the capacity increase of the networks. Since there is mostly a capacity bottleneck in the air interface between a base station and user equipment, one of the main challenges for radio resource management is therefore to enforce precise quality guarantees for users with high expectations on service quality.**

We consider, in this paper, an OFDMA access network with YouTube users, and address the challenge of improving the quality of experience (QoE) of a dedicated user by utilizing the buffered playtime of a YouTube video for scheduling. The advantage of this approach is that scheduling is done according to the *instantaneous* throughput requirement of the end user application, and not by the network by maintaining average quality-of-service (QoS) parameters. The paper describes the concept and provides a simulative evaluation of the approach in an LTE network to demonstrate the benefits.

***Index Terms*—cellular networks, application-aware networking, scheduling, quality of experience (QoE)**

I. Introduction

4th generation (4G) mobile networks based on 3GPP LTE [1] or Mobile WiMAX promise high data rates for mobile use. With some networks already operational and many more in the process of deployment, the available bandwidth enables consumers to use high-quality Internet services which until recently required a high-speed fixed access.

According to traffic and services forecasts such as in [2], mobile video traffic in 2015 will exceed 60% of all generated traffic which will increase 26-fold compared to 2010. A large part of this traffic is generated by smart phones and tablets, a trend which will also increase in the future. In the same time frame, mobile capacity is expected to grow around 10-fold with the deployment of LTE-Advanced and small cell networks.

Dirk Staehle is now at DOCOMO Communications Laboratories Europe GmbH, Munich.

This work has been funded by the German Research Foundation under grant TR 257/28-2 (FunkOFDMA), by European Community's Seventh Framework Programme (FP7/2007-2013) under grant no. 230126 and is also supported by the research proposal CZ.1.07/2.2.00/15.0139 Wireless Communication Teams.

The challenges for mobile operators are therefore twofold: on the one hand, customers have high expectations on the delivered quality of experience (QoE) of the services, which will be to a large extend based in the Internet and not under the control of the operator. On the other hand, the mobile capacity cannot be increased at the same pace as the demand grows. This development implies that for the providing high QoE to the customers, novel solutions based on traffic differentiation according to the application or service requirements are needed.

In literature, most work on QoE-aware scheduling focuses on utility-based scheduling schemes which map instantaneous quality-of-service (QoS) metrics such as throughput or delay to a corresponding QoE utility such as a mean opinion score (MOS). Using this method in [3] a differentiation between voice, streaming, and best-effort traffic is achieved, however, without using actual QoE models for the different traffic types. Other examples of this approach are [4] and [5], which formulate the QoE scheduling problem as an optimization problem for different traffic classes, the latter explicitly for 3.5G HSDPA access networks. In [6], a similar approach is proposed for web traffic by mapping the current user data rate to a web-MOS value.

The utility-based scheduling approach delivers good results for services with relatively static QoS requirements, e.g. for constant bit rate voice or video streaming, or for very elastic traffic such as background traffic. However, applications such as YouTube have a time-varying demand on bandwidth due to codec and user behavior. It would be therefore necessary to adapt the QoS mapping in the scheduler on the instantaneous requirements on the client side in order to guarantee good QoE. This is partially acknowledged in [7], where two utility curves for progressive video download are introduced, one for the streaming and one for the waiting case (i.e. if the user pauses the video).

The contribution of this paper addresses the challenge mentioned above by enabling QoE-aware scheduling of YouTube progressive video download and web browsing. For fine-granular resource allocation according to the current requirements of the application, the scheduler incorporates client-

based feedback in the scheduling decisions at the base station. An evaluation is performed with an LTE system level simulator which implements a detailed model of YouTube and TCP, as well as the LTE protocol stack and wireless channel models.

The remainder of the paper is structured as follows. Section II describes the problem and presents background information about YouTube video streaming and application-aware traffic management. We describe the simulation methodology and the simulation scenario in Section III. The evaluation is done in Section IV. In Section V we summarize our findings and conclude the paper.

II. Utilizing Application Information

The current mobile communication paradigm is to differentiate on service level for provisioning of QoS to the end user. For this purpose, different QoS classes are standardized which are then mapped to different applications according to their approximate requirements.

In 3GPP LTE, bearers are used to forward the data between user equipment and the Internet gateway. Each bearer is set up with a bearer QoS profile that specifies guaranteed parameters on network level. With the classification into different QoS classes, groups of network flows with similar needs are prioritized equally and packet scheduling of different groups is done according to the QoS definitions. However, at present, almost every mobile network exclusively establishes only default bearers without guaranteed transmission resources, since it is not clear how to map IP traffic to QoS classes. Furthermore, due to heterogeneous applications or different end user devices the quality requirements of applications are varying depending on multiple diverse parameters such as screen size of the device, the used coding technique, or the previous usage history of the user.

To quantify the delivered application quality at the end user, the ITU defines QoE. QoE is the overall acceptability of an application or service as perceived by the end-user [8]. Compared to QoS, in addition, other parameters than only network parameters are considered. This includes any subjective and objective parameters such as video content, encoding parameters, usage scenario, network performance, and application state.

For incorporating QoE in the resource management, a common way is currently to determine in a first step the most influential parameters through separate studies [9], [10]. The parameters may depend on the application or even on user preferences. In the second step, if a QoS network parameter is found that significantly influences the QoE, a QoS-based forwarding like in traditional scheduling frameworks is done according to this parameter. Therefore, a predefined QoS-QoE mapping function is commonly used.

However, a QoS-based scheduling alone is often not sufficient to provide an acceptable QoE. This is the case, especially, for applications with time-dynamic bandwidth requirements. Due to encoding, download patterns, or user behavior, for example, an application has no fixed demand on bandwidth. Instead, bandwidth is required depending on application state.

YouTube uses progressive HTTP video streaming which means that the video data is buffered at the client side. The buffering is done according to a two-phased download pattern [11], [12]. At the beginning, the buffer is filled with a certain amount of data (initial block) and afterwards, the buffer is refilled by a periodic rate. The transmission of the initial block is done best-effort-like. The periodic refill is controlled by the YouTube server with a rate that depends on the total video rate. Therefore, YouTube requires a time-dynamic scheduling according to the buffering phase to ensure a smooth running YouTube video and thus, a high YouTube QoE.

Hence, a simple mapping of a QoS parameter such as throughput to YouTube is difficult since it depends on the application state. We propose in the following a specific scheduling scheme which is based on application parameters instead of QoS network parameters.

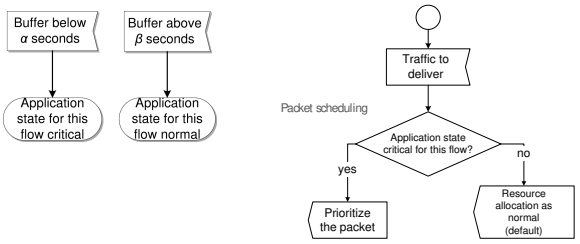
A. YouTube Video Streaming

YouTube uses progressive HTTP video streaming. The default compression format of the video content is H.264/MPEG-4 Advanced Video Coding (AVC). The encoded data is transmitted over the HTTP protocol to the clients and stored in a buffer in the application. The YouTube client pre-buffers a certain amount of data until the playback starts. Afterwards, a periodic refilling of the buffer is done during playback. The two phased download is controlled by the YouTube content servers and adapted for every video according to the total video rate [11], [12]. A buffering period without playback, which is in the following called *stalling*, occurs if and only if the buffer is empty. In contrast, bad network conditions or large delays can be sustained if sufficient video playtime is buffered.

B. Smart Scheduling: Application-aware Traffic Management

Several studies quantify the impact of user and network parameters on YouTube QoE [13], [14]. The most influencing parameters are the duration of the buffering period and the rate of buffering events. Consequently, in this paper the buffered playtime of the YouTube video player is utilized to optimize the user perceived quality. Instead of using a network QoS parameter for scheduling, client-based feedback is used to directly forward the buffer level of each YouTube video to the scheduler at the base station. This directly addresses the application and implicitly takes into account application-specific mechanisms such as buffering strategy, video resolution, or even user interaction.

The scheduling is done as follows. As illustrated in Fig. 1(a), as soon as the buffered playtime of one YouTube client falls below a threshold of α seconds, a signaling event is generated by the client. Additionally, if the buffered playtime exceeds a second threshold of β seconds, again a signaling event is generated. We assume that a logical feedback channel exists between the YouTube client application and the scheduling entity in the base station. A user who watches a YouTube video triggers a feedback event if the playback buffer is exceeding or



(a) Signaling events (b) Scheduling at the base station

Fig. 1. Flow diagram of the scheduler.

falling below one of these thresholds. In the scheduler, a flow is tagged as being in a critical state if feedback is received indicating that the buffered playtime is below the threshold α. A flow is tagged as normal if feedback is received indicating that the threshold β is exceeded.

In Fig. 1(b) the scheduling at the base station is depicted. If the scheduler receives a packet, it checks whether the packet belongs to a flow in a critical state or not. If the packet does not belong to a YouTube video, the state of the flow is considered as normal. If the state of the flow is critical, then the client is prioritized by the scheduler. The scheduler prefers this packet over other users and allocates it to the transmission frame. In all other cases, the packet is passed to the resource allocator as in the normal case which means that the scheduling is done according to a certain fairness metric, which may consider channel quality or service-level QoS parameters.

The proposed scheduling does not follow a proactive approach to optimize the QoE. As a consequence, it should run additionally to some traditional scheduling algorithms that do not take into account application layer parameters but consider channel quality or service-level QoS parameters. Only if a QoE degradation is imminent, in spite of the normal scheduling, this approach will prioritize a flow in order to avoid QoE degradation.

The advantage of this approach is that the scheduling is done according to the state of the end user application to provide an acceptable quality, and not by the network by maintaining certain QoS levels for the application

III. Simulation Methodology and Scenario

For evaluation of the proposed scheduling scheme, one YouTube video was chosen and investigated with two schedulers, namely the round robin scheduler and the buffered playtime scheduler which takes the buffered playtime of a YouTube video into account. The round robin scheduler was selected because of the low implementation effort rather than a proportional fair scheduler or a max-rate based algorithm. Furthermore, a consideration of throughput and fairness in the scheduler, as it is done for instance in the proportional fair algorithm, is not relevant for the statement of the paper.

We simulate 20 s of the YouTube video with detailed application and physical layer models. Additionally, TCP with flow and error control is simulated. The video was randomly selected[1]. It is a popular movie trailer with very low data rate at the beginning and medium alternating rate thereafter. The whole duration of the video is 92.7 s and the average video rate is 463 kbit/s. In Fig. 2 the video encoding is shown for the relevant simulation time. The low bit rate at the beginning of the video is due to a static video title in the first 5 s.

One single mobile cell is simulated with a discrete-time event-based simulation based on the LTE Downlink Link Level Simulator of the University of Vienna[2]. The link level simulator is based on LTE release 8 [15] with PHY and MAC and functions as specified in [16], [17]. The simulator is free for academic usage and can be used for research in the area of LTE signal processing [18] or as a foundation for system level simulations [19]. The simulator implements the complete signal processing chain for the traffic channel. Signaling and control channels are simulated as error-free. For the abstraction of the physical layer to event based simulation of upper layers, we use results evaluated as BLER (Block Error Rate) and throughput per user as available bandwidth for the upper layer. BLER is transformed to a packet error rate. Simulation of upper layers utilize pre-calculated values of packet error rates and throughput for the requested number of users and the available resource blocks. Due to the fact that the user throughput depends on the allocation in the resource grid of LTE, we do not use the average throughput per user calculated from cell throughput, but simulate a detailed resource allocation and obtain an exact user throughput.

The simulation scenario is as follows. Up to 11 users in a single cell are simulated. They randomly move around within the cell with a speed of 1 m/s. The channel model includes path loss, shadow fading, and multipath fading. Shadow fading is generated with zero-mean according to the Gaussian distribution with σ standard deviation of 2 dB. The shadow fading decorrelation distance is assumed to be 50 m. Multipath fading is simulated according to the ITU Pedestrian B profile. Each user may watch a YouTube video, download a file, or browse the Internet.

At application layer, the YouTube Flash Player and a YouTube download server is simulated for YouTube users. It processes HTTP data to display the YouTube video. In particular, it calculates the current buffered video playtime in seconds. The player may stall if the playtime buffer is empty. The play-out delay after stalling is set to 5 s which is the current value of the YouTube video player. The YouTube download server behaviour follows [11] with refinements according to own measurements. The download speed is controlled by the server in two phases. The first phase is the initial pre-buffering. The second phase is a periodic buffer refill, see Fig. 5(c). The periodic phase depends on two parameters, the block size which is set to $b = 64\,\mathrm{kB}$, and the inter-block arrival time which depends on the average total data rate of the YouTube

[1] http://www.youtube.com/watch?v=Q1D5goGz0SY, last accessed 01/12

[2] http://www.nt.tuwien.ac.at/about-us/staff/josep-colom-ikuno/lte-simulators/

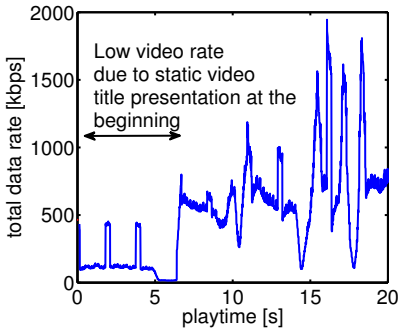
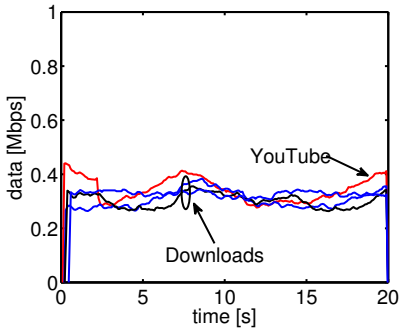
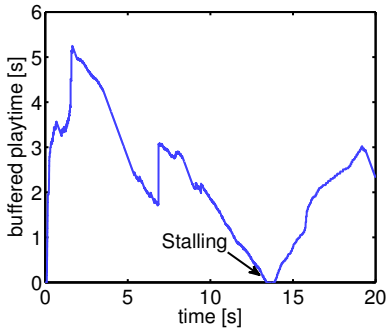


Fig. 2. YouTube video encoding



(a) Throughput



(b) Buffered playtime of YouTube video

Fig. 3. Round robin scheduler with three download users and one YouTube user

TABLE I
WEB SESSION PARAMETERS

volume main object	log-normal: $\ln\mathcal{N}$(10 kbytes, 25 kbytes) $\in$ [100 bytes, 2 Mbytes]
number of embedded objects	truncated Pareto(scale, shape, max): Pr(1.1,2,55)
volume embedded object	log-normal: $\ln\mathcal{N}$(8 kbytes, 126 kbytes) $\in$ [50 bytes, 2 Mbytes]
reading time	neg. exponential: Exp(3s)

video, namely

$$f_{ibr} = \min(\alpha_{th}, p_1 x^{p_2} + p_3),$$

with an upper threshold at $\alpha_{th} = 2096$, x as the total data rate, and $p_1 = 400000$, $p_2 = -1$, $p_3 = -5.71$. The initial buffering period depends on the block size and the inter-block arrival time [11]:

$$f_{ibs} = 32 \cdot b \cdot \frac{1000}{f_{ibr}}.$$

For web browsing users, a simple web server is simulated that is answering HTTP requests. TCP connection handling is done according to the Apache web server default configuration. For HTTP/1.1, Apache 2.2 defines a keep-alive timeout of 5 s. No download speed limit or connection limit is set. For a web session, a web page is defined as a main object and several embedded objects. Embedded objects are for example images, JavaScript code, or CSS style sheets. The data volume of the main object, the size of an embedded object, and the number of embedded web objects per web page are generated according to random distributions, see Table I. The web session client generates a reading time after web page transfer, see Table I which is the time between two successive web page downloads.

TCP New Reno with flow control, error detection, congestion control is simulated for each user to include flow control mechanisms that will influence the packets available for scheduling.

On the packet level, round robin scheduling and buffered playtime scheduling is implemented as described in Section II.

IV. Numerical Results

A. HTTP Downloads and YouTube

This section presents the simulation results of the buffered playtime scheduler with a YouTube video and HTTP downloads. The results are compared to the round robin scheduling algorithm.

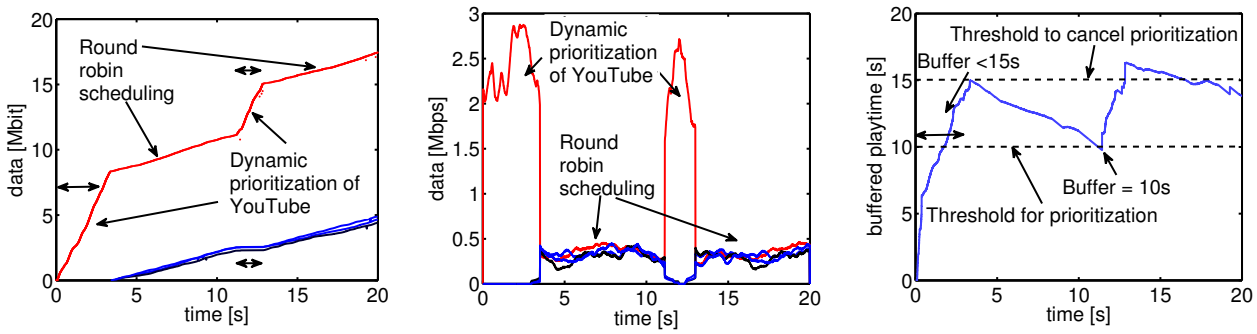
An 3GPP LTE system is simulated with single-input-single-output (SISO) antenna configuration. The simulated bandwidth is set to 1.4 MHz to reduce the simulation time. Four users are active in the cell. One YouTube user and three download users are simulated. The YouTube user starts at time instance zero, the downloads start randomly with a delay. The delay is determined according to an exponential distribution with a mean of two seconds. The main performance metric here is the buffered playtime of the YouTube video. The best effort downloads are responsible for heavy load in the cell which affects the buffered playtime of the YouTube video.

Fig. 3(a) depicts the throughput of the four users. The YouTube user is indicated by the red curve. Download users are shown in blue color. On the x-axis the transmitted data in Mbps is shown. The y-axis shows the simulation time in seconds. The throughput is almost equal for all users and will only be influenced due to the different transmission channel conditions of the users since they are moving.

Fig. 3(b) shows the resulting buffered playtime of the YouTube video over the simulation time. The sharp increase of the buffer at 7 s is due to the video encoding since there is a small period with very low encoding rate from 5 to 6 s of video playtime, see Fig. 3(a). At 13 s the buffer is empty. The video begins a buffering period, and the user experiences video stalling.

In Fig. 4 the same scenario is depicted as in the previous one but with the buffered playtime scheduler which dynamically prefers the YouTube video in the case of low YouTube buffer.

The first sub figure of Fig. 4 presents the cumulative downloaded data of the downloads and the YouTube video player. Together with Fig. 4(b) the difference between the buffered playtime scheduler to the round robin scheduler becomes visible. The YouTube flow is prioritized at the beginning and



(a) Cumulative data (b) Throughput buffered playtime scheduler (c) Buffered playtime of YouTube video

Fig. 4. Buffered playtime scheduler with three download users and one YouTube user

for 1.6 s at about 11 s due to the scheduling strategy. Almost no data is transferred at these time periods to the download users since YouTube is using nearly the whole bandwidth. Outside these time periods, the data is equally scheduled among the users as in round robin strategy. Fig. 4(c) shows the corresponding buffered playtime. The buffer level is always greater than zero, and no stalling occurs. The thresholds for the prioritization are visible: if the buffer level is higher than 15 s, round robin scheduling is used. Since, in this scenario with four users, the throughput during the round robin phase is not sufficient, the buffer level decreases afterwards. If the buffer level falls below 10 s playback time, YouTube is prioritized again.

With such a basic algorithm, the YouTube QoE can be improved at some expense of the download time for best-effort users. For quantifying the YouTube QoE, concrete mapping functions, depending on the length of stalling and the ratio of stalling, are proposed in literature. According to [13], one stalling already results in a QoE degradation from MOS 5 to 3.2 if the stalling length is 3 s until the flash player will restart the video playback. Another buffering period would further decrease the MOS value from 3.2 to 2.5. Contrary to YouTube, the QoE of file downloads is more robust. Especially for long downloads a small delay can be tolerated. In our case the download time of the downloads increases by 3.8 to 5.3 s per download depending on the channel conditions of the YouTube user for the two prioritization periods.

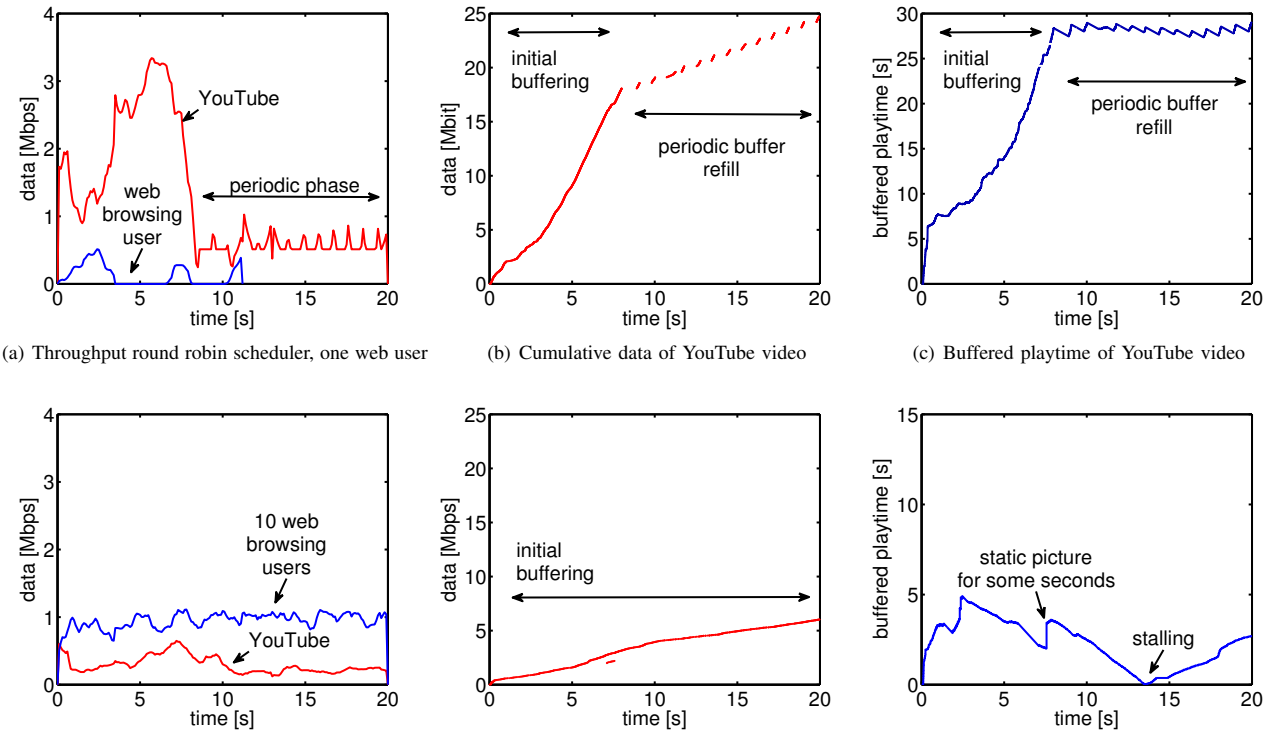
B. Web Browsing Sessions and YouTube

In this section, web browsing users are simulated together with one YouTube user. A web session of a web user is defined as described in the simulation section, see Table I. The number of web users is increased until YouTube QoE is affected. On the one hand, the results show a similar benefit for YouTube users if the buffered playtime scheduler is used. On the other hand, due to the knowledge of the exact buffer level of YouTube, the impact on the web users can be kept to a minimum and adapted with the buffer thresholds used in the buffered playtime scheduler. Two examples for different buffer thresholds are given.

Fig. 5(a)-(c) shows results for one web user and one YouTube user. Fig. 5(a) shows in red color the throughput of the user who is watching the YouTube video and in blue color the web user throughput. The web user is watching three web pages at 7 s, and 10 s. The web traffic is influencing the YouTube throughput: the YouTube throughput is decreasing while the web traffic is increasing during the reading time of the web user. Fig. 5(b) shows the corresponding accumulated data during the simulation time of the YouTube video only. The two download phases can be seen. At the beginning, YouTube is doing an initial buffering. Afterwards, there is a periodic buffer refill which is also reflected by the throughput in Fig. 5(a). With one web user the YouTube video time buffer remains stable over the whole simulation time which can be seen in Fig. 5(c). After the initial buffering, here, the buffer is kept at about 27 s.

Now, Fig. 5(d) shows the situation with 10 web users and round robin scheduler. The blue curve shows the throughput of all web users. The red curve shows the throughput of the YouTube user. The YouTube throughput decreases to about 300 kbit/s - 500 kbit/s due to the round robin scheduling which treats all TCP flows of the users equally. Fig. 5(e) shows that the YouTube player is not even able to complete the initial best effort buffering phase. With 10 users, the buffered playtime in Fig. 5(f) remains below 5 s and stalls again at 13 s.

While the instant buffer level is important for scheduling, there are also some other important points if YouTube is scheduled. One is the maximum reachable buffer level. In Fig. 6(a) the cumulative distribution function of the buffer level of the YouTube video for one, 7 and 10 web users in parallel is plotted. It shows the disadvantage of static buffer refilling of YouTube without client feedback. The buffer is refilled with a constant rate independent of the throughput in the initial phase. Thus, with 7 or 10 web users in parallel, the lower throughput during initial buffering influences the maximum buffer level significantly. If, during the initial phase the throughput is not sufficient, during the periodic phase the buffer can only increase marginally since the periodic refill depends on the average video rate of the video, see Section III. This leads



(a) Throughput round robin scheduler, one web user (b) Cumulative data of YouTube video (c) Buffered playtime of YouTube video

(d) Throughput round robin scheduler, 10 web users (e) Cumulative data of YouTube video (f) Buffered playtime of YouTube video

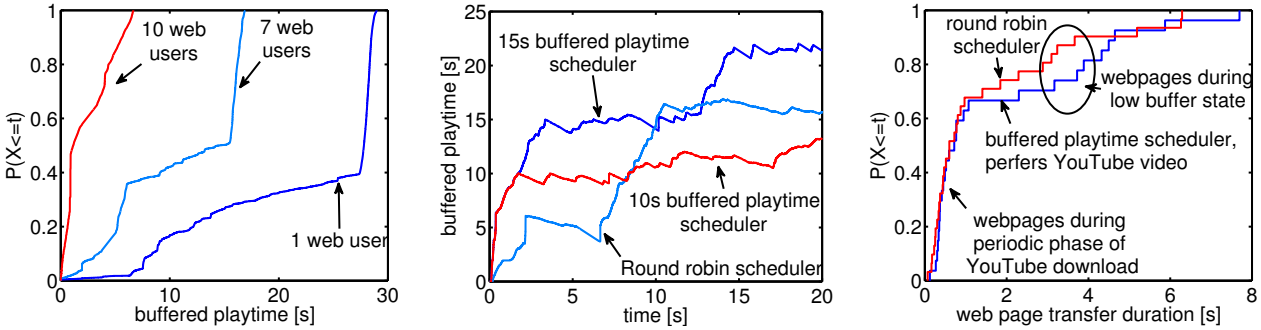
Fig. 5. Web browsing with one YouTube video

to the fact that the transmission time of the initial download block of the YouTube video stream should be kept short, which in turn, means that a high throughput in initial phase should be guaranteed for a YouTube video. Consequently, also the probability of having the maximum buffer level decreases with the number of web users in parallel. This can be again explained by the fact that for YouTube, the probability to get a high throughput during initial buffering decrease due to the load of the web users. With a high throughput, YouTube is able to fill up the buffer to a higher amount of buffered playtime and thus at the periodic refill phase higher fluctuations can be tolerated in a) the network throughput and, even more important, b) within the video data due to the adaptive video encoding.

We now show the buffer progress with a buffered playtime scheduler which signals the current buffer level to the scheduler. Fig. 6(b) contains three curves showing the buffer level over time for different scheduler settings. The curves are evaluated for 7 web users in parallel to the YouTube video. The round robin scheduler is included for comparison. For the top blue curve the scheduler is set to the same parameters as in the download scenario with buffered playtime scheduler. If the video time buffer is below 10 s the YouTube flow is strictly prioritized. At a threshold of 15 s round robin strategy is used until it falls below the critical 10 s. In this scenario the buffer is not significantly decreasing after achieving the 15 s of buffered playtime. A smooth video playback is possible without stalling since the initial prioritization is enough for initially filling the buffer. The initial buffer level is able to compensate the variable encoding of the video for the whole simulation time. Note, this is video specific and depends on the encoding of the video. If the setting of the buffered playtime scheduler is changed to $\alpha = 10\,s$ round robin threshold, and $\beta = 9\,s$ critical threshold, Fig. 6 shows that due to the variable encoding the critical threshold is reached very often at the beginning. Consequently, a prioritization of the YouTube flow is done until the 10 s buffered playtime is reached again. The second scheduler setting has the advantage that the transmissions of web users are delayed for a shorter time period. However, the rate of delaying the web users is higher in comparison to a higher difference between round robin threshold and critical buffer threshold. The impact on the web users is discussed in the next subsection.

C. Impact on Web Browsing Users of the Buffered Playtime Scheduler

Fig. 6(c) shows the web page transfer duration for the round robin and the buffered playtime scheduler. Again, the duration is plotted as CDF. The round robin scheduler is plotted in red, the buffered playtime scheduler is colored in blue. The figure shows that the curves only differ for the download durations longer than 1.7 s. There, the probability of having



(a) CDF of buffered playtime for the YouTube video with a different number of web users

(b) Buffered playtime with different schedulers for 7 web users

(c) Data transfer duration with round robin and buffered playtime scheduler

Fig. 6. Web browsing with one YouTube video

larger download times is more likely when using the buffered playtime scheduler. This is obvious since the buffered playtime scheduler is preferring YouTube flows over the other flows if the buffer state of the YouTube video is low. Thus, the web page transfer duration is longer in comparison with the round robin scheduler. However, at higher buffer levels of the YouTube player the buffered playtime scheduler is acting in the same way as the round robin scheduler. The following trade-off can be seen. Preventing YouTube video stalling is achieved through delaying the web pages in favour of YouTube flows.

V. Conclusion

In this paper, QoE-oriented scheduling for YouTube is described and evaluated. Applications as YouTube have time-varying demand on bandwidth due to encoding, download patterns, and user behaviour. It is therefore necessary to adapt the QoS mapping in the scheduler on the instantaneous requirements on the client side in order to guarantee good QoE at the end user. Consequently, a scheduling algorithm is proposed that dynamically prioritizes YouTube users against other users if a QoE degradation is imminent. The prioritization is done in a proactive way according to the buffered playtime of the YouTube video player.

The results are evaluated with an 3GPP LTE system level simulator which implements a detailed model of YouTube and TCP as well as wireless channel models. First, a YouTube video together with best-effort downloads is simulated. A buffering period of YouTube can be avoided at the expense of download time. Especially for long downloads, the overall QoE is improved since an increase of the download time can be tolerated for them and does not negatively influence the QoE. Second, the scheduling algorithm is evaluated with the most important application today in mobile communication networks. HTTP web browsing is simulated with a YouTube video. The impact on web browsing users of the buffered playtime scheduler is presented. Web browsing users are only affected if the YouTube player runs out of buffered video data. There are two thresholds defined to control the impact on the web users.

Future work will include a precise evaluation of the signaling overhead and the consideration of other applications for the concept of QoE-oriented and application aware scheduling.

References

[1] 3GPP, *TS 36.300 V10.5.0; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description*, 3GPP Std., Sep. 2011.

[2] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 20102015," Feb. 2011.

[3] G. Song and Y. G. Li, "Utility-Based Resource Allocation and Scheduling in OFDM-Based Wireless Broadband Networks," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 127–143, Dec. 2005.

[4] S. Khan, S. Duhovnikov, E. Steinbach, and W. Kellerer, "MOS-Based Multiuser Multiapplication Cross-Layer Optimization for Mobile Multimedia Communication," *Advances in Multimedia*, vol. 2007, 2007.

[5] S. Thakolsri, S. Khan, E. Steinbach, and W. Kellerer, "QoE-Driven Cross-Layer Optimization for High Speed Downlink Packet Access," *Journal of Communications*, vol. 4, no. 9, pp. 669–680, Oct. 2009.

[6] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, P. Mogensen, and J. M. Lopez-Soler, "QoE oriented cross-layer design of a resource allocation algorithm in beyond 3G systems," *Computer Communications*, vol. 33, no. 5, pp. 571 – 582, 2010.

[7] E. Meshkova, J. Riihijärvi, A. Achtzehn, and P. Mähönen, "On Utility-Based Network Management," in *Proc. of IEEE International Workshop on Management of Emerging Networks and Services (MENS)*, Miami, Florida, USA, Dec. 2010.

[8] *ITU-T Recommendation P.10/G.100 Amendment 2 Definition of Quality of Experience (QoE)*, International Telecommunication Union (ITU) Std., Jul. 2008.

[9] M. Fiedler, T. Hoßfeld, and P. Tran-Gia, "A Generic Quantitative Relationship between Quality of Experience and Quality of Service," *IEEE Network Special Issue on Improving QoE for Network Services*, Jun. 2010.

[10] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *Communications Magazine, IEEE*, vol. 46, no. 2, pp. 78–84, 2008.

[11] Alcock, S. and Nelson, R., "Application flow control in youtube video streams," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 2, pp. 24–30, 2011.

[12] A. Rao, A. Legout, Y. Lim, D. Towsley, C. Barakat, and W. Dabbous, "Network characteristics of video streaming traffic," p. 25, 2011.

[13] T. Hoßfeld, R. Schatz, M. Seufert, M. Hirth, T. Zinner, and P. Tran-Gia, "Quantification of YouTube QoE via Crowdsourcing," in *IEEE International Workshop on Multimedia Quality of Experience - Modeling, Evaluation, and Directions (MQoE 2011)*, Dana Point, CA, USA, Dec. 2011.

[14] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang, "Understanding the impact of video quality on user engagement," in *ACM SIGCOMM 2011*. ACM, 2011, pp. 362–373.

[15] 3GPP Technical Specification Group RAN, "E-UTRA; LTE physical layer – general description," 3GPP, Tech. Rep. TS 36.201 Version 8.3.0, March 2009.

[16] ——, "E-UTRA; physical channels and modulation," 3GPP, Tech. Rep. TS 36.211 Version 8.7.0, May 2009.

[17] ——, "E-UTRA; multiplexing and channel coding," 3GPP, Tech. Rep. TS 36.212, March 2009.

[18] M. Simko, S. Pendl, S. Schwarz, Q. Wang, J. C. Ikuno, and M. Rupp, "Optimal pilot symbol power allocation in LTE," in *Proc. 74th IEEE Vehicular Technology Conference (VTC2011-Fall)*, San Francisco, USA, September 2011.

[19] J. C. Ikuno, M. Wrulich, and M. Rupp, "System level simulation of LTE networks," in *Proc. 2010 IEEE 71st Vehicular Technology Conference: VTC2010-Spring*, Taipei, May 2010.