LiveAdapp: Live adaptive mobile video streaming during large scale events

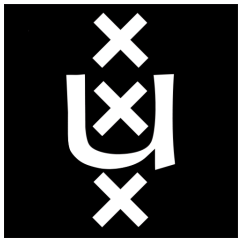
SUBMITTED IN PARTIAL FULLFILLMENT FOR THE DEGREE OF MASTER OF SCIENCE

Suzanne van der Tweel
10253025

MASTER INFORMATION STUDIES
HUMAN-CENTERED MULTIMEDIA

FACULTY OF SCIENCE
UNIVERSITY OF AMSTERDAM

July 14, 2016



1st Supervisor
Sergio Cabrero
CWI

2nd Supervisor
Dr. Frank Nack
ISLA, UvA

LiveAdapp: Live adaptive mobile video streaming during large scale events

S. van der Tweel
s.van.der.tweel@cwi.nl
CWI: Distributed and Interactive Systems
Universiteit van Amsterdam

ABSTRACT

Live uploading video streams are becoming increasingly popular with the recent release of Facebook Live and Periscope. Whereas adaptive live streaming is the standard for video delivery, this technology is not used in sending live video streams from smartphones. Current live streaming apps provide one video quality and are not able to provide stable video streams in bad networking conditions, when many people are streaming at the same time in the same network. Many problems will occur, when all users are competing for available bandwidth. In this paper LiveAdapp is presented which is able to adapt video quality during the capture of a live video stream from smartphones. Several experiments are performed with different adaptation algorithms in a testbed network, to simulate network conditions in large scale events. By the use of a Smart Network, bandwidth is divided over the number of users in the network. This technology together with a mobile live streaming application which is able to adapt the video quality during live video stream, will provide a stable video stream with the best possible video quality. Besides, bandwidth allocation has an important role regarding problems of competition for available bandwidths among users in the same network. Although, the experiments are not tested in real large audience environments, the results of the simulated networking conditions are promising for future developments.

Keywords

Live video streaming, HTTP Live Streaming, large scale events, networking issues, SDN, adaptive live streaming

1. INTRODUCTION

Recently, a trend of user generated content have emerged by the Internet, such as photo and video sharing (Engström, Esbjörnsson, & Juhlin, 2008). According to Cisco Visual Networking Index, the mobile network data traffic has grown with 74 percent in 2015. Whereas global mobile data traffic reached 3.7 exabytes per month at the end of 2015 [1]. Facebook has a 8 billion average of daily videos from 500 million of users, which indicates the need of sharing video with other people around the world [2].

Recent developments in the field of live video streaming from smartphone devices have led to a renewed interest in smartphone applications which support this technology. In 2010, Juhlin, Engström, and Reponen (2010) defined mobile broadcasting as a new type of social medium that allows users to capture video live from their phones and stream it immediately to the internet. With the increasing hype of apps like Periscope, Meerkat and the recent release of Facebook Live, live video streaming is becoming increasingly popular [3]. Periscope users can share live video streams with their friends or with all people in the world. The main focus is people discovering the world through someone else's eyes [4].

Like all social media, this new video sharing technology is quite interesting for large scale events. For example DJ's were using this technology for a while to live interact with the audience during their performance (Shamma, Churchill, Bobb, & Fukuda, 2009). Currently live video streaming is available for everyone from everywhere, which makes it more interesting to investigate. A completely new experience is created in the use of smartphones and live video uploading and directly sharing during large scale events using the smartphone as a medium.

However, large scale events have several exceptional characteristics. Networking issues are one of the most frequently stated problems within wireless networks at large audience environments. Many people are using the same wireless network and a lot of traffic is passing through that network simultaneously. Gupta, Min, and Rhee (2012) outline the extremely poor performance of WiFi hotspots in busy airports or large conventions. Erman and Ramakrishnan (2013) investigated the LTE traffic at the Super Bowl, where almost 75.000 attendees assemble for several hours. They noted that down streaming video traffic accounted for 19.6% of the overall traffic during the event, which is a large share of the whole capacity. By these measurements they conducted that 72% of the video traffic was used by the delivery of the live video stream of the Super Bowl. The livestream was generated by an adaptive bitrate protocol, which could adapt to the available capacity of the network to the viewing request of the user. This means that the video stream was available in several bitrates and the users who wanted to access the stream received the best available quality with the current network conditions. Adaptive video streaming technologies, such as the proposed standard DASH protocol,

[1] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html
[2] http://techcrunch.com/2015/11/04/facebook-video-views/
[3] http://nos.nl/artikel/2092247-hoe-gaat-het-met-de-race-om-de-livestreams.html
[4] https://www.periscope.tv/about

are widely used for video delivery (Akhshabi, Anantakrishnan, Begen, & Dovrolis, 2012). However, this technology is used for delivering video and currently not for live uploading captured video streams.

These adaptive technologies are trying to provide the best possible video quality to the requesting user, but this also has some side effects. If all users in the same network are competing for the available bandwidth to receive the best video quality, undesired effects will appear. Akhshabi et al. (2012) denoted the problem when two or more adaptive streaming players are steaming in the same network. They share a network bottleneck when they are both competing for the available bandwidth. Akhshabi et al. (2012) defines three performance problems in the competition of bandwidth between adaptive video players in the same network. This competition may lead to player instability, unfairness between players and bandwidth underutilization. These problems appear in video downloading, but are also related to video uploading. When this technology is applied to live sending video files, it may lead to unfairness of the video qualities and bandwidth allocation for each user. One user could claim more bandwidth than other users in the same network. Another research performed by Akhshabi, Begen, and Dovrolis (2011), showed that when two adaptive video players are competing for bandwidth, the second player stays in the lowest possible bitrate, when the first player uses the remaining available bandwidth with the highest sustainable bitrate. This approach is obviously an unfair way of sharing bandwidth capacities. These problems are very important in considering adaptive technologies for live video streaming smartphone applications.

Existing live streaming smartphone applications are providing one video quality for uploading video in different networking conditions. By the general use of LTE networks, apps such as Periscope, Meerkat and Facebook Live have chosen for one stable video quality without any adaptation. On the contrary with the adaptive technologies used for video downloading, these applications are not providing the best available video quality with different networking conditions.

Problem statement.

Current live streaming smartphone apps are not designed to cope with different networking conditions. If many application instances are live streaming inside the same wireless network it will suffer from network overload. This may lead to bad image quality of the video stream or freezes due to bandwidth insufficiency. The users that want to upload live video at large scale events will be limited by the networking issues. Besides, users are only able to stream their live stream in one video quality. If there is enough bandwidth available, current applications will not provide uploading the best possible video quality. One of the main concerns is the stability of the video streams, due to the competition of individual adaptive video technologies in the same network. Therefore two general problems regarding live video uploading from smartphone devices can be denoted. First, the best possible video quality is currently not provided to users. Second, due to the competition of individual users in the same network, stable video uploading streams are required. Therefore the following research question is defined:

How can a smartphone live streaming application provide the best possible video quality and stable video in different networking conditions at large scale events?

In this study a research will be done to contribute to solving these problems and to investigate network issues regarding live streaming from your smartphone on large scale events. The performance of different existing live video streaming technologies and protocols will be investigated in these situations. Additionally a smartphone application will be designed based on these technologies in order to provide the best possible video quality and stable video stream with the competition of resources taken into account. This application will be compared with other existing live streaming applications.

2. BACKGROUND

In this section, previous studies about networking issues regarding live video uploading from smartphones are investigated. Current live streaming and adaptive technologies are presented together with the state of the art of existing live streaming apps. Besides, the role of video quality and quality of experience is discussed. First it is important to understand the characteristics of large audience environments. Based on this literature review, several experiments are performed to simulated these networking conditions and are presented in section 3.

2.1 Properties of large scale events

Large scale events provide interesting characteristics, such as the enormous distribution of people, duration of the event that extends over days and participants are set apart from their daily life (Jacucci & Salovaara, 2005). Jacucci and Salovaara (2005) showed in their field studies that mobile phones are enhancing the event's experience on site, rather than using it for communications or documentation. It can be used for maintaining relations to a social network and engagement in the event itself. These events are an unique situation when the large number of people in the same place creates an immense demand on the available resources of the wireless networks (Erman & Ramakrishnan, 2013).

In terms of application usage, there is an increased use of social networking during large events (Shafiq et al., 2013). It is critical for providers to cope with such high demands during crowded events. Shafiq et al. (2013) measured the performance of cellular networks during crowded events compared with routine days. In this research is observed that flow counts of social networking content publishes more than double on the event day as compared to the routine days. Where Juhlin et al. (2010) has defined live streaming from your smartphone as a new social medium, it may also lead to increased use of live streaming apps and the demand of live uploading video during large scale events.

2.2 Live streaming protocols

One of the technologies used for live streaming is capturing live video from a smartphone device and broadcasts it to an external video player on a web page in real time. Live streaming technologies consist for example of sending small pieces of video continuously to a server. The server stores the segments and generates the manifest. A video player, downloads the segments according to the manifest, so the user can watch the video. This approach is used by HTTP Live streaming (HLS) or Dynamic Adaptive Streaming over

HTTP (DASH). Other live streaming technologies use different methods, such as sending single video and audio frames. At the end of this section the state of the art of existing live streaming apps are presented and several experiments were performed to measure the performance of these applications.

2.3 Adaptive bitrate streaming

Adaptation algorithms are widely used for video streaming services like Netflix and Youtube. The video player chooses the best possible video quality with the available bitrate and network conditions taken into consideration. Huang, Johari, McKeown, Trunnell, and Watson (2014) define two main goals that streaming services want to achieve in this process. First they want to maximize the video quality by selecting the highest video rate the network can support. Second, they try to minimize the rebuffering events which will make the player stop, if the buffer goes empty. Besides, if the buffer of the video will run dry, freezes and rebuffering messages will appear (Huang et al., 2014). This will provide an unstable playback of the captured video.

Dynamic adaptive streaming over HTTP (DASH) is becoming the standard technology for video streaming over the Internet (ISO/IEC, 2014). DASH allows video players to adapt the video quality based on the current network conditions. A list of several representations of the video is presented and the DASH player chooses the one that is optimal for the current networking conditions and the device. The video is divided in different segments and based on how fast the current and previous segments are downloaded, the quality and bitrate of the next segment is chosen (Robinson, Jutras, & Craciun, 2012). By using these smaller segments of approximately 2-10 seconds of video an adaptive stream is created by providing a continuous video stream of segments in the maximum possible video quality. Apple has developed its own adaptive streaming protocol known as HTTP Live Streaming (HLS) (Robinson et al., 2012). This protocol is used for live streaming video to all Apple products such as iPhones, iPads and Macbooks. For the design of the smartphone application this protocol is very useful and will be explained furthermore in section 3.

Adaptive bitrate streaming is currently designed to adapt web player clients based on the network conditions. In the case of live uploading video streams from a smartphone on large scale events it would be an interesting finding to adapt the bitrate or resolution before or during capturing the video. The bitrate and quality is then related to the current networking conditions during the event and it would be easier to live stream video with many people at the same time.

2.4 The use of SDN's and programmable networks

Software Defined Networks are a new networking approach which allows more control for network managers. Besides, the needs of programmable networks in research are increasing (McKeown et al., 2008). Kleinrouweler, Cabrero, and Cesar (2016) have designed a programmable SDN. They evaluated how two different mechanisms could contribute to the delivery of high quality and stable streaming video. It shows that their architecture improves the quality of experience by doubling the video bitrate and reducing disturbing quality switches.

By making use of this SDN, live streaming apps could have prioritization on WiFi hotspots on large scale events. This could be a solution to low bandwidth problems and bad image quality and may even lead to provide stable live video streams uploaded live from smartphones. Besides, an application could retrieve networking information from the SDN and could be able to adjust operations based on this information. This smart network also uses a network controller to contain an overview of the internet traffic. With this information it is able to divide the available bandwidth equally over the users inside the network. This could also be a solution in providing stable video streams and reduce bandwidth competition among users.

2.5 Video Quality

Video quality is an important factor in live uploading video streams. If the quality of the video changes it is immediately visible and affects the users who are watching the live stream. Dobrian et al. (2011) denotes the negative impacts on the quality of experience by changing the quality too often caused by a lower resolution or bitrate. This instability could lower the overall quality of experience and reduces user engagement. Providing stable and high quality video would improve the user experience. In the research of De Pessemier, De Moor, Joseph, De Marez, and Martens (2013) a subjective experiment is performed where different measures of QoE were evaluated while watching six different technical scenarios of mobile video content. The fluidity of the image during video playback and the synchronization of image and sound during playback received the highest ratings in terms of importance of a good mobile video watching experience. Regarding the technical details of the experiment, De Pessemier et al. (2013) showed, the low bandwidth connections have longer loading times than medium or high bandwidth connections. Besides, the results obtained by using low bandwidth connections and high quality videos are insufficient for fluent playback of videos. This would have negative impacts on the overall user experience of live streaming applications.

These results suggest that in case of capturing videos during a large scale event, where bandwidth is scarce, these combinations of low connections and high quality should not be used in order to avoid long loading times at the viewers side. If there is few bandwidth available it is important that low quality video quality is captured. This could also benefit for the overall networking problems during large scale events and watching the live videos.

As explained in the background section, several experiments are performed with networks in large scale events but this needs more exploration. Research has proved that users want fluent playback of video without many interruptions and low rebuffering times. Besides, low bandwidth connections together with high quality videos will cause multiple problems in networks. These factors should be taken into account when creating a live video streaming smartphone application.

By using programmable networks, live streaming videos from your smartphone can be prioritized and could get stable bandwidth from networks. With many users inside the same network this might contribute to provide the best possible video quality and stable video. Additionally current streaming technologies use adaptive bitrate streaming to adapt to networking conditions. This technology could be very inter-

esting for capturing live video and immediately uploading them at large scale events. With the use of a smart network the video quality can be adapted before capturing the video by the use of a network controller which provides network information and equally divided bandwidth capacity. Al these factors taken into account might provide a stable video stream in the best possible video quality. In the next section an app prototype is designed and explained. An experiment is conducted to measure the performance of the app prototype compared with existing live streaming apps. Before explaining the implementation of the app prototype, the state of the art of existing live streaming apps are presented.

2.6 State-of-the-art of live video streaming apps

A lot of different protocols are used in mobile live streaming applications in case of video uploading and downloading. After some research in popular existing live streaming applications, most were using RTMP, followed by HLS and 1 was using WEBRTC as streaming protocol. These protocols are used for both uploading video and for the delivery of the videos. RTMP provides low latency between the broadcaster and the viewer and therefore remains popular for live streaming video [5]. Most live streaming applications use their own developed combination of protocols. Periscope, for example, uses RTMP for uploading of the video, because it wants to provide low latency. However, RTMP cannot support wide scale and therefore also uses HLS for video delivery in large scales [6]. Meerkat, on the contrary uses its own HTTP protocol.

This indicates the diversity among these live streaming applications. Popa, Ghodsi, and Stoica (2010) also showed the existence of massive HTTP infrastructures. With the growth of sending video traffic, HTTP delivery is increasingly used in the Internet traffic. Besides, more firewalls are allowing HTTP, and therefore HTTP Live Streaming is the most used live streaming protocol in general. More applications are switching to HTTP Live Streaming (HLS) for the delivery of video streams, because of the wider device capacity. RTMP is not supported by Apple and uses a push model for connection between server and player. In section 2.3 HLS will be explained as the protocol that is used for sending video to iOS devices.

Performance of existing live streaming apps.

Meerkat, Periscope and Facebook Live have the most users among the live streaming apps and are therefore stated as most popular. Therefore several performance tests were performed to measure the stability and video quality of these applications. The video latency between the sender of the live video and the receiver of the video is used to measure the stability of the video. This was done by using videoLat[7], which is a tool that was designed for measuring latency during live video conferences. VideoLat is a tool that measures glass-to-glass video delays by generating QR codes on a screen and then measuring how long it takes until that same QR code is detected by the connecting camera. In this

[5]https://www.quora.com/What-technology-and-or-thirdparty-libraries-do-Meerkat-Periscope-Snapchat-and-others-use-for-live-streaming-audio-and-video-from-mobile-devices

[6]http://nerds.airbnb.com/building-periscope-for-android/

[7]http://videolat.org/

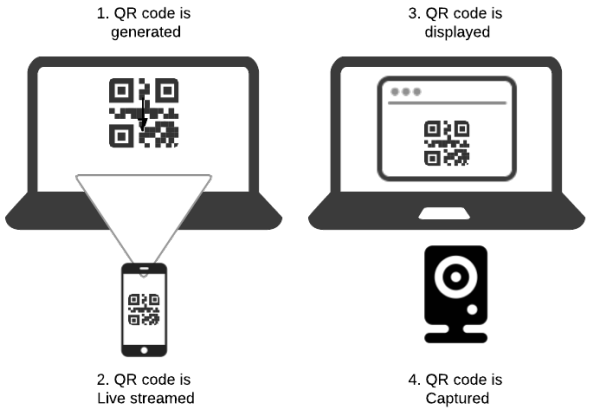


Figure 1: VideoLat experiment setup

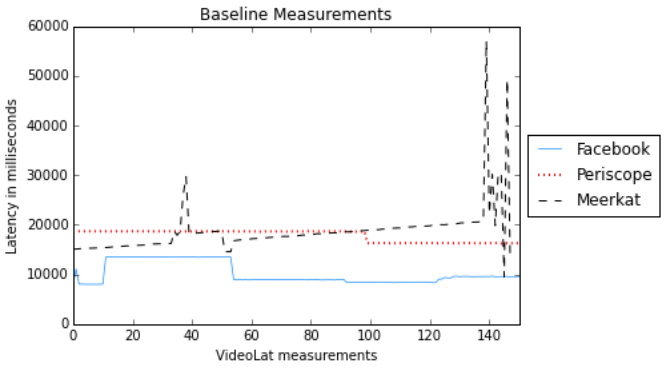


Figure 2: Baseline measurements for Facebook Live, Meerkat and Periscope

experiment a Logitech Carl Zeiss Tessar 1080 HD webcam was used for capturing the QR codes. VideoLat is currently only available for Mac OS and therefore in this experiment Apple Macbook Pros were used.

To measure the latency of the smartphone apps, the apps replaced the webcam and was pointed at the Macbook screen with the projected QR code. The webcam was now pointed to the live stream video player in a Safari web browser where the video is displayed live. The QR code was live streamed with the smartphone and the webcam captured the QR code in the browser. Once the QR code was captured, videoLat generated a new QR code and the latency was measured. The setup of this experiment is shown in figure 1. VideoLat is representative for 500-1000 samples in order to have a good data analysis. However due to time constrains in this paper a measurement of one hour is used. Besides, with this experiment the stability of the video stream was investigated and these conclusion could be drawn with less samples than originally stated for conferencing systems.

Baseline experiment.

The first experiment consisted of a baseline measurement. The network had 50 mbps bandwidth available and should provide the best possible video quality. The results of this experiment are presented in figure 2. During the 60 minutes experiments, around 150 samples were measured. Facebook

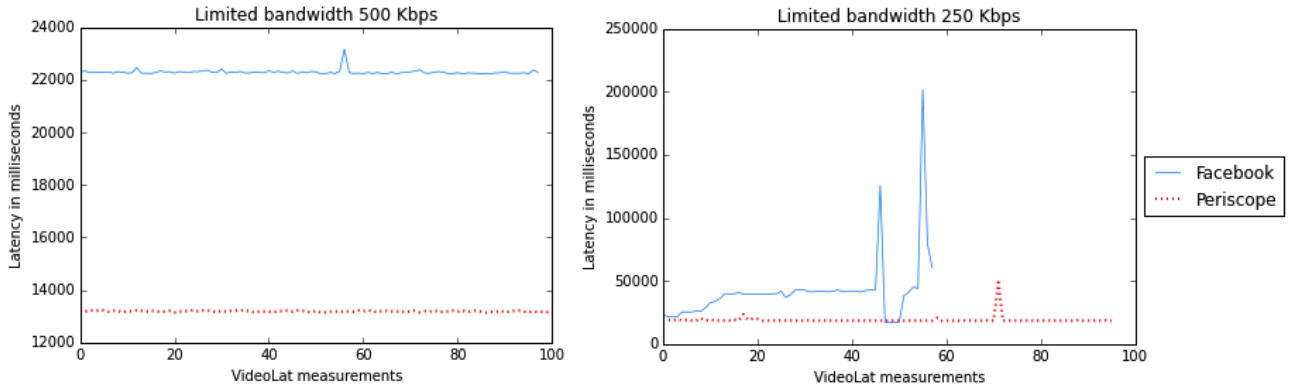


Figure 3: Limited bandwidth experiments of 500 and 250 kbps for Facebook Live and Periscope

Live and Periscope were able to provide a stable stream with small differences in latency during the live video stream. However, Meerkat showed to have an unstable stream during a live stream of 60 minutes. Starting from the first measurement, the latency is increasing each measurement and at the end of the experiment the latency had an enormous delay. Therefore Meerkat was omitted from further experiments, because there was no stable video stream measured with the best networking conditions and available bandwidth. Besides the latency the video quality was measured, by saving the video of the live stream to the smartphone. Facebook live captured with a video quality of 720x720 resolution and Periscope with 320x568 resolution. For Meerkat it was not possible to save the video stream to the smartphone and the video quality could not be determined.

Limited bandwidth experiment.

For Periscope and Facebook Live two other experiments were performed to measure the adaptation and the stability in different networking conditions. For this experiment a private Wi-Fi hotspot was setup for limiting the available bandwidth to 500, 250 and 100 kbps. The results of the 500 and 250 kbps experiments are shown in figure 3. Both apps were not able to start a live stream with a 100 kbps bandwidth. They both had a check before starting the stream, and if the the connection was too poor, the user was not able to go live. Again videoLat was used to measure the latency and 100 measurements for each app were used.

The experiment of 500 kbps shows a clearly stable live video stream for both Periscope and Facebook Live, where Periscope had the lowest latency. However, for the 250 kbps experiment Facebook Live was not able to provide a stable video stream. At the end of the experiment the video player was frozen and the experiment was stopped. Periscope on the other hand, still provided a very stable live video stream. Overall, these results indicate that Periscope is able to provide the most stable stream in different networking conditions.

3. METHOD

In previous studies several experiments are performed about networking problems at large scale events. However, Gupta et al. (2012) states that many solutions are not tested in real networks with realistic network workloads, most are based

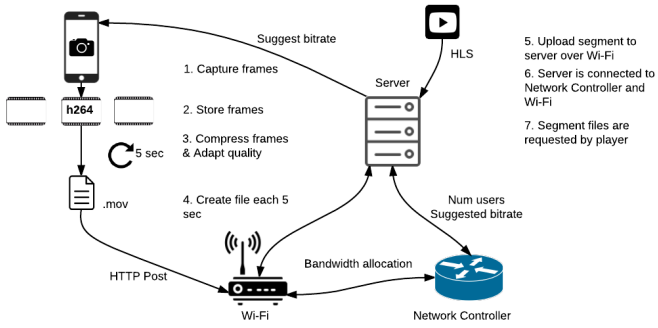


Figure 4: Flowchart of live streaming structure

on theoretical analysis. Therefore in this paper these network workloads are simulated by a network testbed to test the performance of the networks in reality. To simulate the networking conditions in large scale events, bandwidth limitations were applied to the network. With the research question in mind, an iOS app prototype called LiveAdapp, was created which could provide the best possible live video quality to the end user. Besides the live stream should be stable, without any frame drops or interruptions and the bandwidth should be fairly divided among users. Finally the delay from sender to receiver should be acceptable. By using several adaptations algorithms the best adaptation method for live streaming the best possible video quality and stable video stream was investigated.

3.1 The live streaming app prototype

Before explaining the algorithms, it is important how the live stream is implemented inside LiveAdapp. First of all the video frames are captured with for example an iPhone 6s individually. These frames are stored and after five seconds these frames are compressed to a .MOV file on the iPhone. All the captured video is compressed with the h264 codec. The adaptation part of LiveAdapp consisted of changing the quality settings of each five seconds segment file. Each time a new video file of five seconds is created these settings can be adapted. Different adaptation algorithms were used and are presented later in this section. When the compression of the file is finished, the file needs to be uploaded to a server. However, these files need to be uploaded in the right order

and it might happen that one previous file is still uploading. Therefore a waiting queue is created which stores the file temporally and uploads the five seconds segments one by one to the server. Once the file is finished uploading, it is removed from the queue. The file is uploaded with a HTTP Post API call through the Wi-Fi network to the server. The server stores all the segments and creates a manifest of these files. Once the HLS video player requests the manifest, the segments are downloaded by the player and shown to the viewer. Once the buffer size of the video player reached three segments, the video stream started playing. Between the Wi-Fi router and the server, a Smart Network Controller is present. In the case of using the Smart Network, the bandwidth capacity is divided over all users in the network. The server tells the Network Controller the number of users and the Network controller suggests a bitrate for each user. The Smart Network Controller allocates the bandwidth in the Wi-Fi router and the server is able to communicate with LiveAdapp. The server sends the suggested bitrate to LiveAdapp and LiveAdapp adapts the video quality of the next segment accordingly. The flowchart of this process is shown in figure 4.

The length of the segments is a design parameter that could be changed, but in this case was chosen for 5 seconds. This decision was made, because it provides lower latency between the sender and receiver, and makes the latency acceptable. Besides the files are smaller and would use less capacity on the server. The app prototype only saves the video frames and audio is not taken into consideration. There were compression problems with combining the audio and video stream in the implementation of the app, some tests showed that frames were dropped. For the experiments in this research, audio was not relevant for measuring the network performance and was therefore omitted. During the uploading process several performance metrics are measured for the adaptation algorithms that are explained later in this section. The file size, uploading time and queue size is saved for each segment.

There are many ways to upload live video streams as mentioned in the background section. For LiveAdapp HTTP was used for uploading the video segments to the server. Regarding the delivery of the video from the server to the video player in the Safari web browser, HLS was used. HLS generates small chunks of video in different video qualities and creates a playlist with these files. When the video player requests a video stream, the HLS protocol chooses the best video quality based on the current networking conditions and is able to adapt during the video playback. LiveAdapp already created video files of five seconds of video. By this implementation, the video is already cut in segments of five seconds. Therefore less processing was needed on the server. Although, HLS uses adaptive streaming for playing back video, only one video quality was used in the video player to simplify the implementation of the system. It used the maximum video quality it received. In the experiments was assumed that network bottlenecks are occurring in the uploading part, so the player will always download the highest available quality.

3.2 Adaptive streaming algorithms

By creating LiveAdapp two main goals for capturing live video want to be achieved. The best possible video quality is captured, based on the highest bitrate the network can support and a stable video stream should be provided for the end user. Additionally to provide a stable video stream, bandwidth competition should be minimized among different users. The results of existing live streaming apps in section 2.6 indicate that they cannot provide these goals. They use one standard video quality in all networking conditions. After performing several tests with Periscope and Facebook Live we have concluded that Periscope uses a 320x568 video quality and Facebook Live uses 720x720 video quality. Because Periscope provides the most stable video stream, the average bitrate was measured of approximately 338 kbps in a short test. These apps are not adaptive to networking conditions and they will not provide the best possible quality to the user. LiveAdapp uses five quality levels for adaptation and these are explained below.

Quality Level	**Resolution**	**Max Bitrate**
High	720p x 1280	2496000
Medium	480p x 848	1216000
LowMedium	360p x 640	864000
Low	240p x 424	576000
LowLow	144p x 256	256000

Table 1: Quality levels used for adaptation

Quality levels.

Research of De Pessemier et al. (2013) has proven that high quality video and low bandwidth will cause problems in the network. Therefore the app will only capture low quality videos when there is low bandwidth available, to avoid rebuffering in the player side. For the adaptation algorithms five different quality levels were used. These levels are presented in table 1, and are based on the quality levels used by Youtube. Five quality levels are chosen, because as explained in section 2.5 the fluidity of the image received the highest ratings. Besides every stream starts in low level video quality, because it minimizes the delay of the video to the end user. Miller, Quacchio, Gennari, and Wolisz (2012) also selected the lowest representation for the first segment, for minimizing delay for the users viewing request. Besides, the viewing experience is improved by providing the best possible video quality. The quality levels were chosen based on bitrate keys and up and downgrading between different quality levels.

Next, the five different adaptive algorithms will be described that were used in the live streaming app. These algorithms will be tested individually in order to find the best adaptive live streaming algorithm in fluctuating networking conditions. Before the user starts capturing the video he can choose from these algorithms. The algorithms are divided in two different network implementations. The Smart Network algorithm uses a Smart Network Controller and when this is enabled, the bandwidth is divided over the number of users. All the other algorithms use a network where the bandwidth is not allocated to users. Below the adaptation algorithms are explained in pseudo code.

Algorithm Queue.

if Queue size is > 1 **then**
  Downgrade video quality one level

end if
if Queue is empty 5 times in a row **then**
Upgrade video quality one level
end if

In this algorithm the uploading queue is used. When the size of the queue is becoming greater than one, the video quality of the next segment will be changed to one quality level lower. If the queue is empty for 5 times in a row, the video quality is upgraded to one level higher.

Algorithm Aggressive.

The aggressive algorithm uses the bitrate of each file as an adaptive measurement. For each segment the bitrate is calculated by diving the file size by the uploading time of the segment. Each quality level has an maximum bitrate and the video settings for the next segment is chosen based on the bitrate of the previous segment.

Algorithm non-Aggressive.

This algorithm uses exactly the same bitrate as the aggressive algorithm, but takes the average bitrate of the last 5 segments. The adaptation is again chosen by the maximum bitrate for each quality level.

Algorithm Combination.

if Queue size is > 1 **then**
Downgrade video quality one level
end if
if Queue is empty 5 times in a row **then**
Use average bitrate of last 5 segments for choosing quality level
end if

The combination algorithm uses both the uploading queue and the average bitrate of the last 5 segments, just like the non-aggressive and queue algorithm. If the queue is becoming greater than one, the video quality is downgraded one quality level. If the queue is empty for five segments in a row, the video quality is chosen based on the average bitrate.

Smart Network.

The Smart Network algorithm has an extra feature compared with the other adaptation algorithms. This algorithm uses a Smart Network Controller which is able to divide bandwidth capacity over the number of users in the network. The Smart Network algorithm continuously communicates with the server and the server sends the suggested bitrate it should use for the video settings. For example, if there is 2 mbps available and there are 2 users in the network, each user receives exactly 1 mbps. The application retrieves this bitrate from the server and can choose the best possible video quality that is available within this bitrate. If the number of users in the network increases, each user gets less bitrate available and the app is able to adapt accordingly.

3.3 Experiment

In this paper several experiments were performed to test the performance of an adaptive live streaming application with different simulated networking conditions and adaptation algorithms. This application was compared with the existing non adaptive applications Periscope and Facebook Live. To simulate networking conditions in large scale events, the bandwidth of the network was limited. Gupta et al. (2012) stated that many solutions about networking problems are not tested in real networks with realistic workloads. With the limitation of the network, the best situation was created to predict the performance of the apps in realistic network workloads. Besides, the application was tested in a Smart Network provided by CWI Distributed and Interactive Systems group located in Amsterdam. LiveAdapp was designed to cope with this Smart Network and without like explained in section 3.1. The experiment consisted of three parts, the first part was the measurement of all the individual algorithms, the second a multiple users experiment and third a bandwidth competition experiment. The experiments covered the following comparisons:

- Non adaptive live streaming vs adaptive live streaming
- No suggested bitrate vs suggested bitrate by Smart Network
- No bandwidth allocation vs bandwidth allocation provided by the Smart Network

3.3.1 Materials

The adaptive application that was developed for this experiment was named LiveAdapp. This app is an iOS application created in Xcode, and was originally designed for the iPhone 6s. The application was adjusted for different devices in order to perform the multiple users test. For this test an iPad mini, iPad, iPhone 5c, iPhone 6s and iPhone 6 were used. Besides, this also reflects the current state of the art of smartphone devices, where a lot of different devices are widely used in society. Additionally 5 webcams and 5 Macbook Pro's were used for the multiple users experiment by using videoLat.

Smart Network.

The project was supported by CWI Distributed and Interactive systems group, located in Amsterdam. Kleinrouweler et al. (2016) have developed a smart programmable network which was supported by LiveAdapp. As previously explained this Smart Network can divide the bandwidth usage over users inside the network. At CWI a private testbed was created were the Smart Network was enabled if needed. If the Smart Network was enabled it assigned an equal bandwidth to each user in the network and the app requested a suggested bitrate through an API to the server. LiveAdapp adapted to this information accordingly. The same private network was used, when the Smart Network was disabled. For simulating the network conditions this network was limited to several bandwidths, which are illustrated in section 4.2.

Server.

Besides the smart network, CWI provided an server that was used internally. The video segments that were produced by LiveAdapp were all uploaded to this server. The server was creating data with log files of the video segments in order to analyze them later. The uploading time, video bitrate and video quality heights and widths were saved for each stream.

Smartphone apps.

During the experiments several smartphone applications were used and were all tested individually. Besides the different adaptation algorithms, a non adaptive version of

LiveAdapp was created. Whereas the existing apps are non adaptive, an equal comparison was needed to rate the performance of the algorithms. Periscope, Meerkat and Facebook Live all use different protocols and algorithms that are not available open source. Therefore the non adaptive version of LiveAdapp was added to have a representative conclusion. The smartphone apps that were used were Facebook Live, Meerkat, Periscope, LiveAdapp Non Adaptive, LiveAdapp Queue Algorithm, LiveAdapp Aggressive Algorithm, LiveAdapp Non Aggressive Algorithm, LiveAdapp Combination Algorithm and LiveAdapp Smart Network Algorithm.

3.3.2 Procedure

The procedure of the experiments is divided into four sections. First the baseline for LiveAdapp was measured, second the performance of the different algorithms was conducted, third a multiple users test was performed and finally a bandwidth competition experiment was conducted. For all experiments the server logs were saved of each user of LiveAdapp.

Baseline.

Besides the baseline measurements in the background section, LiveAdapp was also measured with videoLat with 50 mbps bandwidth capacity. Therefore the LiveAdapp settings were maximized and set to the 720p video quality level. For this experiment LiveAdapp without adaptation was used.

Algorithms test.

The applications supported five different adaptive streaming algorithms and all of these were tested individually. The adaptation algorithms were tested with a network testbed, which was able to cut off the bandwidth of the network. De Pessemier et al. (2013) also used this technique in their research by limiting the bandwidth of the Wi-Fi connection to simulate different networking conditions.

The different bandwidth capacities that were used in the algorithms tested were 50 mbps, 2 mbps, 1 mbps, 750 kbps, 500 kbps, 250 kbps and 100 kbps. The LiveAdapp app versions were all tested with these capacities. Periscope and Facebook Live were only tested for 50 mbps, 500 kbps, 250 kbps and 100 kbps bandwidth capacity. These applications were not adaptive and would perform the same results in the other bandwidth levels and those were not taken into consideration. These capacities were chosen to check the performance of the adaptation of the algorithms. Besides, these capacities were interesting together with the five video quality levels. For example, 2 mbps will not be enough to provide a high quality stream because the high quality level requires 2.49 mbps or higher. Therefore it is interesting to see the behavior of the algorithms in this different bandwidth capacities.

Multiple users test.

For the multiple users test, 5 Macbook Pros were used with videoLat to measure the latency of the video stream. Periscope was denoted as the most stable existing live streaming application, from the experiments in the background section. Therefore Periscope, the Smart Network Algorithm and the Aggressive Algorithm were tested with multiple users. To make a good comparison between LiveAdapp with the Smart Network enabled and without, the Aggressive Algorithm was also included. The Aggressive algorithm had the most similar implementation compared to the Smart Network algorithm and in section 4.3 will be explained why the Aggressive algorithm was used for this experiment based on the results of the algorithms performance experiment.

For Periscope, videoLat was used to measure the stability of the video. The other two apps could provide more information with server logs and after several videoLat measurements the same conclusions could be drawn between the latency from videoLat and the uploading times from the server logs. VideoLat is a very time consuming measurement and therefore there was decided to only include the videoLat results for Periscope. The multiple users experiment started with one user and each 5-10 minutes another user joined the network. For the LiveAdapp versions, the timestamp of these users was available in the server logs, but for Periscope this information was not provided by videoLat.

Bandwidth competition.

After the five users test, another three users test was performed to measure the competitions in bandwidth between the adaptive apps. This was only done for the Smart Network Algorithm and the Aggressive Algorithm. These tests were performed with an iPhone 6s, iPhone 5c and iPad Mini. The total available bandwidth was set to 4 mbps.

4. RESULTS & DISCUSSION

Four different experiments were performed and the results of these experiments are also divided into different sections. The results of the baseline and algorithms experiment are presented, followed by the multiple users experiment and the bandwidth competitions experiment.

4.1 Baseline experiment

The baseline experiment of LiveAdapp had an average delay of 12 seconds and a standard deviation of 497 milliseconds. This indicates that the non adaptive LiveAdapp version provides a stable and 720p video quality in the best networking conditions.

4.2 Algorithms experiment

For each individual adaptation algorithm a limited bandwidth performance was measured with one user in the network. The goal of this experiment was to show that the algorithms provided the best possible video quality in different networking situations. The stability of the video is expressed in number of video quality switches. In figure 7 in the appendix the video qualities are presented of each limited bandwidth capacity experiment.

The 50 mbps bandwidth and 100 kbps experiment provided the least number of quality switches in all algorithms. All algorithms eventually streamed in the 720p quality video level or in 144p quality. For 50 mbps bandwidth, all algorithms were not able to go higher and for 100 kbps the algorithms were not able to go lower and therefore all stayed in the same quality level during the whole experiment. For most of the algorithms just one quality switch was needed during all bandwidth situations, except for the Queue algorithms. The Queue algorithm is always trying to upgrade when five segments are uploaded in a row and this is also visible in the results. This algorithm had a maximum of 10 switches during the 2 mbps experiment, where the other

algorithms just needed one switch. The most interesting bandwidth was 100 kbps which simulated the worst networking condition, where all the algorithms chose the 144p quality level. The quality switches already showed that all algorithms needed just one switch and proved the stability of the video. However, as shown in table 1, the 144p quality level uses 256 kbps. This means that the encoder produces lower video bitrates, which make streaming in lower networking situations possible.

4.3 Multiple users experiment

For the multiple users experiment three different apps were tested. Periscope, LiveAdapp Aggressive Algorithm and LiveAdapp Smart Network. During the algorithms experiment the Aggressive algorithm produced the most stable video bitrate and uploading times. Besides, the implementation of the algorithm was best comparable with the Smart Network algorithm. Both use one bitrate key for each segment for determining the quality level, one by retrieving the suggested bitrate by the server and the other by calculating it by itself.

Periscope.

The baseline measurement of Periscope indicated a very stable stream for 50 mbps bandwidth capacity with one user. However, the results of this experiment where multiple users stream at the same time are showing something completely different. The latency is fluctuating a lot and for the last users latency is quickly becoming high. For the 1 mbps bandwidth experiment the first two users had a high latency at the end of the experiment, when all four users were streaming at the same time. For 1 mbps bandwidth the fifth user was not able to start streaming. Periscope checked the available bandwidth at the beginning of each stream and therefore the fifth user was not able to stream. Both players had many freezes and caused a extremely high latency for user 1 and 2. User 1 had an average latency of 17 seconds and when the fourth user joined the network, the latency increased to a 200 seconds maximum delay. Therefore these results suggest that Periscope is not able to provide a stable live video stream when multiple users are using the network at the same time. Besides all the streams had one video quality in 320 x 568 resolution and a bitrate of 338 kbps. However, Periscope is providing audio and video in their app, which also leads to higher bandwidth usage. This could be an explanation why only four users were able to stream at the same time.

LiveAdapp.

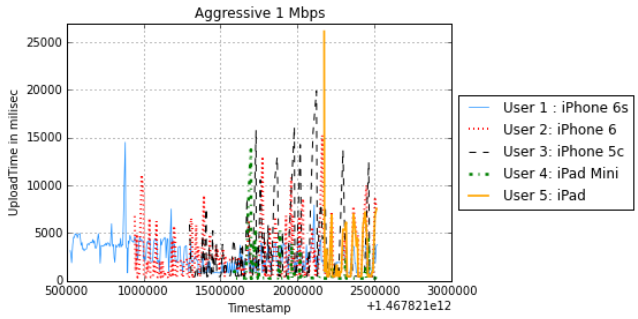
Both experiments for LiveAdapp were measured with the server logs, because these provided more information than videoLat. For both algorithms, 50 mbps bandwidth and 1 mbps bandwidth limitation for five users was performed. During the experiments the video qualities were tracked in order to define quality changes in the video stream. The 50 mbps bandwidth experiments had the same results for both algorithms with 5 changes in video quality. Each user that joined the network, immediately upgraded to high (720p) video quality and kept streaming in this quality during the whole experiment. However, for the 1 mbps experiments the algorithms provided completely different results. The Aggressive algorithm produced a great variety in video qualities during the experiment. Quality changes appeared 308 times for the Aggressive algorithm, where the Smart Network only had 6 video quality changes. The first users required 2 quality switches, first to change the quality level to the available bandwidth at the beginning of the experiment (from 240p to 360p) and second when the next user joined the network (from 360p to 144p). Each time a new user joined a quality switch was immediately changed to 144p quality. A graph of these quality changes is included in the appendix in figure 8. The video qualities are presented together with the timestamp of when the user joined the network.

The Aggressive algorithm calculated the video bitrate by dividing the file size of the segments by the uploading time of the segment and chose the corresponding video quality. Where the Aggressive algorithm needed 308 quality switches in 1 mbps bandwidth, the Smart Network only needed 6 quality switches. These results clearly indicate that the Smart Network algorithm provides a completely stable video quality in the 144p resolution. The Smart Network algorithm closely collaborated with the server to get the available bitrate and changed the video quality accordingly. Additionally the Smart Network equally divided the bandwidth over the users in the network.

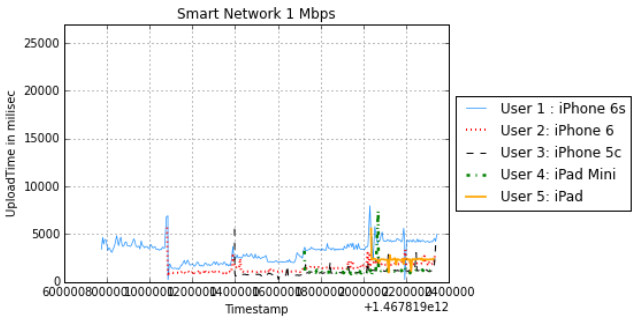
With these results in mind it is very interesting to look into the results of the uploading times of both algorithms, presented in figure 5. The Smart Network algorithm produced much faster uploading times than the Aggressive algorithm when more than two users were streaming at the same time. Until the second user started streaming at the same time, both algorithms provided a general stable uploading time. When the second user started streaming, the Aggressive algorithm produced a lot of diverse uploading times, where the uploading times of the Smart Network were not influenced by the next users. When the fifth user started streaming, the uploading times of the Smart Network algorithm increased for a few segments, but were again stable and all users stayed below 6 seconds of uploading. The Aggressive algorithm on the contrary, had a uploading time of 6 seconds and higher already for user 2 and 3 starting from two users in the same network. Once the fifth user joined the network, user 2, 3 and 5 had a lot of diverse uploading times. Only user 1 and 4 had uploading time below 6 seconds, when five users were streaming at the same time. The results of the Aggressive algorithm clearly prove the earlier stated problems of bandwidth competition of multiple users streaming in the same network at the same time. This algorithm did not use bandwidth allocation among the users, where the Smart Network algorithm did. Therefore after conducting these results, another experiment was performed to measure the bandwidth competition with multiple users.

4.4 Bandwidth competition

The final experiment consisted of a three user experiment and a limited network of 4 mbps. Both the Smart Network algorithm and the Aggressive algorithm were tested. The previous experiment showed a big difference in uploading times between the Smart Network algorithm and the Aggressive algorithm. The Smart Network algorithm uses bandwidth allocation for each user, which means the total available bandwidth is divided equally over each user in the network. To test the role of this bandwidth allocation (the Smart Network Controller), the Aggressive algorithm



(a) Uploading time results Aggressive Algorithm



(b) Uploading time results Smart Network Algorithm

Figure 5: Multiple users experiments for Aggressive and Smart Network algorithm with 1 mbps bandwidth limitation

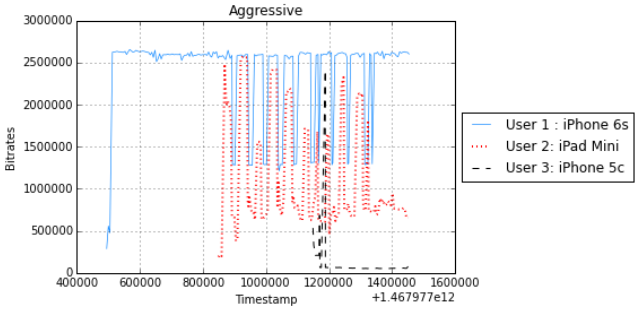


Figure 6: Video bitrate results for bandwidth competition experiments for Aggressive algorithm with 4 mbps bandwidth limitation

was also tested with this allocation enabled. Besides, the Smart Network algorithm retrieves a suggested bitrate from the server, which the aggressive algorithm does not possess. The goal of this experiment was to see how the different users in the network are competing for the available bandwidth which is a huge problem in networks with multiple users. Additionally, the role of the suggested bitrate by the server was investigated. In this experiment three different situations were tested. The Aggressive algorithm without bandwidth allocation, Aggressive algorithm with bandwidth allocation and the Smart Network algorithm with bandwidth allocation.

The Smart Network algorithm and the Aggressive algorithm with allocation enabled produced almost the same results in video bitrates in the bandwidth competition experiment. The first users in all three situations used a video bitrate of about 2.6 mbps with 4 mbps available for one user. When the second user joined the network, the Smart Network and Aggressive with allocation both used around 1.25 mbps for each user. For the last user a similar bitrate was measured, around 1 mbps. With these results, the difference of the suggested bitrate by the server could not be determined because both algorithms produced almost the same results. To measure this difference further experiments are required. The video bitrates for these two algorithms with bandwidth allocation are presented in figure 9 in the appendix.

The Aggressive algorithm without allocation, produced a completely variety in video bitrates. The video bitrates that were produced by the Aggressive algorithm without bandwidth allocation are presented in figure 6. When the third user joined, the Aggressive algorithm produced a very low video bitrate for the third user. The video quality results also show that when three users were streaming, the first user had 720p quality video, the second 480p quality video and the third user had 144p video quality. This indicates that the competition for bandwidth was very high with the Aggressive algorithm without bandwidth allocation and was claimed by one user streaming in high (720p) quality. The other two situations with bandwidth allocation provided medium quality for all users when three users were streaming inside the same network at the same time. This clearly indicates that bandwidth allocation will solve the problem of individual users competing for bandwidths and is contributing to stable video streams with multiple users in the same network.

With limiting bandwidth capacities a simulation was created for the comparison of networks at large scale events. The applications were not tested in real large events where many users are using the same network at the same time. This could also lead to different unstable networking conditions. However, with limiting the bandwidths and using the network with different users at the same time, showed some interesting results. Due to time constrains the experiments were only performed one time and should also be repeated to show better results. The videoLat experiments could provide more reliable results when the experiment had more sample measurements. Finally, the performance of the apps were not tested when the bandwidth was fluctuating from for example low bandwidth available to high bandwidth available.

5. CONCLUSION

In this paper we investigated how a smartphone live streaming application could provide the best possible video quality and stable video in different networking conditions at large scale events. These networking conditions were simulated by limiting the bandwidth of the network. A smartphone iOS application was developed which was able to adapt the video quality in different networking conditions. Unlike existing apps like Periscope, Meerkat and Facebook Live, which provided one video quality for their live streams. Besides, these applications were not able to provide a stable video stream,

in low bandwidth situations and multiple users streaming in the same network.

The results of the different algorithms experiments have also shown that LiveAdapp will provide the best possible video quality in different network conditions. The Smart Network algorithm, which includes the Smart Network Controller with bandwidth allocation, has proven to provide the best possible video quality and stable video in different networking conditions with multiple users. In section 2.5 it was explained that less quality fluctuations are preferred in playing back video to satisfy the user. The Smart Network algorithm will always tell the app in which quality it is able to stream in the current networking conditions and will therefore have a stable video quality. Additionally the Smart Network algorithm only had 6 quality changes in the five user experiment with a limited bandwidth of 1 mbps. Where on the contrary the Aggressive algorithm without bandwidth allocation, which calculates the available bitrate by itself, had 308 quality changes in the same experiment. This means the same problems arise with live adaptive streaming and competitions for available bandwidth, just like in video downloading and the current adaptive streaming standards. However, the results of the bandwidth competition experiment shows the important role of the Smart Network Controller which provides equally divided bitrates among the users in the network. The Smart Network algorithm on the other hand, has more reliable information with the use of the suggested bitrate from the server. The difference of this suggested bitrate has not been proven, but is an interesting field for further exploration.

Besides, Akhshabi et al. (2011) illustrated the role of unfairness in sharing available bandwidths with adaptive video players. Both the Aggressive algorithm and Smart Network algorithm provide equally divided video qualities over multiple users, when bandwidth allocation was enabled. However, the Smart Network algorithm collaborates more closely with the whole network, because it communicates with server continuously. The server will have a complete overview of the whole network and tells the Smart Network algorithm which video quality settings it should use. When more users are involved the Smart Network algorithm will be able to easily adapt to these network changes. Although these experiments were conducted in controlled lab experiments, they do provide interesting results for further developments. The network conditions that were simulated where comparable with networking conditions in large scale events, but networks in real event might provide less stability.

With the use of an adaptive algorithm that works closely with a Smart Network that is able to divide the available bandwidth over users in the network, a promising combination is found in the use of live mobile video streaming in large scale events. This Smart Network has proven to support a stable video stream and is reducing the problem of bandwidth competition between users. This will also improve the user experience and user engagement.

6. FUTURE WORK

Some interesting results are investigated, but do need further exploration in real large events. In controlled experiments the Smart Network algorithm together with the bandwidth allocation inside the network has proven to deliver the best possible video quality the available bandwidth allows and provides stable video streams. In future work, the application could be further developed to include audio and video. Besides, the multiple users experiments were tested for a few different bandwidth allocations and a minimum of segments were captured for each experiment due to time constrains. Additionally in this paper no difference was found for the role of bitrate suggestions from the server to LiveAdapp. This could also be investigated further when more tests are preformed with many users. The use of a Smart Network can also be useful for other scenarios in live video streaming in large scale events. For example, when many users are streaming at the same time and another user wants to join. This will make the available bandwidth for all the users to low to stream. The Smart Network could also then decide this user is not allowed to stream. Therefore the use of the Smart Network can be investigate further.

References

Akhshabi, S., Anantakrishnan, L., Begen, A. C., & Dovrolis, C. (2012). What happens when http adaptive streaming players compete for bandwidth? In *Proceedings of the 22nd international workshop on network and operating system support for digital audio and video* (pp. 9–14).

Akhshabi, S., Begen, A. C., & Dovrolis, C. (2011). An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proceedings of the second annual acm conference on multimedia systems* (pp. 157–168).

De Pessemier, T., De Moor, K., Joseph, W., De Marez, L., & Martens, L. (2013). Quantifying the influence of rebuffering interruptions on the user's quality of experience during mobile video watching. *Broadcasting, IEEE Transactions on*, *59*(1), 47–61.

Dobrian, F., Sekar, V., Awan, A., Stoica, I., Joseph, D., Ganjam, A., ... Zhang, H. (2011). Understanding the impact of video quality on user engagement. In *Acm sigcomm computer communication review* (Vol. 41, pp. 362–373).

Engström, A., Esbjörnsson, M., & Juhlin, O. (2008). Mobile collaborative live video mixing. In *Proceedings of the 10th international conference on human computer interaction with mobile devices and services* (pp. 157–166).

Erman, J., & Ramakrishnan, K. K. (2013). Understanding the super-sized traffic of the super bowl. In *Proceedings of the 2013 conference on internet measurement conference* (pp. 353–360).

Gupta, A., Min, J., & Rhee, I. (2012). Wifox: Scaling wifi performance for large audience environments. In *Proceedings of the 8th international conference on emerging networking experiments and technologies* (pp. 217–228).

Huang, T., Johari, R., McKeown, N., Trunnell, M., & Watson, M. (2014). A buffer-based approach to rate adaptation. In *Proc. 2014 acm conference on sigcomm, sigcomm* (Vol. 14, pp. 187–198).

ISO/IEC. (2014). Information technology- dynamic adaptive streaming over http (dash)-part 1: Media presentation description and segment formats. *ISO/IEC MPEG, Tech. Rep*.

Jacucci, G., & Salovaara, A. (2005). Mobile media sharing in large-scale events: beyond mms. *interactions*, *12*(6), 32–34.

Juhlin, O., Engström, A., & Reponen, E. (2010). Mobile broadcasting: the whats and hows of live video as a social medium. In *Proceedings of the 12th international conference on human computer interaction with mobile devices and services* (pp. 35–44).

Kleinrouweler, J. W. M., Cabrero, S., & Cesar, P. (2016). *Delivering stable high-quality video: An SDN architecture with DASH assisting network elements*.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., ... Turner, J. (2008). Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, *38*(2), 69–74.

Miller, K., Quacchio, E., Gennari, G., & Wolisz, A. (2012). Adaptation algorithm for adaptive streaming over http. In *Packet video workshop (pv), 2012 19th international* (pp. 173–178).

Popa, L., Ghodsi, A., & Stoica, I. (2010). Http as the narrow waist of the future internet. In *Proceedings of the 9th acm sigcomm workshop on hot topics in networks* (p. 6).

Robinson, D. C., Jutras, Y., & Craciun, V. (2012). Subjective video quality assessment of http adaptive streaming technologies. *Bell Labs Technical Journal*, *16*(4), 5–23.

Shafiq, M. Z., Ji, L., Liu, A. X., Pang, J., Venkataraman, S., & Wang, J. (2013). A first look at cellular network performance during crowded events. In *Acm sigmetrics performance evaluation review* (Vol. 41, pp. 17–28).

Shamma, D. A., Churchill, E. F., Bobb, N., & Fukuda, M. (2009). Spinning online: a case study of internet broadcasting by djs. In *Proceedings of the fourth international conference on communities and technologies* (pp. 175–184).

APPENDIX

Acknowledgments

I would like to thank the CWI Distributed and Interactive Systems group for offering great support during writing my thesis and with the development of the iOS application. I would like to offer my special thanks to Sergio Cabrero and Jan Willem Kleinrouweler who provided me guidance and valuable support during the project.

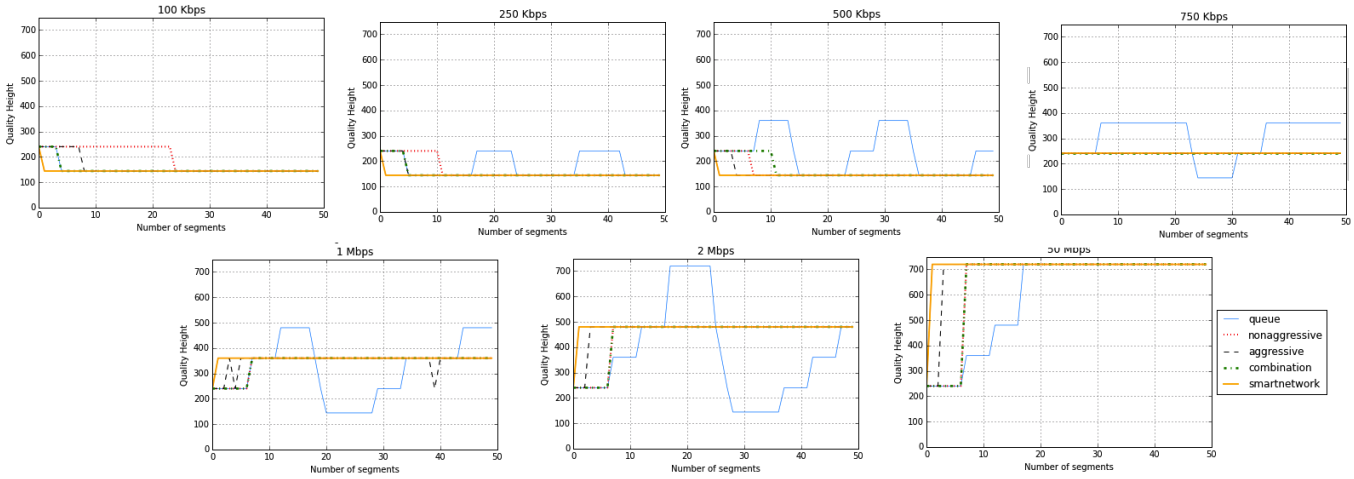
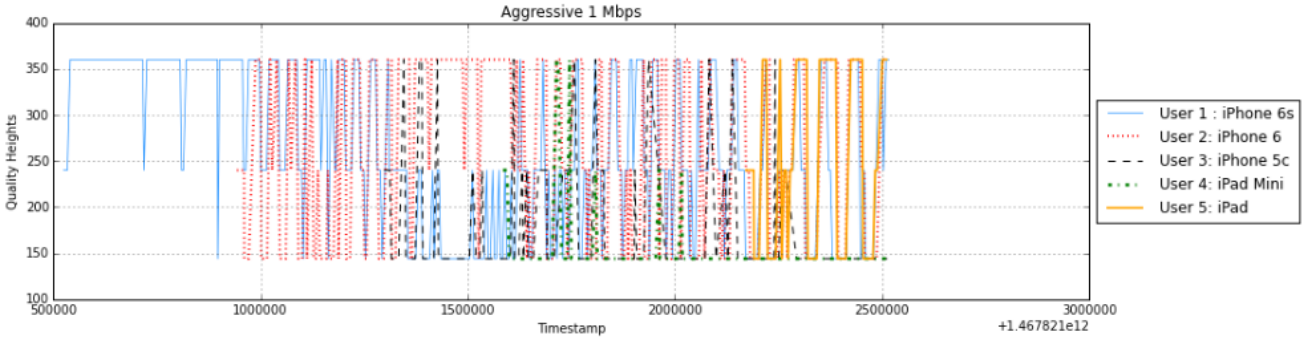
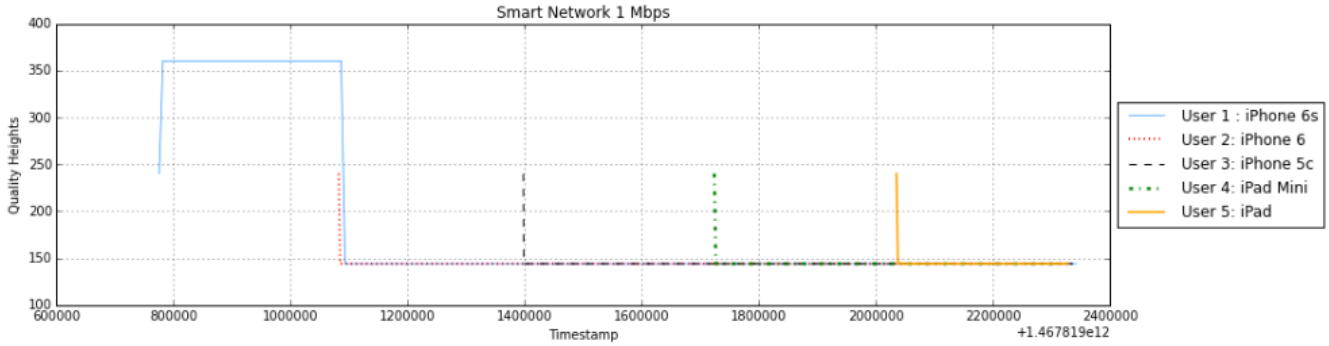


Figure 7: Video quality for each algorithm for limited bandwidths

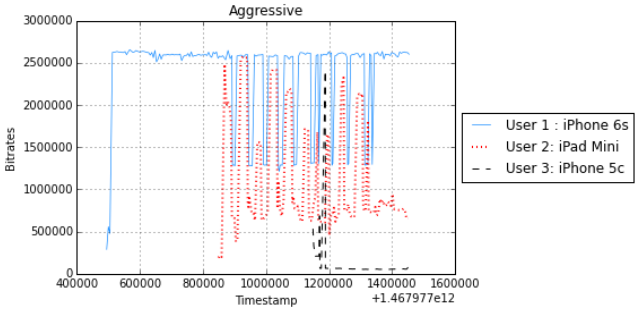


(a) Video quality results Aggressive Algorithm

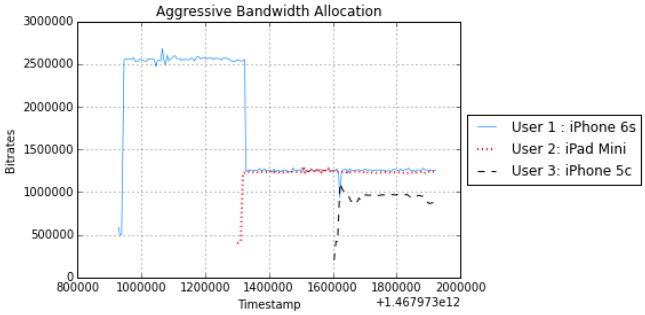


(b) Video quality results Smart Network Algorithm

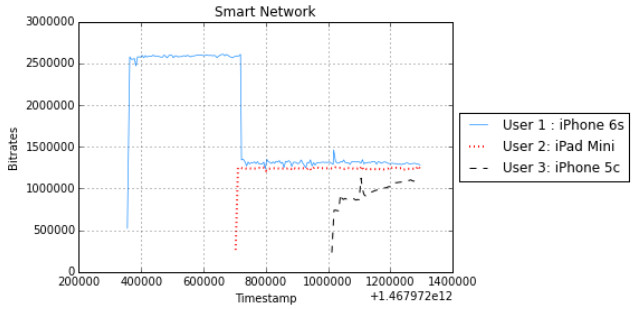
Figure 8: Multiple users experiments for Aggressive and Smart Network algorithm with 1 mbps bandwidth limitation



(a) Aggressive Algorithm



(b) Aggressive Algorithm and bandwidth allocation



(c) Smart Network Algorithm

Figure 9: Video bitrate results for Bandwidth competition experiments for Aggressive and Smart Network algorithms with 4 mbps bandwidth limitation