Department of Computer Science, University of Otago

Technical Report OUCS-2016-02

The Impact of IP Network Impairments on Optimal Playback Buffer Size in Video Streaming

Authors:

Lahiru Ariyasinghe, Zhiyi Huang, David Eyers

Department of Computer Science, University of Otago, New Zealand

---

Department of Computer Science,
University of Otago, PO Box 56, Dunedin, Otago, New Zealand

http://www.cs.otago.ac.nz/research/techreports.php

The Impact of IP Network Impairments on Optimal Playback Buffer Size in Video Streaming

Lahiru Ariyasinghe
Department of Computer Science
University of Otago
Dunedin, New Zealand
lahiru.a@cs.otago.ac.nz

Zhiyi Huang
Department of Computer Science
University of Otago
Dunedin, New Zealand
hzy@cs.otago.ac.nz

David Eyers
Department of Computer Science
University of Otago
Dunedin, New Zealand
dme@cs.otago.ac.nz

ABSTRACT

A key challenge for online video streaming services is how to deliver their data over networks that suffer packet losses and delays while maintaining a good Quality of Experience (QoE). Metrics such as start-up delay, the count of the number of times that re-buffering occurs and re-buffering delays provide useful indicators to the streaming services to measure the impact of IP network impairments (e.g. packet loss and delay) on overall video stream quality. Playback buffering is one of the key application-level techniques that can mitigate the impact of network impairments and protect the video quality by sensibly balancing the effect of the above metrics. However for the mitigation to be effective, while maximising user QoE, setting an optimal playback buffer size is vital. In this paper, a comprehensive analysis is performed to investigate the impact of packet loss and link delay on optimal playback buffer sizing, with respect to video files that contain different amounts of motion. Experimental results indicate that the buffer size that delivers an optimal start-up delay with acceptable levels of playback disruption remains reasonably constant, when changing the degree of motion in the video, and in response to minor variation of link delay and small amounts of packet loss. This is in contrast to the buffer size that ensures no interruptions to playback, which, as expected, rises steadily.

Keywords

Streaming Video, Optimal Buffer Size, Network Impairments, Multimedia, Dummynet

1. INTRODUCTION

There is a rapidly increasing demand for streamed video on the internet. According to the latest Total Audience Report [14], online video streaming viewers are rising at an astonishing rate of 60% per month. Video streaming websites such as YouTube, Hulu and MSN video offer thousands of easily accessible videos to end users. Online video streaming is becoming popular primarily due to its great flexibility and convenience for users. Users can enjoy real-time video rather than waiting for the entire video file to download since video streaming allows users to play the video simultaneously while the video file is still being delivered from the remote sever.

Usually, video data is encoded as frames, which are displayed at fixed frequencies. As video data arrives at the client side, data is positioned into a buffer to be decoded and displayed on the screen at the correct time.

Over time, a large number of video streaming protocols have been developed. However most of the commercial online video streaming services use HTTP to send data to the receivers. One appealing aspect of HTTP is the relative ease with which it can pass through firewalls. In addition there is now a plentiful supply of highly performant HTTP servers.

In a classical streaming session, the client requests a video file from the webserver over HTTP and plays it as video data arrives from the server. One popular example is YouTube, which uses HTTP servers for its progressive download [19].

Regardless of the significant convenience of streaming compared to having to first download videos, user satisfaction when performing video streaming remains a great uncertainty. A research shows that on the Internet, about 13% of home and 40% of business streaming sessions suffer various types of quality degradation [12].

Eventually online video streaming demands the presence of a smooth and flexible sending rate to achieve an acceptable Quality of Experience (QoE). However in times of network congestion, queues can build up inside the routers that cause delays or the dropping of network packets. When this happens, TCP congestion control algorithms will usually abruptly decrease the transmission rate, which will ultimately cause a bandwidth variation and can damage the quality of video streaming.

Techniques such as *adaptive bitrate streaming* and *client side playback buffering* can be employed to eliminate this problem.

Adaptive bitrate streaming [4] is a technique that allows changing the streamed bitrate depending on the detection of congestion. It works by dynamically monitoring CPU and memory capacity and then making corresponding adjustments to video quality. The heart of the process involves encoding the source video at varying bit rates, and then segmenting each of the different bit rate streams into small parts. The client-side player can switch among the different bitrate segments, locating the segments that correspond best to the bandwidth on the user's computer.

Alternatively, client-side playback buffering involves the application maintaining a buffer to alleviate degradations caused by undesirable changes in data rate. Packets are temporarily stored at the client buffer in order to smooth out bandwidth variation.

Figure 1 shows the role of the playback buffer in video streaming. The fill rate can be defined as the rate that data enters the buffer, and the drain rate can be defined as the

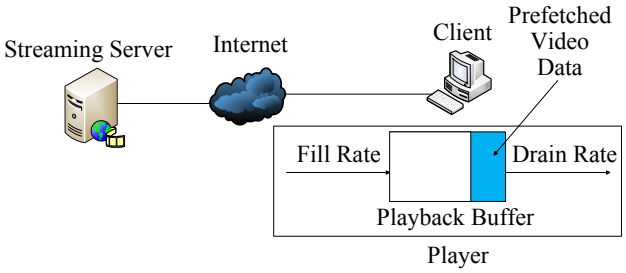

Figure 1: Role of the Playback Buffer

rate that data leaves the buffer. As video data arrives at the client side from the server, data is fed into the client buffer at the fill rate, and then pulled out at the drain rate and is decoded and displayed. With buffered data, the receiver is able to smooth over temporary drops in the received rate.

Choosing a suitable buffer size is important. The smaller the buffer setting, the sooner playback can begin, since the buffer can be filled quickly. However, when TCP congestion control reduces the fill rate below the drain rate, it may cause the buffer to empty and result in unwanted pauses in the video playback. If a large buffer size is set, initial start-up time will be higher, since the player needs to wait for more data to transfer into it at the start, but the experience during playback should be better because there is less chance of the buffer emptying.

Within this paper we analyse the effect of link delay and packet loss on the following metrics namely:

Number of re-buffer events: This is the number of times that playback freezes due to an empty buffer.

Re-buffering delay: When playback freezes because the playback buffer empties, this is the time taken to fill the buffer and resume playback.

Start-up delay: After a viewer has clicked *play*, this is the time taken to fill the buffer and start playback.

The main contribution of this work is to suggest the most appropriate playback buffer sizes to be used to strike a reasonable balance between the above metrics in the presence of different packet loss and link delay values.

The rest of this paper is organised as follows. In Section 2 we describe the properties of tested video samples, selected network settings, definition of buffer optimality and our methodology of testing. Section 3 presents our video streaming testbed used in our analyses. Section 4 presents the experimental results and our suggested means of determining optimal buffer sizes. Section 5 provides related work and background. Finally we conclude our paper and discuss the future work in Section 6.

Video	Frame Rate (fps)	Average Bit Rate (Kbps)
FMV	29	2047
MV	23	1362
SMV	23	1404

Table 1: Attributes of the tested video samples used in this study

2. METHODOLOGY

This section describes the properties of the tested video samples, selected network settings, definitions of playback buffer optimality and the methodology for our experiments.

2.1 Tested video samples

Three different video clips with different types of motion are considered in this study: a Fast Motion Video[1] (FMV: relatively a higher percentage of fast motion), a Slow Motion Video[2] (SMV: relatively a higher percentage of slow motion), and a Music Video[3] (MV), which is a mix of both fast and slow motion. Videos with different dominant rates of motion were used to check the stability of our parameter recommendations.

The video traces 'Foreman', 'Akiyo' and 'News' used as benchmarks in studies like [13] were found difficult to use within our experiments, due to the fact that they are relatively short in length (all of them contain 300 frames each) [7]. Nevertheless, compared to our study, those video traces are heavily used in studies that are done at the video frame level. The lowest playback buffer size that can be set in our real experimental set-up would be able to prefetch all the video frames before starting up the playback. Although we cannot directly use these standard benchmarks, we are confident that our FMV, SMV and MV videos are sufficiently representative because of the acceptable playback times and appropriate levels of motion.

All the video samples we use are downloaded from YouTube and the music video is among the top ten most popular YouTube videos all the time [3]. All are in (`.mp4`) format (`video codec: H.264-MPEG-4 Part 10 and audio codec: MPEG AAC`) with a HD resolution of 1280×720 ('720p'). Audio-related properties such as audio sampling rate of 44 KHz and 2 channel stereo are the same. The total lengths of the FMV, MV and SMV videos are 3:00 (i.e. three minutes), 3:09 and 2:23 respectively. Other important characteristics are summarised in Table 1.

2.2 Network configurations that were tested

The parameter space of all possible network performance constraints is too large to investigate exhaustively. We have derived a subset of values to explore for parameters such as bandwidth, link delay and packet loss from published literature.

2.2.1 Bandwidth

The 2014 Internet Service Provider Survey of Statistics New Zealand [20] states that download speeds of over two-thirds of the broadband internet connections in the country are in the range of 8–24 Mbps, and upload speeds of almost half of the connections are in the range of 1.5–10 Mbps.

[1]https://www.youtube.com/watch?v=LZ3BLcBz9Ls
[2]https://www.youtube.com/watch?v=YH7uhgPD0gY
[3]https://www.youtube.com/watch?v=7PCkvCPvDXk

Download Pipe Bandwidth (Mbps)	16
Upload Pipe Bandwidth (Mbps)	5.75

Table 2: Maximum link bandwidth configurations used for testing

Therefore we have used the average values of above speed categories as the maximum download and upload speeds of the proposed experimental setup.

The maximum link bandwidth configurations tested are listed in Table 2.

2.2.2 Packet loss rate

Les Cottrell et al. have defined the quality levels for internet packet loss rates [5]: 0–0.1% as *Excellent*, 0.1–1% as *Good*, 1–2.5% as *Acceptable*, 2.5–5% as *Poor*, 5–12% as *Very Poor* and greater than 12% as *Bad*.

We have considered the margins of above quality categories up to the *Acceptable* level to be tested with our experimental setup. We thus tested packet loss rates: 0.1%, 1.0% and 2.5%.

2.2.3 Link delay

In the internet, for real-time multimedia, Calyam et al. [16] define the one way delay of (0–150) milliseconds as *Good*, (150–300) milliseconds as *Acceptable*, and values greater than 300 milliseconds as *Poor*.

We have again used the margins of the above quality categories up to the *Acceptable* level to select values for our experimental setup. We thus tested link delay values 150 and 300 milliseconds.

2.3 Definition of buffer optimality

Within this study exploration of the optimal buffer size requirements is based on the chosen performance metrics (defined in the section 1) namely: the number of re-buffering events, the re-buffering delay and the start-up delay.

We aim to find the playback buffer sizes that lead to optimal values for the above metrics. Clearly the optimal value for the number of re-buffering events is zero. With respect to the optimal values for the second and third metrics we refer to results that we have found in research literature.

One prominent study [17] revealed that an increase in the startup delay beyond 2 seconds causes viewers to abandon the video. Using regression, they have shown that an additional increase of the startup delay by 1 second increases the abandonment rate by 5.8%. The study further determined that a viewer who experienced a re-buffer delay that equals or exceeds 1% of the video duration, played 5.02% less of the video in comparison to a similar viewer who experienced no re-buffering.

Note that based on [17] optimal values for re-buffering delays are computed using the selected video length. A re-buffering delay less than 1% of the video duration is taken as optimal. We assume that this re-buffering delay will be tolerable to the viewers, and refer to it as the *tolerable re-buffering delay*.

Based on the above studies and the lengths of our videos (FMV, MV, and SMV), we define the optimal values for start-up delay and re-buffering delay as follows.

- Optimal start-up delay: less than or equal to 2 seconds
- Tolerable re-buffering delay for the FMV: less than 1800 milliseconds
- Tolerable re-buffering delay for the MV: less than 1890 milliseconds
- Tolerable re-buffering delay for the SMV: less than 1430 milliseconds

Relatively larger buffers sizes tend to reduce the number of re-buffering events to zero with the cost of having increased start-up delay and re-buffering delays. In contrast, smaller buffer sizes tend provide ideal values for the start-up delay but network impairments may cause a considerable number of re-buffering events. Therefore, to cater for this tradeoff for each network setting we have defined two measures of optimality based on the above metrics,

Optimal Buffer Size one (OBS_1): The minimum playback buffer size that causes no re-buffering events (ideal value of the first metric) is taken as an optimal buffer. With OBS_1, we can study the behaviour of the second metric (start-up delay), finding out whether it resides within its ideal range (less than or equal to 2 seconds) or not.

Optimal Buffer Size two (OBS_2): The maximum playback buffer size that provides the ideal value for the second metric (start-up delay of within 2 seconds) is taken as the optimal buffer. With OBS_2, we can study the behaviours of the number of re-buffering events and re-buffering delay, discussing whether they reside within their ideal ranges or not.

OBS_1 may enable an optimal start-up delay, O_{SD}, but this is not guaranteed.

OBS_2 may cause zero or more re-buffering events. In the case of re-buffering events, the re-buffer delays observed can be tolerable re-buffering events or otherwise.

2.4 Testing procedure

For each selected link delay or packet loss rate, we start streaming of video data with no playback buffering. Then we progressively increased the buffer size by intervals of 2 seconds, until we obtain the most appropriate buffer sizes for OBS_2 and OBS_1. Our experiments are conducted across a wide set of parameter values. All results presented in the paper are obtained by repeating each video streaming test 10 times for a given buffer size.

3. EXPERIMENTAL SETUP

In this section we provide an overview of the tools used in our video streaming test-bed and a detailed description of our experimental setup. We start by introducing each tool, as well as describing its role within the experimental setup.

3.1 Dummynet

Dummynet [1] is a widely used network link emulator, which is capable of running experiments in user-configurable network environments. It is a part of the FreeBSD kernel, but is also available for Linux and Windows platforms. The core concepts behind Dummynet are *rules* and *pipes*. Rules decide which incoming or outgoing packets must pass through which pipes. These pipes can be configured with certain network parameters, including a maximum bandwidth, packet loss, and delay.

Within our testbed we use Dummynet to emulate diverse network behaviours by configuring different packet loss rates and link delays for a given maximum link bandwidth.

3.2 VLC media player

VLC media player [6] is a highly portable, free and open-source, cross-platform media player that supports many audio and video compression methods and file formats. It also provides rich server capabilities for streaming live and on-demand video. It supports several formats, including MPEG-1, MPEG-2 and MPEG-4, and can stream using unicast or multicast communication.

Within the our testbed, VLC media player acts as a streaming server at one node and as a player at the other node. The VLC server uses HTTP as the stream output method and ffmpeg [2] as the encoder.

3.3 Testbed

As shown in Figure 2, our testbed consists of two PCs connected to the departmental LAN. PC_1 acts as the streaming server and PC_2 acts as the client. Both machines are running VLC version 2.2.1 on top of Microsoft Windows. Specifically, the server machine is running Windows 7 32-bit while the client machine is running a Windows 7 64-bit version. The difference in the OS versions is due to Dummynet only supporting 32-bit Windows versions. The server and client machines run scripts to configure VLC as a streaming server and as a receiving client respectively. Another script sets the appropriate Dummynet network parameters namely: maximum bandwidth, link delay and packet loss rate. The VLC server sends video data to the client. While receiving network data and then displaying it, the VLC client logs details of the streaming session.

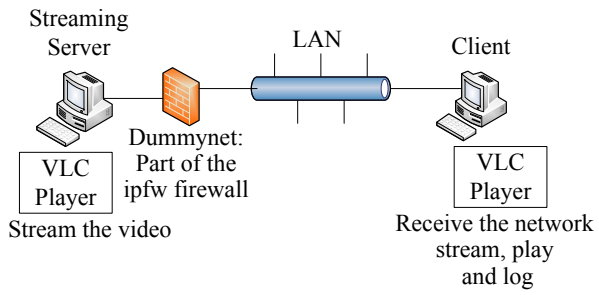

Figure 2: Experimental Test-bed

4. EXPERIMENTAL RESULTS

We tested three types of videos: FMV, SMV and MV. Their results are presented in the following sections. In the experiments, when we study OBS_1 and OBS_2 for a range of link delays, we set the packet loss rate to be zero. Similarly, when we study OBS_1 and OBS_2 for a range of packet loss rates, link delay is set to zero in our testbed. Actually there is a very small delay in the LAN of our testbed, but the delay is small enough to be negligible compared with our configured Dummynet delay of, say 150 ms.

4.1 Behaviour of FMV

This section presents our results obtained studying FMV, and our suggestions of optimal buffer sizes for the chosen link delays and packet loss rates.

4.1.1 Effect of link delay

When the link delay is 150 ms, we recorded a median of 12 re-buffering events and almost no start-up delay (median of 0 seconds within the 10 experiments) with no playback buffering. Figures 3, 4 and 5 present the results we observed when playback buffering is active. According to Figures 3 and 4 it is clear that 10 seconds of playback buffer can be taken as OBS_1, with a non-optimal start-up delay of 10.4 seconds.

For all the 10 experiments performed, using a 10 second buffer consistently caused no re-buffering events. However it has increased the start-up delay considerably, compared to our optimal value of 2 seconds. Irrespective of missing this target, in the presence of tested network conditions, users are very likely to have no re-buffering events with a 10 second playback buffer.

Furthermore it can be shown that a 2 second buffer size can be taken as OBS_2 with a median of 2 re-buffer occurrences. Comparing Figure 5, with OBS_2, of the re-buffering events that occur, the median value is 0.5 for the re-buffering events that can be interpreted as tolerable.

If the preference is given to an optimal start-up delay, we suggest to use a 2 second playback buffer. Out of the 10 experiments performed, in 9 it provided us a start-up delay less than our acceptable value of 2 seconds. So we would extrapolate that users have a 90% probability of observing an optimal start-up with a buffer size of 2 seconds. But ultimately it can result in re-buffering events, and out of those a median less than 50% can be tolerable to the viewers.

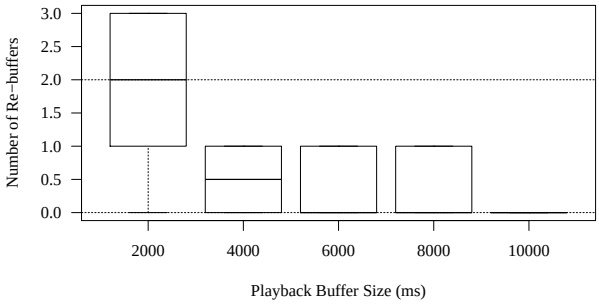

Figure 3: FMV: 150 ms Link Delay

When the link delay is increased to 300 ms, we note that the median of re-buffer events is 10 which comes with almost no start-up delay (median of 0 seconds within the 10 experiments) when playback buffering is inactive. Figures 6, 7 and 8 outline our observed results when playback buffering

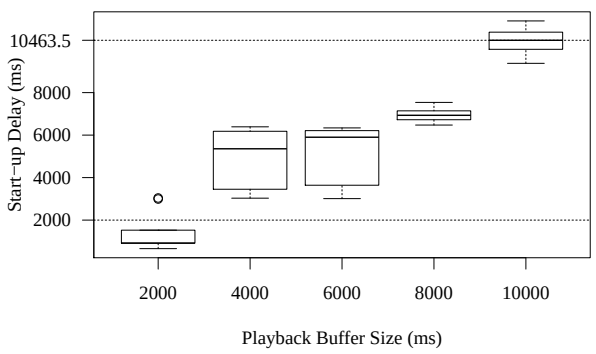

Figure 4: FMV: 150 ms Link Delay

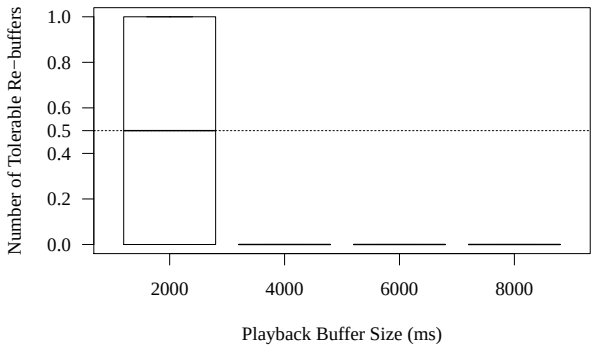

Figure 5: FMV: 150 ms Link Delay

is employed. According to Figures 6 and 7 we suggest that 14 seconds of playback buffer can be taken as OBS_1 with a non-optimal start-up delay of 20.5 seconds. For the 10 experiments performed, in 9 of them we observed no re-buffering events with the 14 second buffer. As in the previous scenario, a 2 second buffer size can be taken as OBS_2 with a median of 3.5 re-buffering events, and as seen on Figure 8 with OBS_2, out of the observed re-buffering events none of them can be classed as tolerable.

Therefore with an increased link delay, if the priority is given to having no interruptions in the video, our suggestion would be to use a buffer size of 14 seconds. With almost a 90% probability we expect the 14 second buffer to deliver no re-buffering events. Otherwise it is highly likely for the 2 second buffer to deliver an optimal start-up delay given the fact that it may associate with some intolerable re-buffering events.

4.1.2 Effect of packet loss

With the FMV, when playback buffering is inactive, we note median re-buffering events of 1, 1 and 4 for the induced loss rates 0.1%, 1.0% and 2.5% respectively. However with-

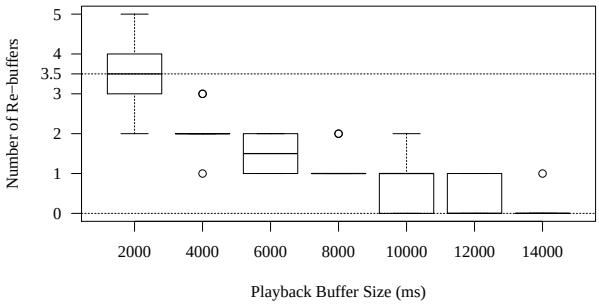

Figure 6: FMV: 300 ms Link Delay

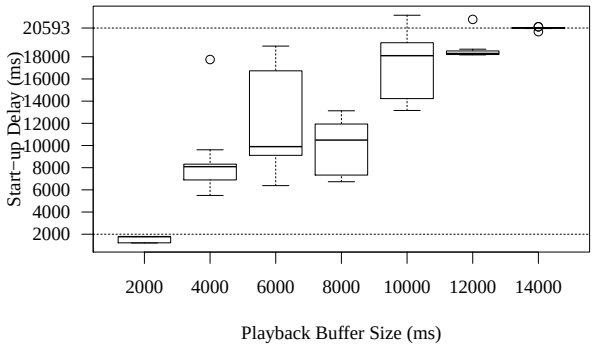

Figure 7: FMV: 300 ms Link Delay

out playback buffering we note almost no delay in start-up (median of 0 seconds within the 10 experiments) with all the tested loss rates. Eventually with a playback buffer of 4 seconds we observe a median of zero re-buffering events for all the above test cases. Therefore 4 seconds can be taken as OBS_1. Figures 9, 10 and 11 present the corresponding start-up delays. It is clear that for all the tested loss rates a 4 second buffer keeps delivering non-optimal start-up delays of (2.484, 2.489 and 2.467) seconds respectively. The 2 second buffer can be chosen as OBS_2 which consistently delivers optimal start-up delays in all the cases. Consequently with the 2 second buffer we recorded median re-buffering events of 1, 1 and 3.5 respectively. But our results indicate all of them as tolerable.

Therefore in the presence of the above loss rates, if the priority is given to the start-up delay, our suggested buffer size of 2 seconds is highly likely to deliver an optimal start-up to the viewers. However 2 seconds may come up with a few re-buffering events, but they are of a tolerable nature. If the importance is having no interruptions, our suggested 4 second buffer is highly likely to deliver such with a slightly increased, and non-optimal, start-up delay.

4.2 Behaviour of SMV

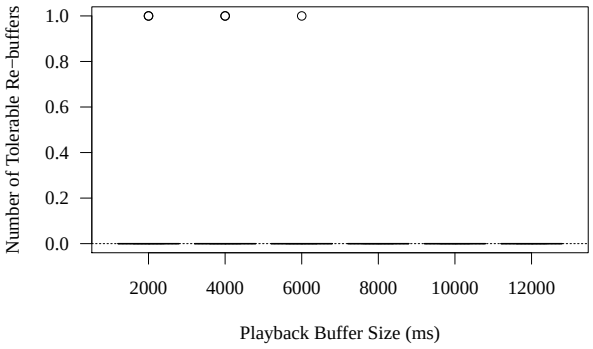

Figure 8: FMV: 300 ms Link Delay

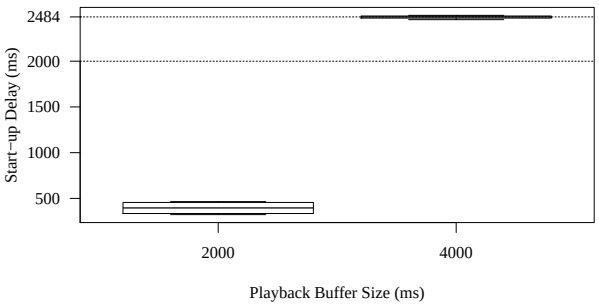

Figure 9: FMV: 0.1% Packet Loss

This section presents our results obtained studying SMV and our suggestions of optimal buffer sizes for a range of link delays and packet loss rates.

4.2.1 Effect of link delay

For a link delay of 150 ms, we record a median of 9 re-buffering events and almost no significant start-up delay (median of 0 seconds within the 10 experiments) with no playback buffering. Results obtained when playback buffering is active are presented in Figures 12, 13 and 14 respectively. According to Figures 12 and 13 it can be concluded that 6 seconds of playback buffer can be taken as OBS_1 which associates a non-optimal start-up delay of 4.8 seconds. The 2 second buffer size can be nominated as OBS_2 with a median of 1.5 re-buffering events. Figure 14 provides an indication of the tolerable re-buffering events. With the chosen OBS_2, out of the detected re-buffering events, none can be classed as tolerable, in the terms we have defined.

When the link delay is 150 ms, we suggest a 6 second playback buffer for the viewers who are watching comparatively slow motion videos. With such a buffer size users are highly unlikely to encounter re-buffering events. But as a consequence they will have to tolerate an increased start-up delay. Our suggestion for OBS_2 is having an 80% chance

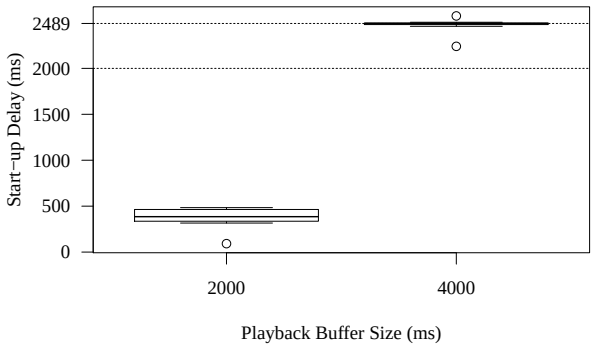

Figure 10: FMV: 1.0% Packet Loss

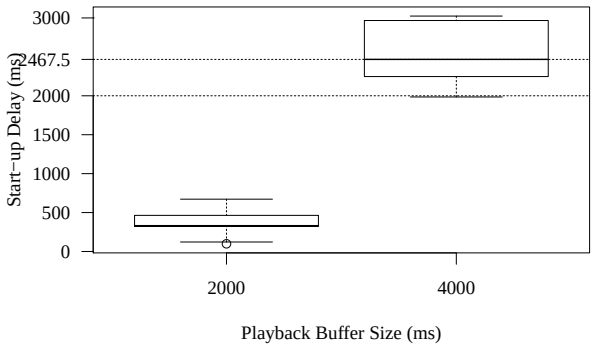

Figure 11: FMV: 2.5% Packet Loss

of providing an optimal start-up delay. But there is a high probability for the re-buffering events that occur with this OBS_2 to be classed as intolerable.

With an increased link delay of 300 ms, we record a median of 6 re-buffering events and an insignificant start-up delay (median of 1 ms within the 10 experiments) when playback buffering is inactive. According to Figures 15 and 16 we select a 10 second buffer as OBS_1, with a 2 second buffer as OBS_2. OBS_1 is associated with a non-optimal start-up delay of 17.9 seconds while OBS_2 is associated with a median of 1.5 re-buffering events. However our results do not report any of the above re-buffering events as being tolerable.

If the viewer preference is more towards having no re-buffering events, we suggest a buffer size of 10 seconds but note the excessively increased start-up delay. We expect the 10 second buffer to deliver no re-buffering events with a 90% probability. As in the previous experiments, a 2 second buffer should deliver a start-up delay well below the minimum acceptable value. However with such a buffer size, viewers are likely to have a small number of re-buffering events that would be classed as intolerable.

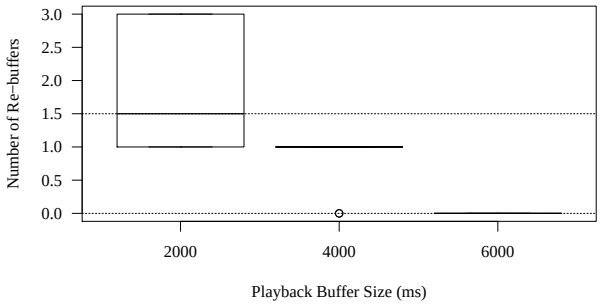

Figure 12: SMV: 150 ms Link Delay

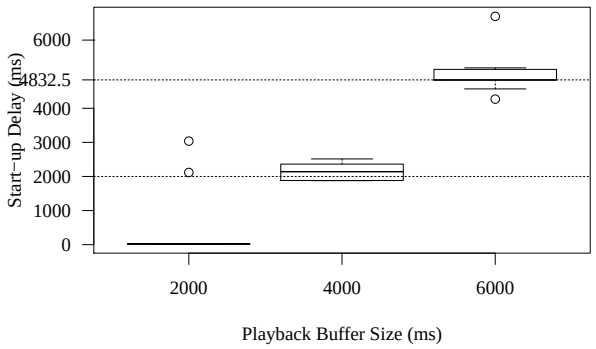

Figure 13: SMV: 150 ms Link Delay

4.2.2 *Effect of Packet Loss*

For SMV, when the packet loss rates are 0.1%, 1.0% and 2.5%, we observe median re-buffering events of 0, 1.5 and 1 respectively with no playback buffering. Correspondingly, for each case we record almost no delay in start-up (median of 0 seconds within the 10 experiments). However with a playback buffer of 2 seconds we record a median of 0 re-buffering events for all the above loss rates. Resultant median start-up delays are given in Figures 17, 18 and 19. It is clear that median start-up delays in all cases are well below our threshold, which is 2 seconds. We conclude, with the above tested loss rates, for the SMV, our OBS_1=OBS_2=2 seconds.

Therefore we suggest that in the presence of above loss rates, by using a 2 second buffer, viewers are highly likely to have an optimal start-up delay with almost no interruptions.

4.3 Behaviour of the music video (MV)

As a supplementary experiment we conduct the same testing with a music video, for a selected link delay and a packet loss rate. We present only the significant results—further details are available on request.

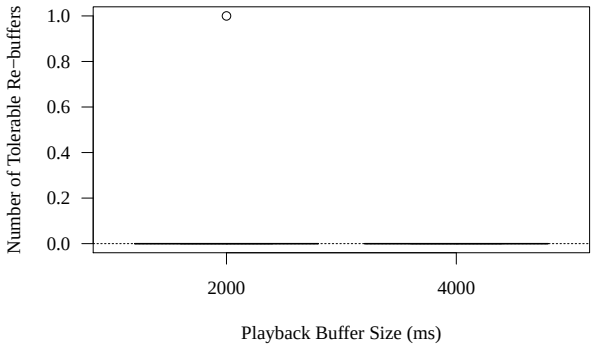

Figure 14: SMV: 150 ms Link Delay

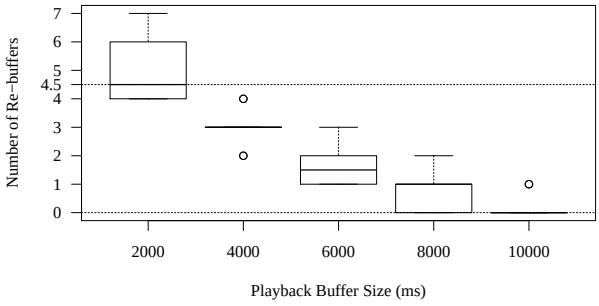

Figure 15: SMV: 300 ms Link Delay

For the MV with a link delay of 150 ms, we record a median of 9 re-buffering events and almost no start-up delay (median of 0 seconds within the 10 experiments) with no playback buffering. Results obtained when playback buffering is enabled are presented in Figures 20, and 21 respectively. From these figures it is clear that an 8 second playback buffer can be taken as OBS_1 which associates a non-optimal start-up delay of 17.1 seconds. Still the 2 second buffer size can be chosen as OBS_2 with a median of 2 re-buffering events. However Figure 22 suggests there is a high probability for the above re-buffering events to be classed as intolerable (median is 0). Therefore we interpret that with OBS_2, out of the noted re-buffering events, nothing significant can be taken as tolerable.

So, as for the SM video, when the packet loss rate is 2.5%, the MV measurements indicate that 2 seconds is suitable for both OBS_1 and OBS_2.

Suggestions of optimal buffer sizes presented in this section are summarized in the Tables 3, 4 and 5.

5. RELATED WORK

This section gives an overview of the related work on the problem of determining the best playback buffer size in video

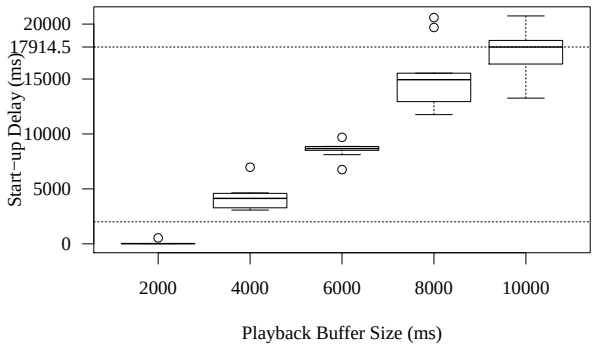

Figure 16: SMV: 300 ms Link Delay

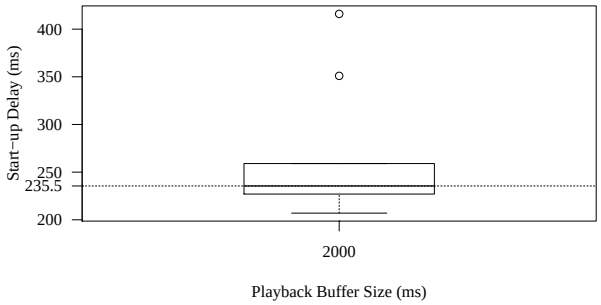

Figure 17: SMV: 0.1% Packet Loss

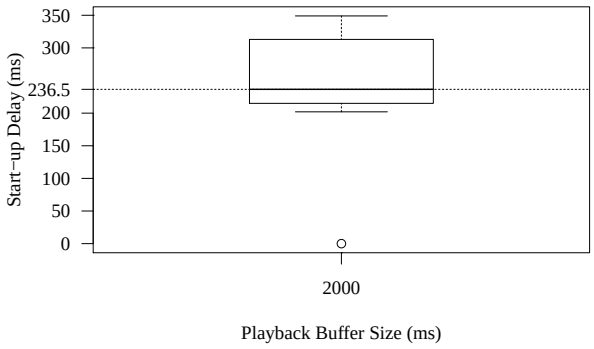

Figure 18: SMV: 1.0% Packet Loss

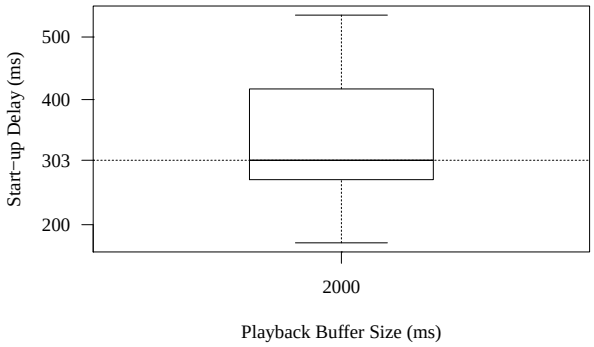

Figure 19: SMV: 2.5% Packet Loss

streaming.

Commercial video streaming has shifted away from custom protocols such as RTSP to HTTP due to its widespread support, including that most network firewalls permit it. Outside of custom protocols for video streaming, there seem to be very few studies that relate buffer sizes to video streaming quality of experience.

An analysis on the buffer size for video streaming over HTTP was carried out by Nukhet and Turhan [15]. Within their work, they collected over 1000 hours of video over LANs and WANs. Their experimental results suggest using a buffer size of 5 seconds when bandwidth estimation is impossible. We focus instead on finding the best buffer size when link quality measurements (*i.e.*, amount of packet loss and link delay) are available.

According to Zambelli [19], in early versions of Windows Media Player and Silverlight, the default buffer size was 5 seconds. Later, Microsoft introduced a novel method of delivery called progressive download over HTTP, where the video content is split into multiple short chunks and encoded to the chosen delivery format. Chunks are typically a few seconds long, with the client requesting the individual video chunks from the web server.

Akshabi, Begen and Dovrolis [8] have performed experiments to measure how Microsoft Smooth Streaming and Netflix players react to persistent and short-term bandwidth variations. The authors used Wireshark to capture traffic and Dummynet to change available bandwidth. Their experimental results indicate that playback buffer size in Smooth Streaming decreases when the available bandwidth is less than the requested bitrate and increases when the available bandwidth increases. However, Netflix employs a large playback buffer (up to a few minutes) and sometimes changes to bitrates higher than the available bandwidth as long as the playback buffer is almost full. Unlike [19] and [8] our emphasis is to provide more general suggestions for the ideal buffer sizes without coupling to a specific commercial player or streaming service.

Goel, Krasic and Walpole [9] showed that a significant fraction of the latency at the application layer occurs at the sender side of TCP due to throughput-optimised TCP implementations. They have developed an adaptive buffer-size for the sender that reduces this latency. A slight modification has been made to the TCP stack on the sender that can be enabled per socket. Their modification limits the send buffer size to fixed parameters, allowing a trade-off between

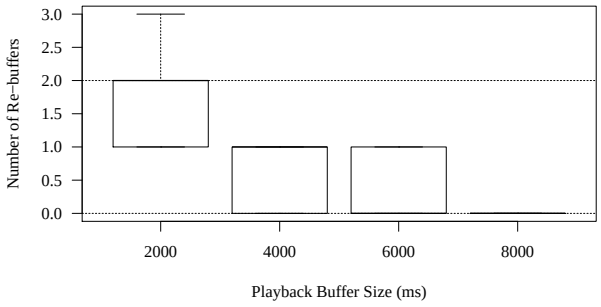

Figure 20: MV: 150 ms Link Delay

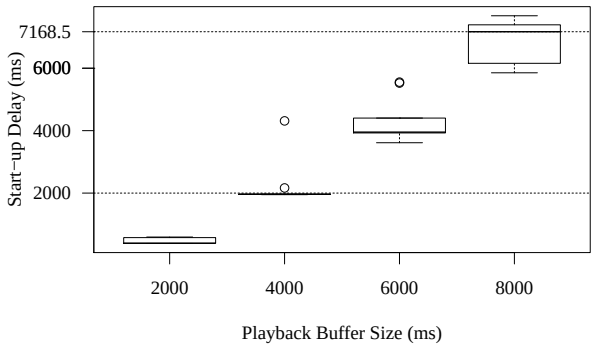

Figure 21: MV: 150 ms Link Delay

latency and throughput.

Hasan, Huang and Werstein [10] studied the effect of the transport protocol's send buffer size on the performance of streaming media and proposed a dynamic send buffer tuning approach (DBAT) which correspondingly provides congestion feedback to the application. Their main idea is to couple the application's control loop with transport protocol's control loop through cross layer information exchange.

Compared to [9] and [10], instead of proposing optimization techniques at the transport level, our goal is to perform an extensive experimental analysis to suggest the best suited playback buffer sizes to be used at the application level. This helps to make our approach generally applicable

Tan, Cui and Apostolopoulos [18] proposed to reduce the effect of a buffer underflow when there are multiple streaming sessions by redistributing resources. Their study puts a label on each packet of a streaming session corresponding to the buffer occupancy on the client. They then use scheduling so that packets in a session with labels smaller than the buffer size transmit sooner than others. Ordering at the resource bottleneck is based on labels carried in the packets. In this way, streaming sessions with a small playback buffer receive higher throughput, therefore fewer pauses. Again, this is approach requires a much more complex integration

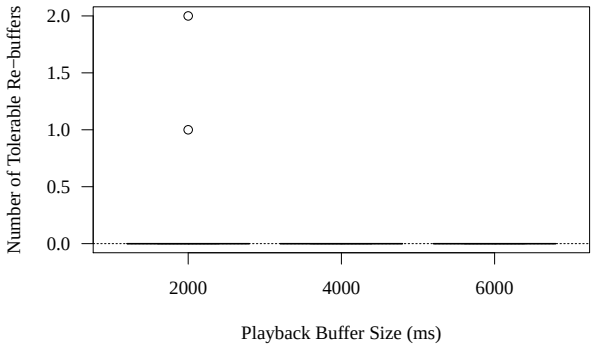

Figure 22: MV: 150 ms Link Delay

	150 ms	300 ms	0.1%	1.0%	2.5%
OBS_1(s)	10	14	4	4	4
OBS_2(s)	2	2	2	2	2

Table 3: Suggested optimal buffer sizes for the FMV

into the operating system than our approach.

Ho and Lee [11] suggested a predictive buffering algorithm for streaming video from multiple senders to a receiver over the best-effort Internet. They estimated the mean and variance of the aggregate throughput of multiple senders, and then used these estimated parameters to predict the future bandwidth availability. By appealing to the Central Limit Theorem, the future bandwidth availability will tend to be normally distributed, irrespective of the distribution of the measurement bandwidth availability. This insight enables their buffering algorithm to predict, at runtime, the buffering time required to ensure playback continuity. Our work is different from [11], since we consider the single sender and multiple receivers scenario.

6. CONCLUSIONS AND FUTURE WORK

Selecting an appropriate buffer size is essential if users are to have a smooth video streaming experience. Conservative choices that produce few interruptions reduce the responsiveness of streaming applications to users. Thus we want to be able to choose values as close as possible to the optimal buffer size. Within our experimental study, we define buffer optimality based on several metrics namely: start-up delay, the number of re-buffering events, and the re-buffering delay. Primarily we have considered three different video samples throughout this study. For all the tested video clips, with varying link delay and packet loss rates, we have suggested ideal playback buffer sizes that provide no visual interruptions. Furthermore our experimental results conclude that irrespective to the level of link delay or packet loss, a 2 second playback buffer is very likely to provide an optimal start-up delay for the spectrum of tested videos, based on our chosen model of user broadband. It should be noted that this value of 2 seconds is only a representative figure from the step-size of the playback buffers that we tested within the delay and loss parameter space. Since the step-size is rel-

	150 ms	300 ms	0.1%	1.0%	2.5%
OBS_1(s)	6	10	2	2	2
OBS_2(s)	2	2	2	2	2

Table 4: Suggested optimal buffer sizes for the SMV

	150 ms	2.5%
OBS_1(s)	8	2
OBS_2(s)	2	2

Table 5: Suggested optimal buffer sizes for the MV

atively coarse-grained, our suggested values for OBS_2 and OBS_1 may have imprecision in the order of one second. Using the same methodology but with a finer-grained step-size would increase the precision of the results.

However, a key and valuable conclusion from our study is that, for the variety of tested videos and network environments, that it is reasonable for OBS_2 to be kept constant, in contrast to the way that OBS_1 changes in response to delay and loss variations.

Additionally we emphasize that buffer sizes recommended in this study are likely to change based on the specific codecs used and the TCP congestion control mechanism employed by the base operating system. Nonetheless, we have used specific choices that are commonplace on the internet today.

As a future work, we intend to gather more traces and investigate how the optimal buffer sizes change when using different codecs and for different resolutions of video. We are currently working on an adaptive algorithm that can change the playback buffer size dynamically based on the predicted network throughput and client-side needs.

7. REFERENCES

[1] The Dummynet project. http://info.iet.unipi.it/~luigi/dummynet/. Accessed: 2015-09-05.

[2] FFmpeg. https://www.ffmpeg.org/. Accessed: 2015-12-11.

[3] Most viewed videos of all time (over 100 million views). http://trace.eas.asu.edu/yuv/index.html. Accessed: 2015-09-04.

[4] The next big thing in video: Adaptive Bitrate Streaming. http://web.archive.org/web/20100619225207/http://pro.gigaom.com/2009/06/how-to-deliver-as-much-video-as-users-can-take/. Accessed: 2015-09-02.

[5] Tutorial on Internet monitoring and PingER at SLAC. http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#loss. Accessed: 2015-09-08.

[6] VLC media player. http://www.videolan.org/vlc/. Accessed: 2015-09-10.

[7] YUV video sequences. https://www.youtube.com/playlist?list=PLirAqAtl_h2r5g8xGajEwdXd3x1sZh8hC. Accessed: 2015-09-02.

[8] S. Akshabi, A. Begen, and C. Dovrolis. An experimental evaluation of rate adaptation algorithms in adaptive streaming over HTTP. In *Proceedings of the second annual ACM Conference on Multimedia Systems*, pages 157–168. ACM, February 2011.

[9] C. Goel, A. Krasic and J. Walpole. Low-latency adaptive streaming over TCP. *ACM Transactions on Multimedia Computing, Communications and Applications*, 4(3), August 2008.

[10] S. Hasan, Z. Huang, and P. Werstein. Dynamic send buffer active tuning for low latency streaming media. In *Proceedings of the Australasian Telecommunication Networks and Applications Conference*, pages 140–145. IEEE, December 2007.

[11] P.Y. Ho and J.Y.B. Lee. Predictive buffering for multi-source video streaming over the Internet. In *Proceedings of Global Telecommunications Conference*, pages 1–6. IEEE, November 2006.

[12] Guo. Lei, Tan. Enhua, Chen. Songqing, Xiao. Zhen, Spatscheck. Oliver, and Zhang. Xiaodong. Delving into Internet streaming media delivery: A quality and resource utilization perspective. In *Proceedings of the 6th ACM SIGCOMM on Internet Measurement*, pages 217–230. ACM, October 2006.

[13] A.N. Murshed, A.F. Khalifeh, and M.A. Al-Taee. Quality of experience analysis of real-time video streaming over lossy networks. In *IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies*, pages 1–6. IEEE, December 2013.

[14] Nielsen. The total audience report december 2014. December 2014.

[15] O. Nukhet and T. Turhan. On optimal receiver buffer size in adaptive Internet video streaming. *Journal of Zhejiang University SCIENCE A*, 7(1):112–118, January 2006.

[16] Calyam P., M. Sridharan, W. Mandrawa, and P. Schopis. *Performance Measurement and Analysis of H.323 Traffic*. Springer, Berlin Heidelberg, 2004.

[17] S. S. Shunmuga Krishnan and R.K. Sitaraman. Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs. In *Proceedings of the 2012 ACM Conference on Internet Measurement*, pages 211–224. ACM, November 2012.

[18] W. Tan, W. Cui, and J.G. Apostolopoulos. An experimental evaluation of rate adaptation algorithms in adaptive streaming over HTTP. 2009.

[19] A. Zambelli. IIS Smooth Streaming technical overview. March 2009.

[20] Statistics New Zealand. Internet Service Provider Survey: 2014. October 2014.