On-line estimation of the QoE of progressive download services in multimedia access networks

S. Latré, N. Staelens, P. Simoens, B. De Vleeschauwer,
W. Van de Meerssche, F. De Turck, B. Dhoedt, P. Demeester
IBBT - IBCN, Department of Information Technology, Ghent University, Belgium
Gaston Crommenlaan 8/201, 9050 Gent, Belgium
Tel: +32 9 33 14 981, Fax: +32 9 33 14 899
E-mail: Steven.Latre@intec.ugent.be

S. Van Den Berghe, R. Huysegems, N. Verzijp
Alcatel-Lucent Bell Labs
Copernicuslaan 50, B-2018 Antwerpen, Belgium

Abstract – Current broadband access networks are evolving towards an infrastructure for multimedia services such as BroadcastTV, Voice over IP and Video on Demand. As these multimedia services have high quality demands, operators want to know how the end user perceives the quality of each service, defined as the Quality of Experience (QoE). We present a novel algorithm that predicts the QoE of progressive download services in real-time. This QoE is defined in terms of play-out buffer starvations. This algorithm runs on an intermediary node in the end-to-end path and can be deployed without needing any additional feedback from the client besides the normal TCP traffic. This is important, as operators often do not have control over the home network. The algorithm allows for a better estimation and optimization of the quality of progressive download services. The performance of this algorithm is tested using a novel validation framework.

Keywords: QoE, progressive download, access network

1 Introduction

In today's broadband access networks, operators are deploying a myriad of new multimedia services such as Video on Demand (VoD), BroadcastTV and on-line gaming. As these multimedia services have stringent demands in terms of packet loss, delay and jitter, their deployment poses new challenges for service operators. Operators are therefore continuously striving to optimize the Quality of Experience (QoE), defined as the quality of the service as perceived by the end user. The exact interpretation of this QoE depends on the service itself. The QoE of on-line gaming applications can be translated to the fluentness of the gameplay, while the QoE of BroadcastTV services relates to the visual quality of the image.

In this paper, we focus on the QoE of VoD services. VoD services often use progressive downloads to deliver their content. Also in the Internet, sites such as YouTube and Metacafe and on-line TV applications such as Babelgum use progressive downloads as a transportation technique. Progressive download can be seen as a "play while download" technique: a video file is downloaded from a server and stored into a play-out buffer at the client. When enough data has been received, the video client starts playing the video. The perceived quality of progressive download services differs from the perceived quality of UDP based multimedia services such as BroadcastTV. If a packet goes lost for a UDP based service, the video information belonging to this information cannot be reconstructed and this will result in visual errors such as blockiness or frame freezes. The QoE of UDP based multimedia services can therefore be defined as the visual quality of the image itself. This is not the case for progressive download based services. The underlying transport layer of progressive download based services is the TCP protocol. As TCP is a reliable transport layer protocol, packet loss does not result in loss of information but in a retransmission of the lost packet. Drops in the QoE of progressive download based services only occur if the retransmitted packet arrives too late. In this case, a buffer starvation occurs and the video client needs to refill the play-out buffer. Therefore, the QoE of progressive downloads depends on the amount of play-out buffer starvations and the initial startup delay.

Since play-out buffer starvations occur at the video player, the QoE of progressive downloads is ideally measured at the client side in the home network. However, as the home network is under control of the end user, operators often only have access to the access network. If no feedback from the video player to the access network is available, the operator has no way of determining the QoE of progressive download based services.

In this paper, we present a novel algorithm to estimate the QoE of progressive download based services intermediary

in the end-to-end path (e.g. the access node in the access network) without the need of any feedback from the home network. The algorithms supports progressive downloads that are being transmitted in an MP4 container [1]. We use the MP4 container, as it is the most commonly used multimedia container in today's broadband access networks. Based on the TCP data and acknowledgement packets and by inspecting video specific information from the progressive downloads, the algorithm tries to determine the initial startup delay, when a play-out buffer starvation occurs and how long it lasts. Furthermore, we present a validation framework where the QoE of progressive download based services can be measured. The validation framework is video codec independent.

The remainder of this paper is structured as follows. In section 2, relevant work concerning monitoring and optimizing the QoE of video services is discussed. Section 3 presents the prediction algorithm while section 4 describes the validation framework. Finally, in section 5, we evaluate the performance of the QoE prediction algorithm.

2 Related work

Several solutions have been proposed to assess the QoE of both video [2] [3] and audio services [4]. However, these solutions try to compare the original video or audio signal with a distorted one, caused by packets that go lost in an unreliable delivery protocol such as UDP. Also, these solutions use information obtained at the streamer and player side. Our algorithm focuses on the QoE of progressive downloads, where other techniques are necessary than the ones proposed earlier. Furthermore, our algorithm performs a prediction of the QoE without needing any information at the player side, besides the normal TCP traffic.

High amounts of packet loss and jitter can degrade the QoE of video sequences. Visual impairments caused by packet loss are perceived differently from impairments caused by jitter leading to play-out buffer starvation. In the former, blocking artifacts will be visible whereas in the latter frame freezes occur and video play-out becomes jerky. Nevertheless, results in [5] show that jitter degrades the QoE as much as packet loss does. [6] presents a summary of the impact of delay and jitter on perceptual quality. Recently, a video quality metric has been proposed in [7] which detects and evaluates the impact of frame freezing and frame dropping on the QoE. The proposed metric shows significant correlation with scores of subjective tests, but additional research is needed to study the performance of the proposed metric in a progressive download scenario, where no frames are dropped when the play-out buffer re-buffers.

As operators often only have access to the access network, solutions have been proposed to monitor client side information intermediary in the network. In [8], DSL Forum presented the CPE WAN Management Protocol that enables devices in the home network to be monitored. However, as these devices must support the CPE WAN Management protocol, this monitoring technique fails if home network devices do not implement this protocol. In [9] and [10] monitoring techniques for measuring QoS parameters in the access network are proposed that gather monitor information by investigating the traffic flows itself. However, these solutions focus on QoS parameters such as packet loss and jitter and do not estimate the QoE of the monitored services. In this paper, we combine the QoS parameters measured in [10] with video specific information to assess the QoE.

In recent years, different frameworks to monitor the QoE of video services have been proposed. In [11], Evalvid is introduced which uses a trace file based mechanism to measure QoS parameters such as packet loss, delay and jitter and QoE parameters such as video quality metrics. Evalvid supports H.264, MPEG-4 and H.263 and can be used over both real and simulated networks. In [12] we proposed a framework to enable the scalable simulation of real video sequences in a NS-2 [13] network without depending on the video codec itself. However, this codec independency comes with a cost of losing video information to assess the video quality. [14] uses neural networks to assess the QoE in video delivery networks. However, similar to the solutions presented in [11] [12] it uses client side information. All frameworks focus on the QoE in terms of visual errors, and can therefore not be used for progressive download based services.

3 Play-out buffer starvation prediction

In this section we describe the details of our algorithm that predicts the initial startup delay of a progressive download, the number of play-out buffer starvations and the length of each starvation. The algorithm monitors information from both the network layer and application layer. Network layer information is collected by inspecting the information in the TCP data and acknowledgement packets, while the metadata information of an MP4 container is used to collect application layer information. The MP4 container is a multimedia container format used to store audio and video streams. A multimedia container format packs different streams (e.g. an audio and video stream) into one common file and provides synchronization information, allowing for a synchronized playback of the packed content. The MP4 container supports many widely supported video codecs including H.264, MPEG-4 Part 2, MPEG-2 and MPEG-1. Therefore, the play-out buffer starvation prediction algorithm works with the most common video codecs as long as they are transmitted using the MP4 container.

Figure 1 gives an overview of the scope of the algorithm. Our algorithm runs on an intermediary node in the network and predicts when packets arrive at the client. The network

layer information is partially extracted using ANTMA [10], an earlier proposed algorithm that monitors QoS parameters such as packet loss and delay from TCP traffic. To guarantee its accuracy, ANTMA makes two important assumptions on the node that monitors the TCP traffic. First, the node must be able to see all data and acknowledgement packets traveling in both directions. Second, no re-ordering of packets may occur between the monitoring node and the TCP receiver. Both of these assumptions are valid on the access node in an access network. As an access node is the sole entity that provides a home network access to the Internet, the first assumption holds. Also the second assumption is valid when ANTMA is deployed on the access node. Packet re-ordering is caused when packets take different routes to their destination or by internal buffers in high-performance routers. None of these situations normally occur in home networks as typical home network devices such as routers, switches, etc. do not cause packet re-ordering.

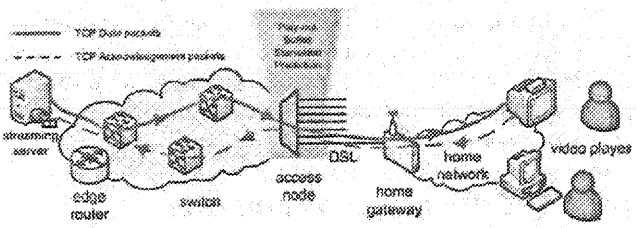

Figure 1 Scope of the play-out buffer starvation prediction algorithm. The access node monitors the video player behavior based on network layer and application layer information.

Figure 2 illustrates which information is monitored and how the algorithm processes it. At the network level, the algorithm monitors each packet passing through, reads its packet size and estimates its delay based on the round trip time. If a packet is lost, the monitor information extracted from this packet is discarded. By monitoring this information, the algorithm can estimate the arrival time of each packet at the video player. At the video layer, the algorithm monitors the decoding time and the frame size. As this information is available in the metadata of the MP4 container, the video does not need to be decoded but the algorithm can read the information by inspecting the frame header. Therefore, the cost of extracting this information is low.

Using this information, we can estimate when each frame arrives at the video player. By comparing the frame arrival time with the frame decoding time we can predict when a starvation occurs in the play-out buffer for a given play-out buffer size. This is done by mimicking the play-out buffers behavior. At startup time, the play-out buffer is filled with frames until the decoding time of the frames is greater or equal than the initial buffer size (in Figure 2 denoted by *BUFFER_SIZE*). The frame arrival time of the first frame with a decoding time greater or equal than *BUFFER_SIZE* is defined as the initial startup delay. Possible play-out buffer starvations are detected by checking for each frame if equation (1) holds. Here, *FAT(f)* is the frame arrival of frame f, *DTime(f)* is the decoding time of frame f and *FillTime* is the overall time needed for re-buffering the play-out buffer. Initially, this is set to the initial startup delay but whenever a play-out buffer starvation occurs, the *FillTime* increases.

$$FAT(f) - FillTime \leq DTime(f) \quad (1)$$

As long as equation (1) holds, the frame can immediately be played. When the *FAT(f)* becomes so high that equation (1) does not hold, a play-out-buffer starvation occurs. The length of this starvation is calculated the same way as the initial startup delay. Video playback is supposed to be halted and the play-out buffer is refilled. We define the length of a play-out buffer starvation as the difference in frame arrival time between two frames f_1 and f_2 where f_1 is the frame where video playback halted and f_2 is the first frame in the sequence for which equation *(2)* holds.

$$DTime(f_2) - DTime(f_1) \geq BUFFER\_SIZE \quad (2)$$

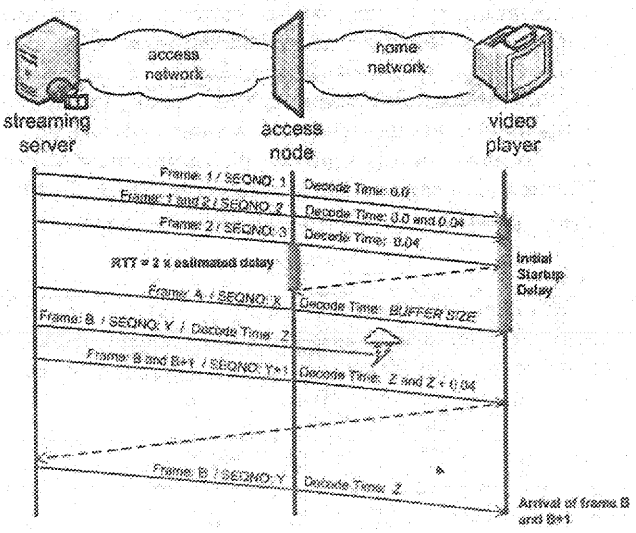

Figure 2 Overview of the monitored information at the access node. The algorithm measures the decoding time and length of each video frame and estimate the arrival time of each data packet.

4 Validation framework

In order to test the performance of the play-out buffer starvation prediction algorithm we constructed a validation framework that enables measuring the play-out buffer behavior in case of progressive download or HTTP streaming. By using the proposed architecture, it is possible to monitor the video play-out buffer and measure when

buffer starvations occur and how long it takes each time to refill the buffer. Figure 3 provides an overview of the proposed architecture. As illustrated, our framework uses several trace files to measure both video and network related parameters. Each trace file extracts network or video related information at the streamer and video player side.

Before the video file is being streamed (or downloaded) through the network, a video trace file is generated that contains detailed information about every frame in the encoded sequence such as frame size, decoding time and presentation time. Using this information, the framework knows exactly when each frame needs to be played. After the video trace file has been generated, the video sequence can be sent through the network. While the video is being streamed, a sender trace file and a receiver trace file are generated which contain the timestamp, id and payload size of every packet being respectively sent or received through the network. By inspecting the generated receiver trace file and the video trace file, the *etmp4tcp* tool determines how many buffer starvations occurred during video playback and how long it took each time to refill the play-out buffer. This calculation is based on how common video clients handle video playback. When the next frame necessary for decoding is not available in the play-out buffer, a starvation occurs and the play-out buffer is refilled. Information about each starvation is saved in a buffering behavior file. The first line in the resulting file indicates the initial start-up delay. Furthermore, by inspecting the sender and receiver trace files in more detail, some additional information can be gathered such as the number of retransmissions, the number of lost packets, the number of out-of-order packets, etc.

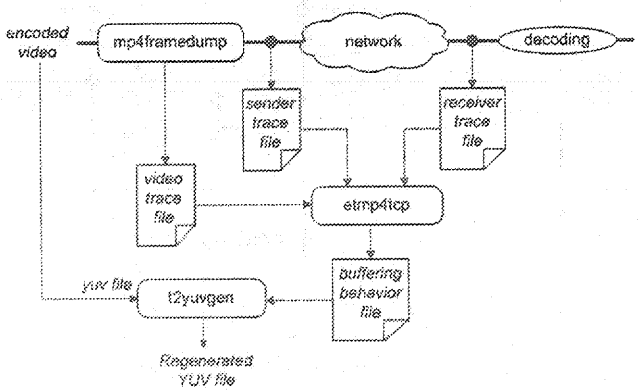

Figure 3 Overview of our validation framework. Based on trace files the initial start-up delay and the number of play-out buffer starvations are calculated.

By providing the buffering behavior file and the uncompressed YUV file of the original encoded video sequence as input to the *t2yuvgen* program, a new YUV sequence can be generated that visualizes the impact of the play-out buffer starvation on video fluidity playback. Suppose, for example, that after displaying frame x, a buffer starvation occurred and it took an additional 5 seconds to refill the play-out buffer. In this case, the duration of frame x will be adjusted so that it is displayed for the entire re-buffering period. Therefore, the validation framework can generate a new video for subjective quality assessment.

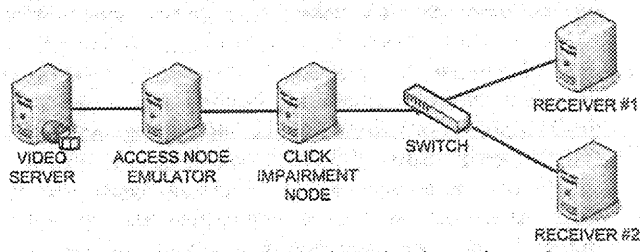

Figure 4 Physical test-bed used to investigate the performance of the play-out buffer starvation prediction algorithm. The impairment node can introduce losses or a limited amount of bandwidth.

The correctness of the validation framework has been validated by comparing the results obtained through the validation framework with the perceived play-out buffer starvations during playback of a video using the VLC [15] client. The same tests were carried out for varying levels of congestion and play-out buffer size. These tests indicated that the validation framework succeeds in detecting the length and position of each play-out buffer starvation.

5 Performance evaluation

We tested the performance of the play-out buffer starvation prediction algorithm on a physical test-bed as depicted in Figure 4. The test-bed emulates an xDSL-environment, consisting of a video server, an access node and two clients residing in the home network. The algorithm is deployed on an access node emulator. At the video server, a web server is running that accepts connections from the receiver who is downloading a progressive download stream. The impairment node is a Click Modular router [16] where packet loss can be introduced. The impairment node is also capable of limiting the bandwidth between the access node and the receiver.

We tested the performance of the algorithm by introducing different levels of congestion. In this case, a receiver plays a progressive download video requested from the video server. The video played is a 2 minutes long MPEG-4 video with a resolution of 720x540, a frame rate of 25 fps, an average bit rate of 2Mbps and a peak bit rate of 5.5Mbps. No packet loss occurs due to bit errors but the bandwidth is limited by the behavior of the second client who is watching a UDP based video stream. We varied the available bandwidth from 5.5Mbps to 1.5Mbps in steps of 0.25 Mbps. We tried to predict the number of play-out buffer starvations and the length of each starvation using the algorithm presented in section 3. In our tests, we defined a play-out buffer starvation as the moment when the buffer needs to be (re)-filled. Therefore, the initial startup delay is also seen as a play-out buffer starvation. As the number and length of play-out buffer starvations also depends on the size of the play-

out buffer we varied its size from 0.5 seconds to 5 seconds with steps of 0.5 seconds.

We present absolute and relative values. The absolute values compare average predicted values, obtained through the prediction algorithm, with the average exact values, obtained through the validation framework. These values indicate how the play-out buffer starvations react on varying congestion levels but do not give much information about the performance of the prediction algorithm itself. The relative values denote the accuracy of the prediction algorithm, defined as equation (3). In this case, the relative error is investigated. We use relative numbers as the importance of an error decreases when the amount of play-out buffer starvations increases too. Each test was repeated 25 times; we present average values together with the standard deviation.

$$accuracy = \frac{|\, predictedValue - realValue \,|}{actualValue} \qquad (3)$$

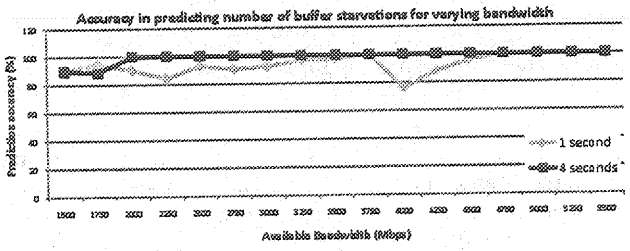

Figure 5 Accuracy of the prediction algorithm in terms of predicted number of play-out buffer starvations for a fixed play-out buffer size. As the available bandwidth increases, the accuracy increases as well.

5.1 Predicting the number of play-out buffer starvations

Figure 5 illustrates the average accuracy of the prediction algorithm for a fixed play-out buffer size of 1 second and 4 seconds. As illustrated, the achieved accuracy can be regarded as very good. For high levels of congestion, the achieved accuracy is around 90% and increases up to 100% when the congestion level decreases. This indicates that the prediction algorithm is able to almost exactly predict the amount of buffer starvations; only in a few rare occasions the algorithm misses one starvation. As the amount of available bandwidth decreases, the prediction algorithm makes a few mistakes, but the average prediction is still very good.

This high accuracy is also reflected in Figure 6, which compares the predicted number of play-out buffer starvations with the actual number, obtained through the validation framework. As the available bandwidth increases, the number of play-out buffer starvations decreases. The algorithm is able to predict this decreasing trend and achieves very good average values. Also the standard deviation values are reasonable, denoting a small variance on the obtained values. Figure 6 also indicates how the number of play-out buffer starvations increases gradually when the available bandwidth decreases from 5.5Mbps to 2Mbps. This limited number of starvations occurs when the bit rate of the video reaches its peak. We also see how the amount of play-out buffer starvations increases extremely when the available bandwidth drops below the average video bit rate (in this case 2Mbps). In this case, play-out buffer starvations occur almost continuously.

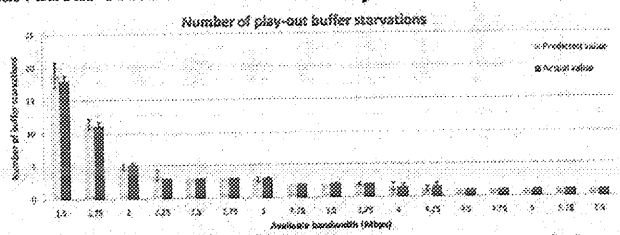

Figure 6 Comparison between the predicted number of play-out buffer starvations and the actual number for a fixed play-out buffer size of 1 second.

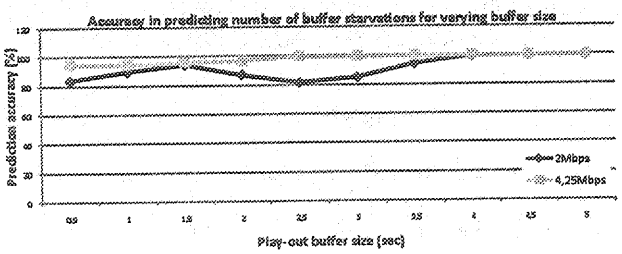

Figure 7 Accuracy of the prediction algorithm in terms of predicted number of play-out buffer starvations for a fixed amount of available bandwidth. As the play-out buffer size increases, the accuracy increases as well

Figure 7 illustrates the effect the play-out buffer size has on the accuracy of the prediction algorithm for an available bandwidth of 2Mbps and 4.25Mbps. In both cases, increasing the play-out buffer size has a beneficial effect on the accuracy and even leads to an accuracy of 100% for a high play-out buffer size (i.e 2.5 seconds for an available bandwidth of 4.25Mbps and 4 seconds for an available bandwidth of 2Mbps). The relative values are illustrated in Figure 8. For a small play-out buffer size, a lot of play-out buffer starvations occur but they will have a smaller length when compared to a high play-out buffer size. Again, we see that the accuracy decreases when there are more play-out buffer starvations (i.e. a small play-out buffer size). For a play-out buffer size of 0.5 seconds and an available bandwidth of 2Mbps, the accuracy drops to 83.95%. However, this result is still reasonable when compared to the absolute values. In this case, we measured an average of 14.16 play-out buffer starvations, while the prediction algorithm only predicted 11.8. As the actual number of play-

out buffer starvation is so high, the prediction algorithm can afford to make a few mistakes and still provide a reasonable indication of the QoE. For a 2 minutes long video sequence, the occurrence of 14 play-out buffer starvations will already degrade the QoE of the progressive download immensely. In this case, the actual accuracy of the algorithm is of less importance.

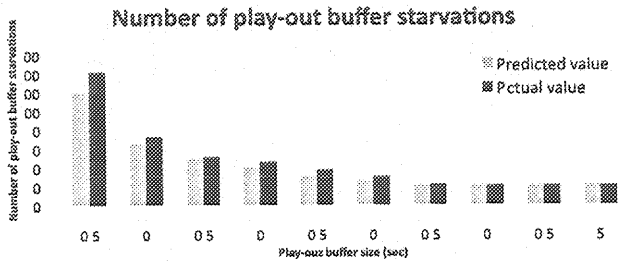

Figure 8 Comparison between the predicted number of play-out buffer starvations and the actual number for an available bandwidth of 2Mbps

Table 1 Overview of the measured standard deviation values on the accuracy for a varying play-out buffer size.

Available bandwidth	Play-out buffer size (sec) 0.5	2.5	5.0
1.5	6.71	3.95	1.41
3.5	2.51	1.37	0.12
5.5	1.61	0.78	0

Table 1 summarizes the obtained standard deviation values on the accuracy for a varying play-out buffer size and congestion level. These results show how an increasing level of congestion or packet loss or play-out buffer size increases the variance of the prediction algorithm. In both cases, the actual number of play-out buffer starvations increases. Therefore, it is clear that the more buffer starvations occur, the lower the accuracy of the prediction algorithm is. However, as explained earlier, its exact accuracy will also be of less importance since the QoE of the progressive download based service already is highly degraded.

5.2 Predicting the initial startup delay

There is one predicted play-out buffer starvation that can easily be mapped to an actual play-out buffer starvation in all circumstances. This is the play-out buffer starvation that occurs when the play-out buffer is initially filled. This initial startup delay gives an indication of how well the prediction algorithm can predict the length of each play-out buffer starvation.

Figure 8 denotes the accuracy in terms of initial startup delay under varying levels of congestion and two play-out buffer sizes. As illustrated, the accuracy is not really influenced by the varying levels of congestion and remains more or less stable. Although, the accuracy is somewhat lower than when predicting the number of play-out buffer starvations, these results are still very acceptable taking into account that it is not easy to estimate the duration of an occurring event. The accuracy of the prediction algorithm is approximately 89% and 81% for a play-out buffer size of 1 and 4 seconds, respectively. These values mean that, for an initial startup delay of 1 second the prediction algorithm predicts the delay to be 0.89 seconds.

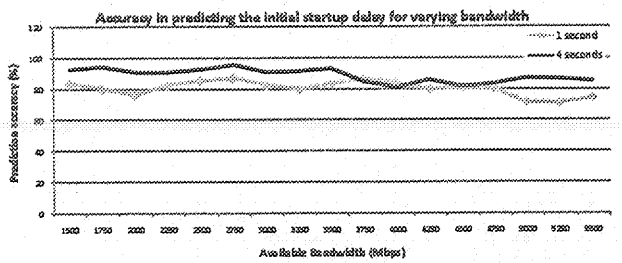

Figure 9 Accuracy of the prediction algorithm in terms of initial startup delay for a fixed play-out buffer size.

The accuracy of the prediction algorithm in predicting the initial startup delay for a varying play-out buffer size is depicted in Figure 9. Similar to predicting the number of play-out buffer starvations, increasing the play-out buffer size has a beneficial effect on the accuracy of the prediction algorithm. However, in this case, the impact is not very high. Again, the overall accuracy is still very reasonable, although not as high as in the prediction of the number of play-out buffer starvations.

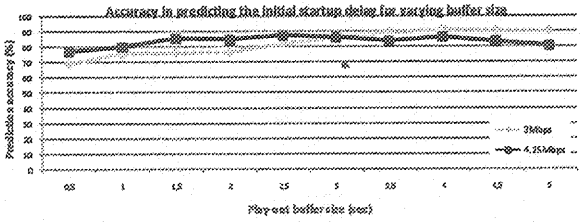

Figure 10 Accuracy of the prediction algorithm in terms of initial startup delay for a fixed congestion level.

The overall accuracy of the prediction algorithm can be seen as very good in predicting the amount of play-out buffer starvations as in this case the average accuracy is approximately between 80% and 100%. The accuracy of the initial startup delay prediction is reasonable with an average accuracy between 75% and 90%. Although the prediction algorithm cannot provide an exact prediction of how many frames a starvation lasted, it provides a good indication of how long the starvation was.

6 Conclusion

In this paper, we proposed an algorithm to predict the QoE of progressive downloads in real-time. The algorithm is deployed on an intermediary node in the access network and only depends on the flow of TCP data and acknowledgement packets and does not require additional feedback from the client side. Therefore, the algorithm can be deployed in the access network to monitor the QoE of services running to the home network. Based on network level information the algorithm estimates when each packet arrives at the client side. Metadata information from the MP4 container is then used to predict when a play-out buffer starvation occurs resulting in the video player needing to rebuffer and how long each play-out buffer starvation takes.

We investigated the performance of this algorithm in different scenarios using a novel validation framework that is able to measure the QoE at the client side. Tests show that the prediction algorithm is very accurate in determining the number of play-out buffer starvations and has a reasonable accuracy in determining the initial startup delay.

7 Acknowledgement

The research was performed partially within the framework of the CELTIC RUBENS project. Steven Latré would like to thank the Special Research Fund of Ghent University (BOF) for financial support through his Ph.D. grant. Nicolas Staelens would like to thank the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT) for financial support through his Ph.D grant. Filip De Turck would like to thank the Research Foundation Flanders (FWO-V) for support through his post-doctoral grant.

8 References

[1] ISO/IEC 14496-14:2003 Information technology - Coding of audio-visual objects - Part 14: MP4 file format, International Organization for Standardization, Geneva, Switzerland

[2] S. Mohamed, G. Rubino, "A study of real-time packet video quality using random neural networks", IEEE Trans. Circuits Systems Video Technology". 12 (12) (2002) 1071--1083.

[3] E. Bertini, G. Santucci, "Visual quality metrics", Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization, 2006, pp. 1-5

[4] M. Masugi, "QoS mapping of VoIP communication using self-organizing neural network", in: IEEE Workshop on IP Operations and Management, 2002, pp. 13-17

[5] M. Claypool and J. Tanner, "The effect of jitter on the perceptual quality of video", in ACM Multimedia 1999, Orlando, FL, 1999, pp. 115-118

[6] S. R. Gulliver and G. Ghinea, "The perceptual and attentive impact of delay and jitter in multimedia delivery", in IEEE Transactions on Broadcasting, vol. 53, no 2, June 2007, pp. 449-458

[7] R. R. Pastrana-Vital and J. Gicquel, "Automatic quality assessment of video fluidity impairments using a no-reference metric", in VPQM, Scottsdales, 2006

[8] J. Bernstein, T. Spets, "CPE WAN Management Protocol (TR-069)", DSL Forum, May 2004

[9] P. Simoens, B. De Vleeschauwer, W. Van de Meerssche, F. De Turck, B. Dhoedt, P. Demeester, "RTP connection monitoring for enabling autonomous access network QoS management", Proceedings of NOC2007, the 12th European Conference on Networks and Optical Communications, 2007

[10] B. De Vleeschauwer, P. Simoens, W. Van de Meerssche, F. De Turck, B. Dhoedt, P. Demeester "Enabling autonomic access network QoE management through TCP connection monitoring", Proceedings of ACNM2007, the 1st IEEE Workshop on Autonomic Communications and Network Management, 2007

[11] J. Klaue, B. Rathke, A. Wolisz, "EvalVid - A Framework for Video Transmission and Quality Evaluation", In Proc. of the 13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, pp. 255-272, Urbana, Illinois, USA, September 2003

[12] S. Latré, F. De Turck, B. Dhoedt, and P. Demeester. Scalable simulation of QoE optimization for multimedia services over access networks. In International Conference on Internet Computing (ICOMP), Las Vegas, Nevada, 2007

[13] The Network Simulator - NS-2, [online] http://www.isi.edu/nsnam/ns/

[14] D. De Vera, P. Rodriguez-Bocca, G. Rubino. QoE Monitoring Platform for Video Delivery Networks. 7th IEEE International Workshop on IP Operations and Management (IPOM 2007). IEEE CS Press, San José, USA, November 2007.

[15] VLC Media Player, [online] http://www.videolan.org/vlc/

[16] The Click Modular Router Project, [online] http://www.read.cs.ucla.edu/click/

PROCEEDINGS OF
THE 2008 INTERNATIONAL CONFERENCE ON
INTERNET COMPUTING

ICOMP'08

Editors

Hamid R. Arabnia
Victor A. Clincy

Associate Editors

Ashu M. G. Solo
Joan Lu

WORLDCOMP'08
July 14-17, 2008
Las Vegas Nevada, USA
www.world-academy-of-science.org

ISBN: 1-60132-073-6
Printed in the United States of America