LEBANESE AMERICAN UNIVERSITY

Dynamic Adaptation for Video Streaming over Mobile Devices

By

Rayan A. Jalloul

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

School of Arts and Sciences
July 2013

Lebanese American University

School of Arts and Sciences - Beirut Campus

Thesis Proposal Form

Name of Student: Rayan Jalloul I.D.#: 200503202

Program / Department: Computer Science and Mathematics

On (dd/mm/yy): 11/01/2013 has presented a Thesis proposal entitled:

Dynamic Adaptation Algorithm for Video Streaming over Mobile Devices

in the presence of the Committee Members and Thesis Advisor:

Advisor: Dr. Sanaa Sharafeddine [redacted]
(Name and Signature)

Committee Member: Dr. Nashat Mansour [redacted]
(Name and Signature)

Committee Member: Dr. Ramzi Haraty [redacted]
(Name and Signature)

Comments / Remarks / Conditions to Proposal Approval:

Date: 11/01/2013 Acknowledged by [redacted]
(Dean, School of Arts and Sciences)

cc: Department Chair
School Dean
Student
Thesis Advisor

Lebanese American University

School of Arts and Sciences - Beirut Campus

Thesis Defense Result Form

Name of Student: Rayan Jalloul I.D.: 200503202

Program / Department: MS in Computer Science/Computer Science and Mathematics

Date of thesis defense: July 25, 2013

Thesis title: Dynamic Adaptation for Video Streaming over Mobile Devices

Result of Thesis defense:

☐ Thesis was successfully defended. Passing grade is granted

☒ Thesis is approved pending corrections. Passing grade to be granted upon review and approval by thesis Advisor

☐ Thesis is not approved. Grade NP is recorded

Committee Members:

Advisor: Dr. Sanaa Sharafeddine [redacted]
(Name and Signature)

Committee Member: Dr. Ramzi Haraty
(Name and Signature)

Committee Member: Dr. Abdul-Nasser Kassar
(Name and Signature)

Advisor's report on completion of corrections (if any): completed

Changes Approved by Thesis Advisor: yes Signature: [redacted]

Date: _____ Acknowledged by _____
(Dean, School of Arts and Sciences)

cc: Registrar, Dean, Chair, Thesis Advisor, Student

Lebanese American University

School of Arts and Sciences - Beirut Campus

Thesis Approval Form

Student Name: Rayan Jalloul I.D. #: 200503202

Thesis Title: Dynamic Adaptation for Video Streaming over Mobile Devices

Program: Master of Science in Computer Science

Department: Computer Science and Mathematics

School: Arts and Sciences

Approved by:

Thesis Advisor: Dr. Sanaa Sharafeddine

Committee Member: Dr. Ramzi Haraty

Committee Member: Dr. Abdul-Nasser Kassa[redacted]

Date: July 25, 2013

THESIS COPYRIGHT RELEASE FORM

LEBANESE AMERICAN UNIVERSITY NON-EXCLUSIVE DISTRIBUTION LICENSE

By signing and submitting this license, you (the author(s) or copyright owner) grants to Lebanese American University (LAU) the non-exclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic format and in any medium, including but not limited to audio or video. You agree that LAU may, without changing the content, translate the submission to any medium or format for the purpose of preservation. You also agree that LAU may keep more than one copy of this submission for purposes of security, backup and preservation. You represent that the submission is your original work, and that you have the right to grant the rights contained in this license. You also represent that your submission does not, to the best of your knowledge, infringe upon anyone's copyright. If the submission contains material for which you do not hold copyright, you represent that you have obtained the unrestricted permission of the copyright owner to grant LAU the rights required by this license, and that such third-party owned material is clearly identified and acknowledged within the text or content of the submission. IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN AGENCY OR ORGANIZATION OTHER THAN LAU, YOU REPRESENT THAT YOU HAVE FULFILLED ANY RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT. LAU will clearly identify your name(s) as the author(s) or owner(s) of the submission, and will not make any alteration, other than as allowed by this license, to your submission.

Name: Rayan A. Jalloul

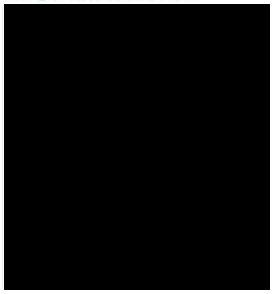
Signature: [redacted] Date: 07-AUG-2013

PLAGIARISM POLICY COMPLIANCE STATEMENT

I certify that I have read and understood LAU's Plagiarism Policy. I understand that failure to comply with this Policy can lead to academic and disciplinary actions against me.

This work is substantially my own, and to the extent that any part of this work is not my own I have indicated that by acknowledging its sources.

Name: Rayan A. Jalloul

Signature: 

Date: 07- AUG - 2013

This thesis is dedicated to my parents for their love, support and encouragement and to their endless sacrifices that have made me who I am today.

I also would like extend my dedication to my siblings who have been a great source of motivation and support all the way.

ACKNOWLEDGMENTS

This research would not have been possible without the help and assistance of many people.

First I would like to express my deep gratitude to my supervisor, Dr. Sanaa Sharafeddine, for her guidance throughout my thesis work. I am also sincerely grateful to the thesis committee members, Dr. Ramzi Haraty and Dr. Abdul-Nasser Kassar, for agreeing to serve on my committee. My gratitude goes also to all my colleagues at the IT department of the Lebanese American University for their assistance throughout the implementation phase.

Finally, endless appreciation goes out to my loving parents, family and friends for being a constant source of inspiration and support during my work. I would not have made it without you.

Dynamic Adaptation for Video Streaming over Mobile Devices

Rayan A. Jalloul

Abstract

Mobile video is expected to generate over 66% of mobile data traffic by 2017. The quality of experience, however, can be frustrating due to wireless channel variations that lead to bandwidth fluctuation. Moreover, different kinds of video players, plugins, and network protocols may be required at the client side to run a given video. As a result, MPEG with the participation of many industries developed the MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH) in order to standardize video streaming over HTTP. MPEG-DASH aims at enhancing the user's quality of streaming experience by dynamically switching between different video encodings depending on the underlying network conditions. The MPEG-DASH standard was published as ISO/IEC 23009-1:2012 in April, 2012. The ISO standard, however, does not provide a normative definition of a DASH client nor identify a specific adaptation algorithm to be used by the client. As a result, several MPEG-DASH implementations with different adaptation methodologies have been proposed in the literature. In this thesis, a new MPEG-DASH dynamic adaptation algorithm for mobile devices is proposed to provide enhanced quality of experience. A comparative study between the proposed adaptation approach and existing MPEG-DASH implementations is presented. The study is based on experiments conducted under different network simulations and results show that the proposed approach ensures a stepwise transition between different video encodings that is consistent with the varying network throughput and client conditions.

Keywords: MPEG-DASH, Adaptive Streaming, Mobile Video, Mobile Devices, Quality of Experience.

TABLE OF CONTENTS

CHAPTER **Page**

LIST OF TABLES

LIST OF FIGURES

ABREVIATIONS

AMOLED:	Active-Matrix Organic Light-Emitting Diode
ARM:	Advanced RISC Machines
BMFF:	Base Media File Format
CDN:	Content Distribution Network
HDS:	HTTP Dynamic Streaming
HLS:	HTTP Live Streaming
HTTP:	Hypertext Transfer Protocol
IP:	Internet Protocol
ISO:	International Organization for Standardization
ITEC:	Institute of Information Technology
LAN:	Local Area Network
LGPL:	Lesser General Public License
MPD:	Media Presentation Description
MPEG:	Moving Picture Expert Group
MPEG2-TS:	MPEG-2 Transport Stream
MPEG-DASH:	MPEG Dynamic Adaptive Streaming over HTTP
OSMF:	Open Source Media Framework
PSNR:	Peak Signal-to-Noise Ratio

QDASH:	QoE-aware DASH
QoE:	Quality of Experience
RTP:	Real-time Transport Protocol
RTSP:	Real Time Streaming Protocol
TCP:	Transmission Control Protocol
UDP:	User Datagram Protocol
URL:	Uniform Resource Locator
VLC:	Video LAN Client
VNI:	Visual Networking Index
WVGA:	Wide Video Graphics Array

CHAPTER ONE

INTRODUCTION

Internet access nowadays is becoming a widely used service on mobile devices. The recent studies of the Cisco Visual Networking Index (VNI) (2013) shows that the smartphones data traffic is expected to grow exponentially and to constitute 67.5% of the total mobile data traffic in 2017. It is also projected that in 2017, the mobile video traffic will generate 66.5% of the total mobile data traffic. Figure 1.1 shows the traffic share forecast for the next coming years and the figures in the legend refer to the traffic share in 2017.

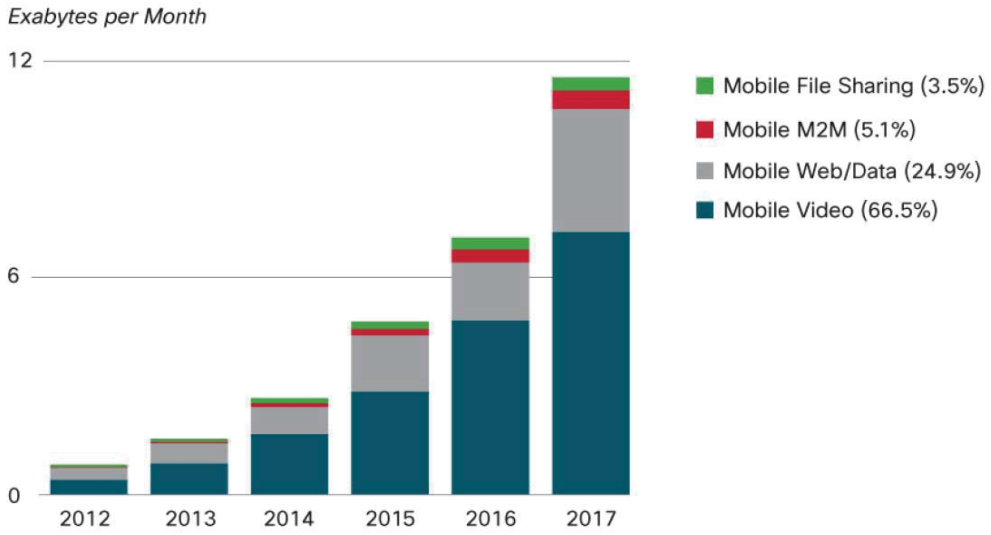


Figure 1.1 - Cisco VNI mobile traffic share forecast

Mobile users expect to have a high quality video experience especially that high quality media is being available on the internet. However, it can be very frustrating for the user to watch a video over the internet for many reasons. The client may need different kinds of players and plugins, should have enough bandwidth, should support different network protocols and should deal with the streaming interruption. Accordingly and with the participation and wide support of many industries, the

Moving Picture Expert Group (MPEG) developed the Moving Picture Expert Group - Dynamic Adaptive Streaming over HTTP (MPEG-DASH) technology, in order to standardize video streaming over HTTP and to make it a simpler and flexible experience by enabling interoperability between clients and servers of various vendors.

Streaming media refers to the delivery method of a multimedia content that is constantly being delivered by a streaming provider and received by and presented to an end-user ("Streaming Media", 2007). The next section provides an overview on the different available streaming methods. Subsequently, the MPEG-DASH standard is presented in details in the section that follows.

1.1 Available Streaming Methods

There are mainly three streaming methods for delivering multimedia content to the client; true streaming, progressive download and adaptive streaming over HTTP.

True streaming streams and provides the content directly to the user. Media streams are not stored on the client's device but instead are streamed and played directly to the client. The most commonly used protocols for true streaming are Real Time Streaming Protocol (RTSP) and Real-time Transport Protocol (RTP). The RTSP is a network protocol designed to control streaming media servers and is used to establish and control media sessions between terminals. RTP is the most used protocol by RTSP servers for delivering media streams ("Streaming Media", 2007).

The main disadvantage of true streaming is that RTSP runs by default on port 554 and that port is usually blocked by firewalls (Fecheyr-Lippens, 2010). In addition, true streaming requires a special streaming server.

In progressive download, the file is downloaded to the client over HTTP and then played locally which provides a high playback quality (Stockhammer, 2011b). HTTP

streaming can be implemented over the already available infrastructure. It uses a well-known port, port 80, that is commonly used and is not blocked by firewalls. However, the transport protocol used for HTTP Streaming is TCP. TCP has a much higher overhead compared to UDP. It tends to handle the transfer of an entire movie file as a one-time transaction which may result in wasting the bandwidth if the user decides to stop watching the movie after the progressive download has started. In addition, this approach is not bitrate adaptive and does not support live media streaming (Stockhammer, 2011b).

Adaptive streaming over HTTP, addresses the weakness of both, the true streaming and the progressive download, mentioned above. It uses HTTP as the protocol for delivering media and HTTP URL's to request media from the server. HTTP has no firewall issues. In addition, HTTP server technology is a commodity and therefore supporting HTTP streaming for millions of users, is cost effective especially that existing Content Distribution Network (CDN) structure can be used. Moreover, the client manages the streaming without having to maintain a session state on the server. Therefore, a large number of streaming clients does not impose any additional cost on server resources beyond standard web use of HTTP.

Adaptive streaming over HTTP delivers the best video quality to the client based on the experienced network and client conditions. It detects the real time status and conditions of the client and network, such as the bandwidth between the client and server, and accordingly adjusts the quality of a video stream. This requires the server to have multiple profiles and qualities of the same video encoded in different bitrates. Consequently, the player client switches between streaming the different encodings depending on the available resources and the underlying network conditions. This would result in having little buffering time, since the video encoding would be close to the rate at which the data is being delivered, resulting in fast startup time and a good overall experience for all types of connections ("Adaptive bitrate streaming", 2011). Several players like Apple HTTP Live Streaming (HLS), Microsoft Smooth Streaming and Adobe's HTTP Dynamic Streaming (HDS) use this approach of Adaptive HTTP Streaming ("Adaptive bitrate streaming", 2011).

1.2 MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH)

Moving Picture Expert Group (MPEG), with the participation and wide support of many industries, developed the MPEG-DASH technology; in order to standardize adaptive video streaming over HTTP and to make it a simpler and flexible experience by enabling clients and servers of different vendors to work together. MPEG-DASH stands for MPEG Dynamic Adaptive Streaming over HTTP. It is an international ISO standard that attempts to unify HTTP streaming to a single standard. It was published as ISO/IEC 23009-1:2012 in April, 2012. MPEG-DASH provides a universal delivery format by reusing the existing content, devices and infrastructure. It enhances the user's quality of media streaming experience by dynamically switching between different video encodings depending on the underlying network conditions.

The MPEG-DASH structure consists mainly of three types of files; "Manifest" (.mpd) which is an XML file describing the segments, "Initialization File" that contains headers needed to decode bytes in segments and the "Segment Files" which contain playable media.

Media Presentation Description (MPD) provides metadata for requesting media segments, such as URLs, in order to locate and download segments. It also provides information regarding the number of representations as well as the characteristics of each representation which is used for rate adaptation purposes, such as the minimum buffer time, representation's bandwidth and segment's duration. The MPD consists of three major components; periods, representations and segments (International Organization for Standardization, 2012) as shown in Figure 1.2 (Stockhammer, 2011a):

- Periods, which is a sequence of one or more periods, is the outermost part of the MPD. Typically they represent the larger pieces of media that are played out sequentially. Each period contains one or more adaptation sets or groups.

Each group may contain one or more encodings of the same media content. Each alternative encoding is called a representation.

- Representations can have different bitrates, frame rates or video resolutions. Each representation describes a series of segments by HTTP URLs. Each representation consists of one or more segments.
- Segments are units that can be referenced by an HTTP-URL. They represent the largest unit of data that can be retrieved by the client. Segments may include media data or metadata to decode and present the included media content. It can be short (1 sec to 10 sec) and long (10 sec to 2 hours).

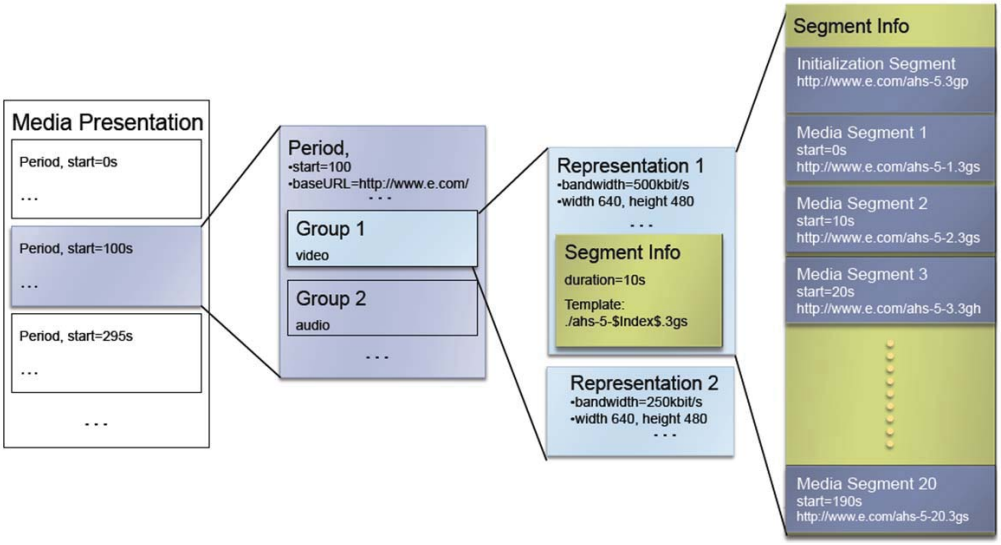


Figure 1.2 – MPD Structure

MPEG-DASH defines the MPD format and delivery formats using ISO BMFF and MPEG2-TS. MPEG-DASH is neither a system nor a protocol (Stockhammer, 2011b). It does not specify content provisioning such as size and duration of segments, number and bitrates of representations, etc... It also does not specify the normative client behavior like the switching between different representations and the way the MPD file is transported. Figure 1.3 shows a typical MPEG-DASH deployment. The formats and functionalities of the red blocks are specified in this standard.

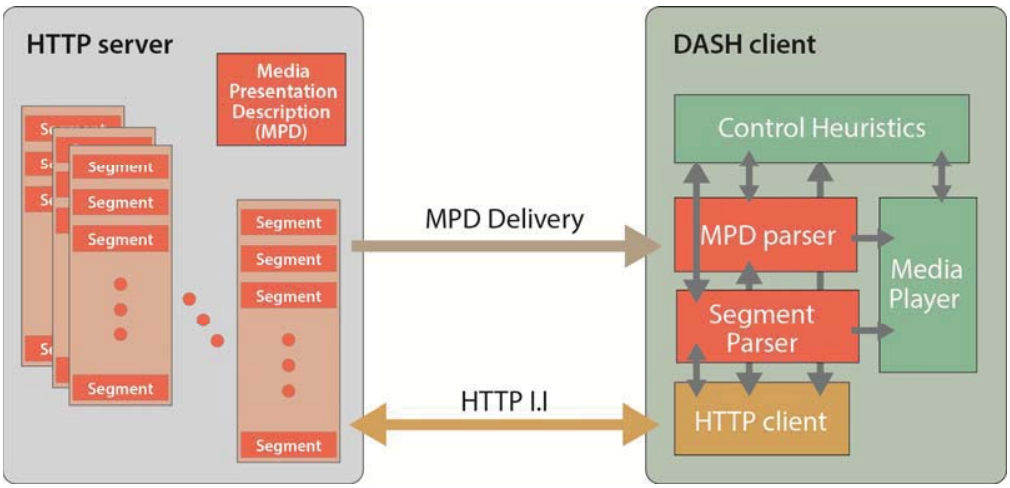


Figure 1.3 – Typical MPEG-DASH deployment

1.2.1 Default Client Behavior

This section presents the default MPEG-DASH client behavior. Figure 1.4 shows an illustration of the MPEG-DASH default client behavior. The video server stores the MPD file and all the different encodings of a video stream. In this figure, the video is encoded into three quality levels A, B and C. The client can be any streaming device; desktop, laptop, phone, notepad, etc… The client's video player is responsible for streaming the video in a dynamic adaptive manner depending on the quality recommended by the quality adaptation algorithm of the client.

The client starts first by requesting the MPD file from the server. Once received, the client parses the file and creates a list of accessible segments for each representation. The client starts first by requesting the lowest encoding available on the server. Afterwards, the quality adaptation algorithm on the client side starts updating the recommended quality based on several criteria such as the network throughput. The client selects the suggested segment from the selected representation and then sends the request (HTTP GET request) to the server. For instance, in this example, the selected quality level of the third segment is quality B.

This process of checking for the appropriate quality level and requesting video segments takes place continuously on each get request and while the client is watching the video. The video player handles reassembling those fragments. The end

result would be having a video with segments of various quality levels. For example, the unified received video fragments of Figure 1.4 consists of the first two segments of quality A, the third, fourth, fifth and sixth segments of quality B and the seventh, eighth and ninth segments of quality C.

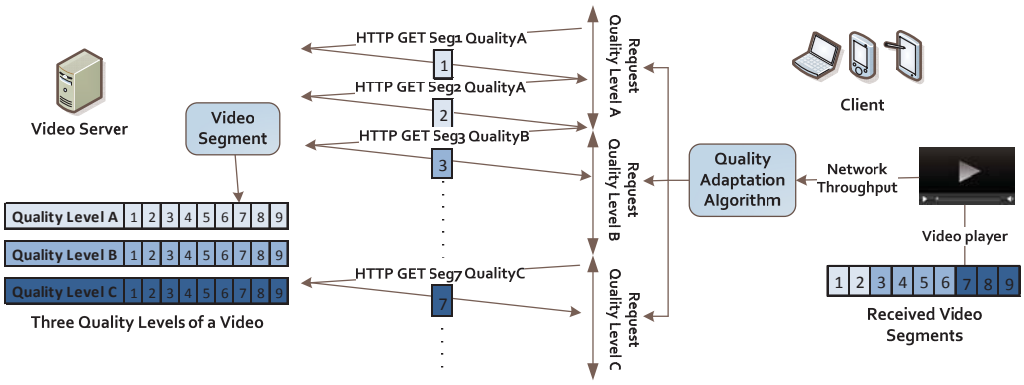


Figure 1.4 - MPEG-Dash with illustration on a default client behavior

As mentioned earlier, the MPEG-DASH ISO standard does not provide a normative definition of a DASH client nor specify the adaptation algorithm to be used by the client. It only specifies the MPD format and the segments format. As a result, several MPEG-DASH implementations have been proposed in the literature and each player uses a different implementation and a different adaptation algorithm. Until this date and as far as we know, there has not been any work that analyzes the current MPEG-DASH implementations. Furthermore, there has not been any proposed adaptation algorithm specific for mobile devices and that provides a balance between the delivered video quality and the video streaming experience. In this thesis, we propose a new MPEG-DASH dynamic adaptation algorithm for mobile devices and present a comparative study between this proposed adaptation algorithm and the current MPEG-DASH implementations.

The rest of the report is organized as follows. Section 2 is devoted to the literature review. In Section 3, the methodology used in this study is defined. Section 4, elaborates more on the experimental setup and implementation procedure. Section 5,

presents an interpretation of the results and outcomes. Finally, the proposed future work and conclusions are stated in sections 6 as a closure to this thesis report.

CHAPTER TWO

LITERATURE REVIEW

Dynamic Adaptive streaming over HTTP is a new video streaming method for delivering high quality media over the internet. It is supported by many industries and is expected to be more popular in the near future. Several proprietary solutions are nowadays available in the market and some of them are undergoing major enhancement in order to ensure a better quality of experience. In this section, we go through the DASH research work that has been around for the last couple of years and we focus more on the work related to the DASH adaptation algorithm.

Thomas Stockhammer (2011b) provided some background and insight to the specifications of Dynamic Adaptive Streaming over HTTP. In his paper, he presented an overview of the design principles and provided a normative description of the Media Presentation Description file, the segments format, and the delivery protocol.

Figure 2.1 presents the suggested architecture for the media distribution for streaming over HTTP. The media preparation process creates segments with different encodings of the media content. These different encodings of the media, along with the Media Presentation Description (MPD) file, are then hosted on one or many media servers. The media servers are HTTP servers and the communication with these servers is based on HTTP connection. The MPD file provides the metadata needed for the client to request the segments using HTTP GET requests. The client's media player controls and manages the streaming session and adjusts the requested bitrates depending on the client state.

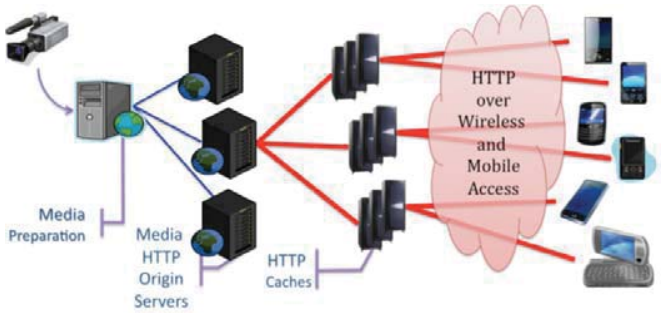


Figure 2.1 - Possible media distribution architecture

Müller and Timmerer (2011) presented an implementation of a VideoLAN Client (VLC) media player plugin that enables MPEG-DASH. The implementation of the MPEG-DASH standard is based on VLC media player and is fully integrated into its structure. The plugin is licensed under the GNU Lesser General Public License (LGPL) and is available as an open source. It has a modular design consisting of a central core and more than 380 modules. The DASH plugin is located at the stream filter module of VLC and consists of four major components shown in Figure 2.2 and of two controller classes.

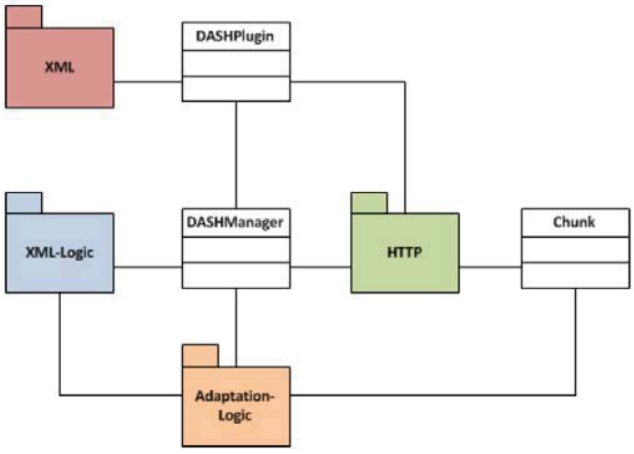


Figure 2.2 - High-Level DASH Plugin Architecture

The first component is the “XML” component that handles the XML parsing of the MPD and in addition provides support to the other components. The second component is the “HTTP” component that is responsible for handling all the HTTP connections during the session. The third component is the “XML-Logic” component that adds the logic to the data representation provided by the XML component. The fourth component is the “Adaptation-Logic” component that handles the adaptation process. This component is very flexible and can be changed without affecting the other components. As a result, different adaptation logics or MPEG-DASH profiles can be integrated into this module thus making it attractive for the research community.

Müller, Lederer, and Timmerer (2012) implemented an MPEG-DASH improved model based on the DASH VLC Plugin by introducing HTTP/1.1 pipelining. A detailed evaluation was presented of the proposed MPEG implementation in comparison to the most popular dynamic streaming solutions over HTTP, i.e., “Microsoft Smooth Steaming”, “Adobe HTTP Dynamic Streaming (HDS)” and “Apple HTTP Live Streaming (HLS)”. These systems were evaluated under restricted conditions and that is in vehicular environments. The segment length in this study was two seconds as it was required by “Microsoft Smooth Streaming”. Bandwidth traces were captured under vehicular mobility and in real world mobile network. The experimental results showed that “Microsoft Smooth Streaming” attained the highest average bitrate and the second lowest number of stream switches. “Apple HTTP Live Steaming” provided the lowest overall bitrate in comparison to the other systems. “Adobe HTTP Dynamic Streaming” was the only system that did not deliver a smooth playback. Moreover, the adaptation process was somehow unpredictable and the rate selection was more of a binary decision between the highest and the lowest representation. Consequently, this does not guarantee a smooth playback and leads to several stalls followed by re-buffering periods and eventually provided a low Quality of Experience. Finally, the proposed VLC prototype implementation achieved the minimum criterion of smooth playback and the second best overall bitrate which indicates the capability and competency of the MPEG-DASH standard.

Liu, Bouazizi, and Gabbouj (2011) presented an innovative method to determine the segment duration that provides accurate and fast rate adaptation in adaptive streaming over HTTP. The current state-of-the-art rate adaptation methods rely on historical segment information in order to estimate the network capacity. Based on the varying network resources, the client adjusts the bitrates of the segments and requests the media segments accordingly. However, this approach estimates the lowest segment duration that produces a smooth rate based on the underlying network conditions by using the following derived formula:

$$SD = RTT \frac{8+\frac{6p}{b}\sqrt{\frac{2b}{3p}}+6\sqrt{\frac{16}{9}-4bp\varepsilon+\frac{2p}{3b}\left(4\sqrt{\frac{2b}{3p}}+1\right)}}{9bp\varepsilon}$$

where, *RTT* denotes the round trip time, *b* represents the number of acknowledged packets, *p* represents the packet loss rate and ϵ denotes the upper limit of the variance of the average congestion window. This duration is further refined in order to prevent buffer drainage.

The simulation results of the proposed segment duration method showed that the rate adaptation algorithm was capable of achieving the right media bitrates based on the network conditions. In addition, this approach reduced the play-back interruption compared to the rate adaptation methods of the available adaptive streaming methods over HTTP.

Mok, Luo, Chan, and Chang (2012) proposed a QoE-aware DASH (QDASH) system, to improve the client's quality of video experience. Adaptation schemes, such as Adobe's Open Source Media Framework (OSMF), rely on historical network throughput received by the video player for quality level switching; and on video buffer to reduce possible artifacts caused by sudden change in network conditions. However, depending on the throughput measurements and averaging techniques, the measured values do not always reflect the correct network conditions and hence wrong decisions can be made when choosing the quality level. Accordingly, this paper presented a new approach that detects the highest quality level supported by the current network conditions and presented an enhancement to the Adobe's OSMF adaptation algorithm by proposing a QoE-aware quality adaptation algorithm for

DASH, based on gradual quality level switching taking into consideration the network conditions.

QDASH system is composed of two modules; QDASH-abw and QDASH-qoe. QDASH-abw is an approach that detects the highest quality level supported by the current network conditions. This module is part of a measurement proxy that is placed in front of the video server and that controls the video packets in streamed data. The available bandwidth is measured by RTT variations. The evaluation results of this methodology showed that QDASH-abw is precise and is sensitive to bandwidth variations. Based on this timely bandwidth measurement, QDASH-qoe assists the client in selecting the most suitable video quality level. Both modules, the network measurement procedure and the QoE-aware quality adaptation, were integrated into a comprehensive DASH system, a QoE-aware DASH (QDASH) system.

In order to assess the QoE of the rate transitions of QDASH, subjective experiments were carried out instead of the common objective approaches. Objective metrics, such as Peak Signal-to-Noise Ratio (PSNR), consider only the spatial quality and not the switch in quality and that is why in this case it is not considered beneficial. Accordingly, subjective assessments are implemented in order to evaluate the observed QoE. The video player used was developed using Strobe Media Playback (SMP) and Open Source Media Framework (OSMF). The OSMF quality adaptation algorithm was modified to follow a predefined quality pattern using predefined rule sets. The subjects rate the QoE based on different rule sets. Each rule set, R, designates a scenario and is expressed as follows:

$$R = \{ < l_0, d_0 > , < l_1, d_1 >, \ldots\ldots ,< l_k, d_k > \} \text{ ; where } k = 0, 1, 2, ..., k$$

The rule set contains at least one rule tuple. This tuple is represented by $< l, d >$ where l is the quality level and d is the bytes that need to be downloaded at quality l. The player plays the tuples sequentially, one after the other, until all the tuples are used. So, these rule sets are used to define simulations for various quality level changes. In these experiments, a simulation with the maximum quality drop and that is a drop from 4 Mbps to 400 Kbps after playing the first three fragments is presented. So the subjects first watch three video fragments at the quality defined in

the first tuple and afterwards one or two intermediate quality levels are inserted and played to finally reach the lowest quality level. These different emulations are represented by different Rule sets.

The videos used are eleven short video clips of different kinds, such as sports, animation, etc…. The duration of each video is around 90 seconds. These videos were encoded in different quality levels. A total of 24 volunteers took part in this subject assessment, 5 females and 19 males. The subjects were with normal vision and were not experts in QoE assessment. They were informed that they will experience a sudden drop in quality. Moreover, they were recommended to watch the video in full-screen mode, and were requested not to alter the video playback time. Each subject was asked to watch 11 videos selected randomly and belonging to different Rule sets. After each playback, the subjects were asked to separately rate the video quality in terms of the sound and picture quality and playback smoothness. Moreover, they were asked to provide a score on the observed overall quality rating from '7' (Excellent) to '1' (Bad). After validating that all rule sets were equally distributed over all samples, 242 valid samples were obtained. No subject rated any sample '1' and only two samples were given a rate of '7'.

The results of this rating showed that users prefer a gradual quality transition instead of a direct quality switching to the required quality level. The rule sets without any intermediate levels provided the lowest quality level and inserting intermediate levels gave a better QoE. Moreover, two intermediate levels had a trivial influence in improving the QoE while one intermediate level provided the highest perceived QoE.

Adxic, Kalva, and Furht (2012) presented a new content preparation approach for adaptive streaming over HTTP. This approach handles content encoding and provides an optimized content-based segmentation algorithm for adaptive streaming over HTTP. The segmentation process provides a balance between the network requirements and rate distortion. Moreover, a save in bandwidth is attained while reserving quality. This segmentation process consists first of a scene detection phase. Afterwards, I-frames are inserted on each scene-cut. Based on the scene duration, the optimal segment duration is detected and if needed additional I-frames are added. The next step consists of choosing the ideal frames for segment boundaries and this

selection is based on the frame rate with which the content is encoded with. As a result, the segmentation process is implemented allowing the maximum possible performance with the least rate distortion.

The experimental results of this algorithm showed an outstanding performance compared to common segmentation practices. The video stream was customized for providing the user with a better quality of experience. In addition, it saved on average 10% of bandwidth while reserving the same quality level in comparison to popular segmentation techniques. This approach is appropriate for both “Live” and “On-Demand” streaming. The internet service providers and the end users can benefit from this methodology by saving bandwidth without any loss in quality.

CHAPTER THREE

METHODOLOGY

This chapter provides information about the experimental methodologies that are used in the implementation of the proposed MPEG-DASH adaptation algorithm. First, the main quality adaptation algorithms available in the literature are stated and then the proposed MPEG-DASH adaptation algorithm is presented. Next, the VLC media player architecture is defined. Finally, the datasets used for testing and comparing the performance of the algorithms are listed.

3.1 Quality Adaptation Algorithms

The quality adaptation algorithm is the main element of MPEG-DASH. Most of the current adaptation schemes are based on the client's network throughput where the selected quality level is chosen as close as possible to the measured throughput.

In section 3.1.1 and 3.1.2, two MPEG-DASH adaptation algorithm implementations that are used in the comparative study are specified. Both implementations are based on VLC media player. In section 3.1.3, the proposed MPEG-DASH adaptation algorithm is presented.

3.1.1 Original VLC Media Player Algorithm

The VLC media player adaptation algorithm depends on the historical network throughput of the whole session, at download time of segment *i*, in order to determine the most suitable quality level. The average throughput is calculated by determining the bytes read over the whole session and dividing it by the total time of the session, at download time of segment *i*. So on each get request, this algorithm selects the maximum available bandwidth close to the measured average throughput

of the session. However, if the buffer level is less than 30%, this algorithm selects the minimum available encoding, which eventually results in a non-stepwise transition between different encodings.

The VLC Media Player adaptation algorithm is as follows:

$$\text{maxbw}(s_i) = \begin{cases} \text{Lowest}_{\text{Encoding}} & \text{if} \quad 0.00 \leq bl_i < 0.30 \\ \text{Avgbps}_{\text{Session}} & \text{if} \quad 0.30 \leq bl_i < 1.00 \end{cases}$$

Where;

$i \in [1 , N]$	–	The segment index
bl_i	–	The buffer level at the download time of segment *i*
$\text{maxbw}(s_i)$	–	The maximum bandwidth available for segment *i*
$\text{Lowest}_{\text{Encoding}}$	–	The segment with the lowest available encoding of the movie
$\text{Avgbps}_{\text{Session}}$	–	The average bitrate of the whole session at the download time of segment *i*

3.1.2 DASH VLC Plugin (Buffer)

The adaptation algorithm that is used by the DASH VLC Plugin (Müller, Lederer, & Timmerer, 2012) measures the effective bitrate while downloading each segment and accordingly builds an adaptation decision based on the below algorithm. This algorithm uses the non-stepwise approach. The maximum bandwidth of segment s_i is determined by multiplying the measured bandwidth while downloading the s_{i-1} segment by a predefined factor. The measured bandwidth while downloading a segment is calculated by dividing the bytes read while downloading this segment by

the time it took to download it. The predefined factor is based on the buffer level as shown below.

$$\text{maxbw}(s_i) = \begin{cases} \text{bw}(s_{i-1}) * 0.3 & \text{if} \quad 0.00 \leq bl_i < 0.15 \\ \text{bw}(s_{i-1}) * 0.5 & \text{if} \quad 0.15 \leq bl_i < 0.35 \\ \text{bw}(s_{i-1}) & \text{if} \quad 0.35 \leq bl_i < 0.50 \\ \text{bw}(s_{i-1}) * (1 + 0.5 * bl_i) & \text{if} \quad 0.50 \leq bl_i < 1.00 \end{cases}$$

where;

$i \in [1, N]$ – The segment index

bl_i – The buffer level at the download time of segment *i*

$\text{bw}(s_i)$ – The bandwidth which was measured during the download of segment *i*

$\text{maxbw}(s_i)$ – The maximum bandwidth that is available for segment *i*

3.1.3 The Proposed Algorithm

The proposed adaptation algorithm determines the maximum available bandwidth of segment s_i based on equation (1).

$$\text{maxbw}(s_i) = \{ w_1 \text{repbw}(s_{i-1}) + w_2 \text{bw}(s_{i-1}) \quad \text{where} \quad w_1 + w_2 = 1 \qquad (1)$$

Where;

$i \in [1, N]$ – The segment index

$\text{bw}(s_i)$ – The bandwidth which was measured during the download of segment *i*

$\text{repbw}(s_i)$ – The representation's bandwidth of segment *i*

w_1 and w_2 – weighting factors

maxbw(s_i) – The maximum bandwidth that is available for segment *i*

This estimation is done on each http get request. The maximum available bandwidth value is calculated by adding of the value of the representation's bandwidth of segment s_{i-1} multiplied by a weighting factor w_1, to the measured bandwidth during the download of segment s_{i-1} multiplied by a weighting factor w_2.

The representation's bandwidth is the minimum bandwidth in bits per second (bps) of a theoretical constant bitrate channel over which the segments of a representation can be continuously delivered to the client (International Organization for Standardization, 2012). The measured bandwidth while downloading a segment is calculated by dividing the bytes read while downloading this segment by the time it took to download it. The summation of the weighting factors, w_1 and w_2 is equal to 1. These weights are set in order to regulate the impact of the measured bandwidth on the previously requested bandwidth in a way to ensure a stepwise transition between the different encodings. The weights of this algorithm are set as follows; w_1 is set to 0.2 and w_2 is set to 0.8. These values are set based on experimental tests that were done using different weights and under different network condition. More details are provided in section 4.2.

3.2 VLC Architecture

VLC is a multiplatform program, written in C. It has a modular design consisting of a central core and more than 380 modules ("VLC media player", 2013).

LibVLCcore is the main core of the framework. It provides an object oriented layer to C and is responsible for module loading and unloading. Moreover, it offers a set of abstract functionalities related to data input, multiplexing/de-multiplexing, in addition to audio and video output (Campanelli, 2013).

Modules make up the main layers of VLC. They provide most of the framework's functionalities and are categorized according to their capabilities. The five main layers of VLC are as follows (Müller & Timmerer, 2011):

- Interface: consists of all the modules related to the user's interaction like play, stop and forward.
- Access: contains all the modules that open files or streams. This layer provides the demux layer with data.
- Demux: different formats are demultiplexed at this layer.
- Decoder: provides modules that are capable of decoding the demultiplexed data from the demux layer.
- Output: displays the decoded data on the screen

The DASH plugin is located at the stream filter module of VLC and consists of four major components and two controller classes as discussed in chapter 2 (Müller & Timmerer, 2011). The Adaptation-Logic component of the DASH plugin handles the adaptation process and can be modified without affecting the other components. Consequently, different adaptation logics can be integrated in this module for research purposes.

3.3 Datasets

The datasets used in this research are public DASH datasets specific for android mobile devices. These datasets were published for research work by Lederer, Müller, and Timmerer (2012) at the Institute of Information Technology (ITEC) from the Alpen Adria Universitaet Klagenfurt.

"Big Buck Bunny" and "Of Forest and Men" are the two datasets selected from the datasets container in order to perform all the experiments. These datasets were generated using DASH content generation tool DASHEncoder, an open source DASH content generation tool (Lederer, Müller, & Timmerer, 2012). Table 3.1 summarizes the characteristics of these datasets.

“Big Buck Bunny” is a short animation film and has a quality of □20p. The content of this film has been encoded at □different bitrates□□00, 1200, 1□00 and 2000 Kbps. The duration of the movie is □minutes and 33 seconds. Each encoding consists of □0 segments and the size of each segment is 1□seconds.

“Of Forest and Men” is a movie that has a quality of □□□p. The content of this movie has been encoded at □ different bitrates□□00, 1100, 1□00, 1□00, 2000, 2□00, 3000, □000 and □000 Kbps. The duration of the video is 10 minutes. Each encoding consists of 31 segments and the size of each segment is 1□seconds.

Table 3.1 - Datasets characteristics

Dataset	Type	Duration	Quality	Encodings (Kbps)	Segments per Encoding	Segment Duration
Big Buck Bunny	Animation	10 min	□20p	□00, 1200, 1□00 and 2000 Kbps	□0	1□sec
Of Forest and Men	Movie	□min and 33 sec	□□□p	□00, 1100, 1□00, 1□00, 2000, 2□00, 3000, □000 and □000 Kbps	31	1□sec

CHAPTER FOUR

IMPLEMENTATION

In this chapter, the main methodologies used in the implementation of the proposed M□E□-DASH adaptation algorithm are defined. Section □1 gives an overview of the experimental setup. Section □2, provides verification for the selection of the weights in the proposed algorithm. In the last section, the different network simulations that were implemented on both datasets are presented.

4.1 Experimental Setup

In this section, the network diagram of the experimental setup is presented. The video streaming experiments took place over two sites. The first site is the "Lebanese American University" in Beirut, Lebanon. The other site is "Alpen Adria Universitaet Klagenfurt" in Klagenfurt, Austria.

The streaming server is located in "Alpen Adria Universitaet Klagenfur". The datasets as per section 3.3 are all available on this server for the public in order to be used for research purposes.

On the "Lebanese American University" site, the network infrastructure consists of the following: smartphone, wireless router, Local Area □etwork (LA□) switch, packet shaper and wise access point. The packet shaper is used for bandwidth management in order to control the bandwidth limits for the presented experiments. The phone used is Samsung □alaxy S II (□T-I□100), a touchscreen Android smartphone. This smartphone was produced by Samsung Electronics. It was launched with □ingerbread platform, Android 2.3. Later, it was updated to Ice Cream Sandwich, Android □0.□ and in □anuary 2013 updated to □elly Bean, Android □1. The □alaxy S II has a 1.2 □Hz dual-core A□M Cortex-A□ processor, 1□□B of user memory and 1 □B of □AM. The screen display is 10.□cm □ V□A Super AMOLED □lus. The camera is □ Megapixel that enables 10□0p full high definition video

recording. The user-replaceable battery gives up to ten hours of heavy usage, or two days of lighter usage (“Samsung alaxy S II”, 2013).

The phone connects to the Linksys wireless router. A private Internet rotocol (I) address is assigned to the smart phone. The wireless router is connected to the Cisco 30 LA switch where a Virtual Local Area etwork that connects the phone to the Internet is configured specifically for experimental purposes. In order to configure the different bandwidth emulations that are used in these experiments, the LA switch is directly connected to a aketeer Blue Coat 00 packet shaper. The acket shaper is connected to the Internet through the wise access point through a public I address. So the packet shaper is connected to the LA through the Cisco 30 LA switch and to the Internet through the wise access point.

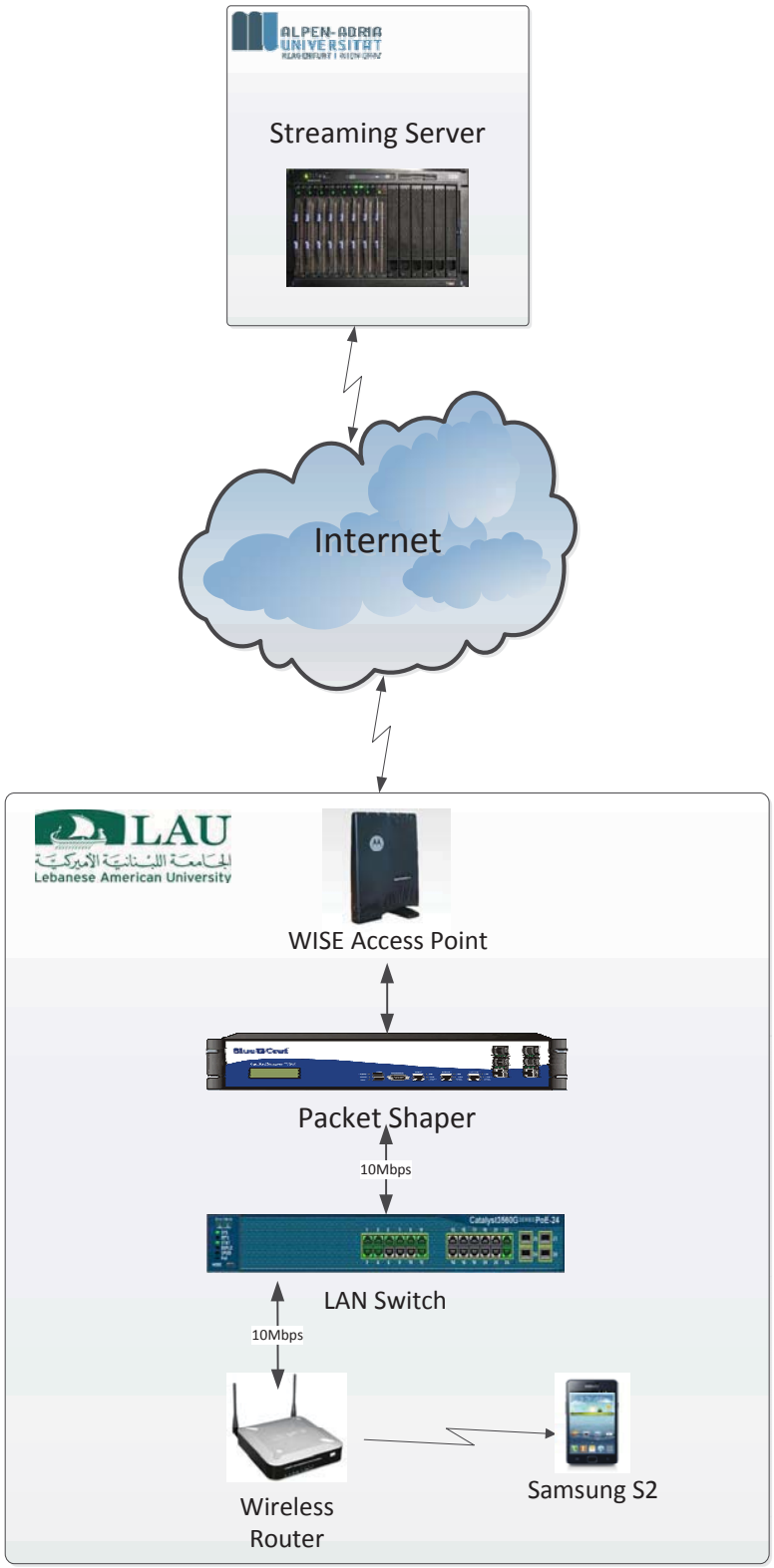


Figure 4.1 - Network Diagram

4.2 Weights Selection

In order to determine the right weights to be used in the proposed adaptation algorithm, several experiments were implemented with variable values for w_1 and w_2 where $w_1 + w_2 = 1$.

The values for w_1 in these experiments were as follows: 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55 and 0.6. These experiments were done using the "Of Forest and Men" dataset and under stable and variable network conditions. Since the algorithm is based on bandwidth values, the results presented here are expected to be consistent with other datasets regardless of the video type. In addition, this dataset has many encodings and accordingly, it can provide more valid and clear results.

In this section the most interesting results, that validate the decision of setting the weights values to $w_1 = 0.2$ and $w_2 = 0.8$, are only listed. "Variable rate bandwidth" emulations were run on this dataset using the different weights listed above. The bandwidth was consecutively set to 2 Mbps, 3 Mbps, 4 Mbps, 2 Mbps, 1 Mbps and 3 Mbps for the duration of 20 seconds, 30 seconds, 30 seconds, 30 seconds, 40 seconds and 20 seconds respectively. The test duration was set to 200 seconds. The results are shown in Figure 12, Figure 13, Figure 14 and Figure 15.

Proposed Algorithm - $w_1 = 0.2$

Figure 12 shows the results of setting the weights to $w_1 = 0.2$ *and* $w_2 = 0.8$. As shown in this figure, the switching between the different encoding was in response to the changes in the network conditions. The first encoding selected was the lowest encoding that is 400 Kbps. However, since the bandwidth was set to 2 Mbps, the next get request selected the 2 Mbps encoding based on equation (1), and the same goes for the switching to 3 Mbps and 4 Mbps. However, with the decrease in bandwidth to 2 Mbps and then 1 Mbps, this proposed algorithm encodings decreased stepwise from 4 Mbps to 2.4 Mbps, 1.1 Mbps and 400 Kbps consequently. When the bandwidth was restored back to 3 Mbps the proposed algorithm selected the 3 Mbps encoding on the next request. The total number of switches is 4

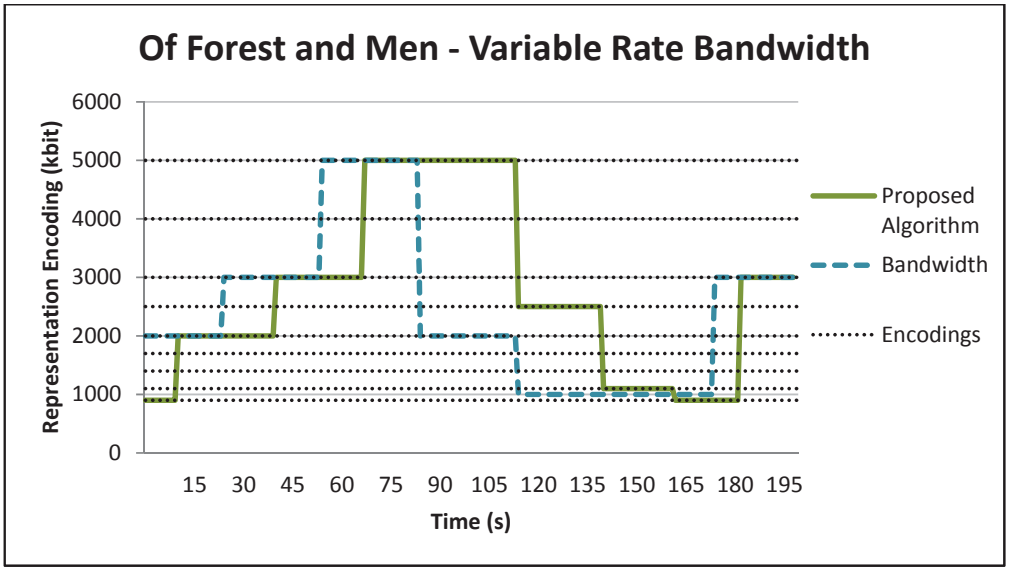


Figure 4.2 - "□roposed Algorithm" streaming "Of Forest and Men" over variable rate bandwidth where $w_1 = 0.2$ and w_2 □0.□

Proposed Algorithm - $w_1 = 0.3$

Figure □3 shows the results of setting the weights to $w_1 = 0.3$ *and* $w_2 = 0.7$. In this approach, we notice the increase in the number of switches in both directions, when the bandwidth was increased and deceased. Specifically, we notice that the second get request on the 11th second did not request the 2 Mbps encoding directly, instead the 1.□Mbps was selected and by this adding an additional switch to this adaptation algorithm. In addition, when the bandwidth was decreased, the selected encodings were higher than the previous approach and thus resulted in a delay in response to the change in conditions, mainly when the bandwidth was restored back to 3 Mbps. □e can see that in the previous approach when the bandwidth was directly restored to 3 Mbps, the next selected bandwidth was 3 Mbps which is the desired encoding. However, in this case an additional switch to □00 Kbps can be observed on the 1□□th second and the increase in bandwidth was to 2□00 Kbps instead of 3 Mbps on the 1□3rd second. The total number of switches is □

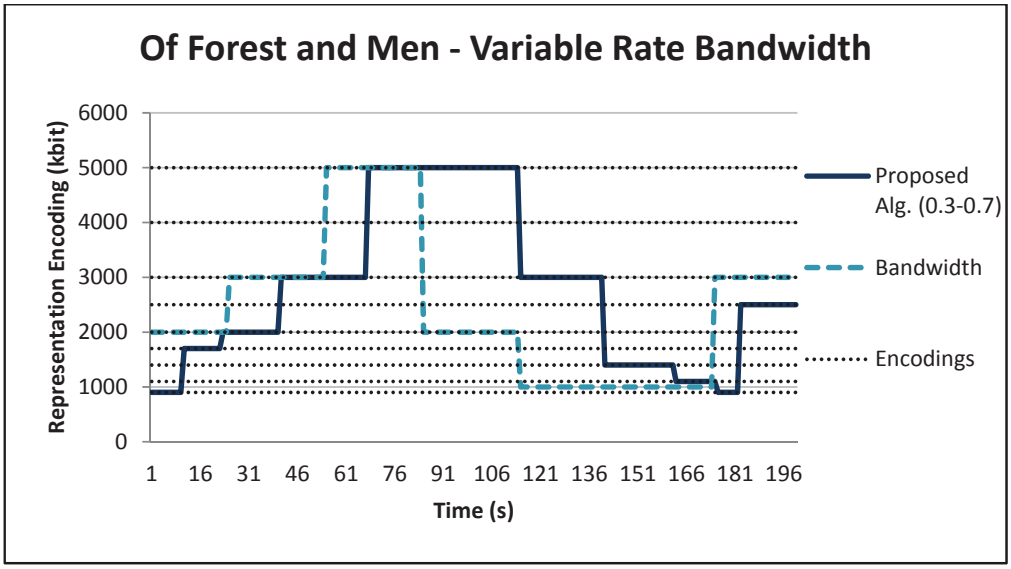


Figure 4.3 - "□roposed Algorithm" streaming "Of Forest and Men" over variable rate bandwidth where $w_1 = 0.3$ and w_2 □0.□

Proposed Algorithm - $w_1 = 0.5$

Figure □.□shows the results of setting the weights to $w_1 = 0.5$ *and* $w_2 = 0.5$. In this approach, we notice the increase in the number of switches in both directions, when the bandwidth was increased and deceased, even more than the "$w_1 = 0.3$" approach. □ e can see the delay in the response to the changes in bandwidth due to the increase in the number of switches. In addition, when the bandwidth was increased to 3 Mbps on the 1□3rd second, the next encoding requested on the 1□3rd second was only 2 Mbps instead of 3 Mbps. The total number of switches is 11.

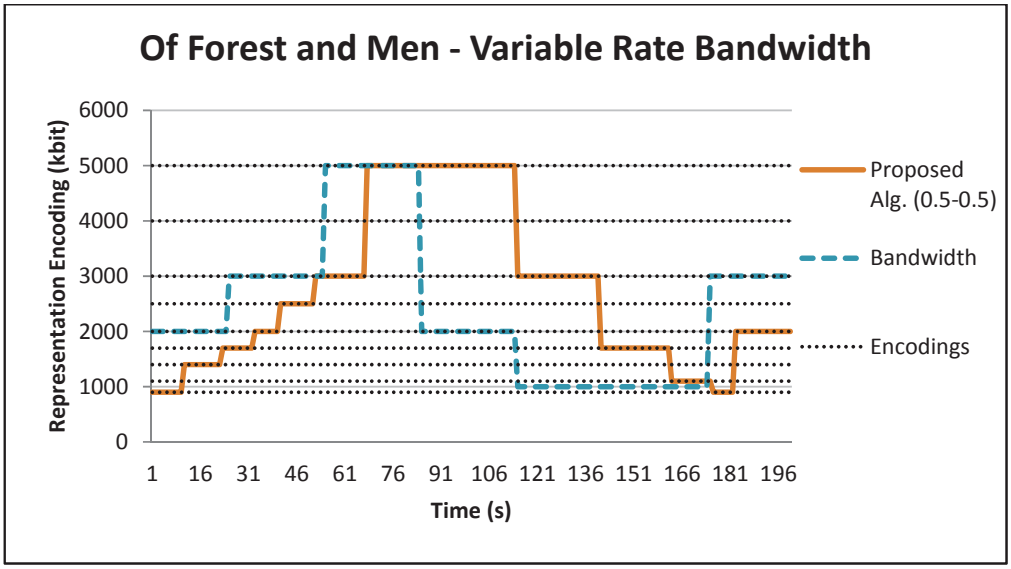


Figure 4.4 - "□roposed Algorithm" streaming "Of Forest and Men" over variable rate bandwidth where $w_1 = 0.5$ and w_2 □0.□

Proposed Algorithm - $w_1 = 0.8$

Figure □.□shows the results of setting the weights to $w_1 = 0.8$ *and* $w_2 = 0.2$. In this approach, we can see that the target available bandwidth rates are not met. Instead the range of the requested encodings was between □00 Kbps and 3 Mbps which does not make use of the available bandwidth. The total number of switches is 11.

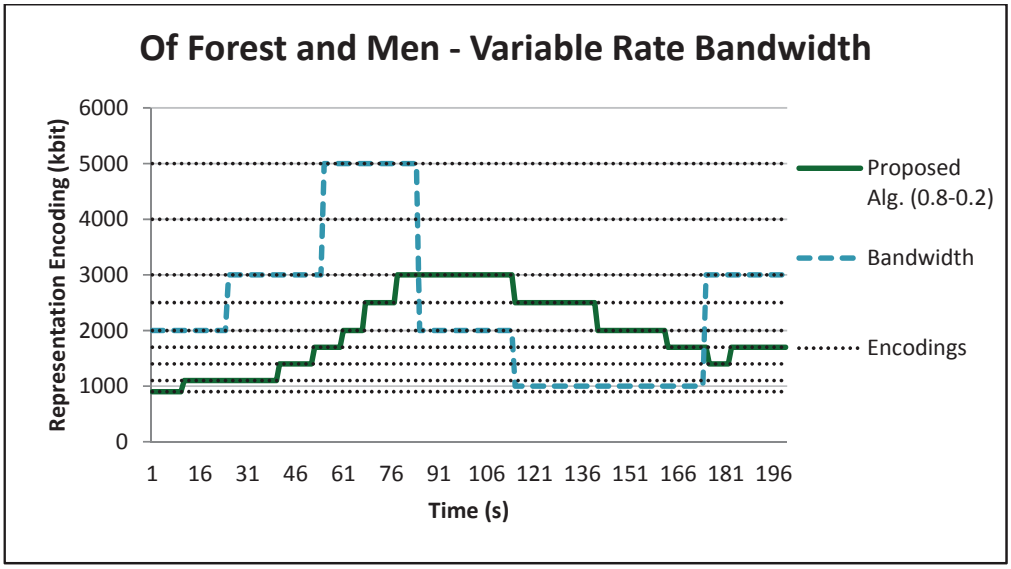


Figure 4.5 - "□roposed Algorithm" streaming "Of Forest and Men" over variable rate bandwidth where $w_1 = 0.8$ and w_2 □0.2

The table below summarizes the weights selection results:

Table 4.1 - □ eights selection results

Weights	Lowest Encoding	Highest Encoding	Number of switches
w_1 □0.2 □ w_2 □0.□	□00	□000	□
w_1 □0.3 □ w_2 □0.□	□00	□000	□
w_1 □0.□□ w_2 □0.□	□00	□000	11
w_1 □0.□□ w_2 □0.2	□00	3000	11

The results showed that the values $w_1 = 0.2$ and w_2 □ 0.□ provided a better performance. This choice is based on the assessment results of the experiments done by Moke et al. (2012) discussed in chapter two, where the results showed that two intermediate levels had a trivial influence in improving the □oE while one intermediate level provided the highest perceived □oE. Accordingly, higher number of switches is not required for having a better quality of experience. Moreover, □i, Eg, Eichhorn, □riwodz, and Halvorsen, performed a study on the effect of spatial flicker in adaptive streaming and the results of the subjective assessments showed

that high frequency of quality changes might lead to a decrease in the quality of experience (□i, Eg, Eichhorn, □riwodz, & Halvorsen, 2011)

And as shown in Table □1, $w_1 = 0.2$ and w_2 □0.□provided an adaptive switching with minimal number of switches under variable rate bandwidth while making use of the maximum available bandwidth. Moreover, these weights guaranteed a stepwise transition between different encodings under stable network conditions as shown later in chapter □ Accordingly, the weights selected and used throughout the rest of the experiments were $w_1 = 0.2$ and w_2 □0.□

4.3 Experimental Simulations

The experiments were done using two datasets "Big Buck Bunny" and "Of Forest and men". For each dataset, the packet shaper was configured at low, high, average and variable bitrate bandwidth depending on the available encodings of each dataset.

Eight experimental simulations where done using the Original, Buffer and □roposed algorithm as shown in Table □2:

Table 4.2 - Experimental simulations

Bandwidth	Big Buck Bunny	Of Forest and Men	Test Duration (s)
Low Bandwidth	1 Mbps	1 Mbps	1□0 seconds
High Bandwidth	2 Mbps	□Mbps	1□0 seconds
Average Bandwidth	1.□Mbps	2.□Mbps	1□0 seconds
Variable □ate Bandwidth	• 1.0 Mbps for 2□s • 1.□Mbps for 30s • 2.0 Mbps for 30s • 1.2 Mbps for 30s • 0.□Mbps for □0s • 1.□Mbps for 2□s	• 2.0 Mbps for 2□s • 3.0 Mbps for 30s • □0 Mbps for 30s • 2.0 Mbps for 30s • 1.0 Mbps for □0s • 3.0 Mbps for 2□s	200 seconds

CHAPTER FIVE

RESULTS AND ANALYSIS

In this chapter, the different network emulations that the experiments were based on are listed. Moreover, the results of these different experiments, using the "Big Buck Bunny" and "Of Forest and men" datasets, are discussed in details.

5.1 Big Buck Bunny Experiments

Low Bandwidth

The low bandwidth emulations of the "Big Buck Bunny" were done over 1 Mbps. Accordingly, the packet shaper bandwidth was set to 1 Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □1, Figure □2 and Figure □3 for the Original, Buffer and □roposed algorithm respectively.

The "Big Buck Bunny" experiments under 1 Mbps show a similar performance for the "Original" and the "□roposed Algorithm" implementations with a stable encoding of □00 Kbps as shown in Figure □1 and Figure □3. On the other hand, when it comes to the "Buffer" approach, Figure □2 shows that at the 113th second, the selected encoding increased to the maximum available bandwidth and that is 2 Mbps. This is the result of having a full buffer. However, at the 1□0th second a drop again to □00 Kbps occurs as result of having a low buffer level. Consequently, the "Original" and the "□roposed Algorithm" provided better performance.

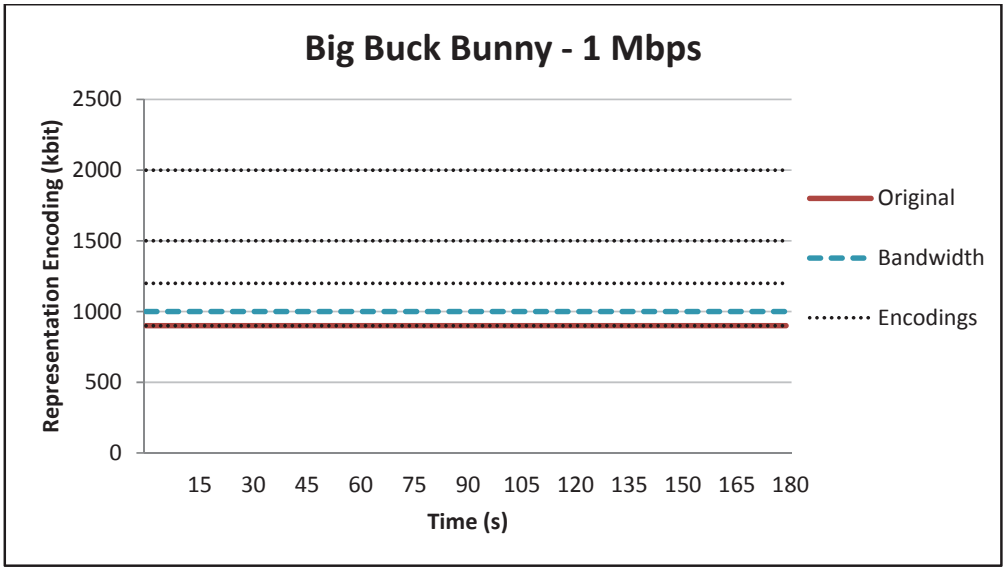


Figure 5.1 - "Original Algorithm" streaming "Big Buck Bunny" over 1 Mbps

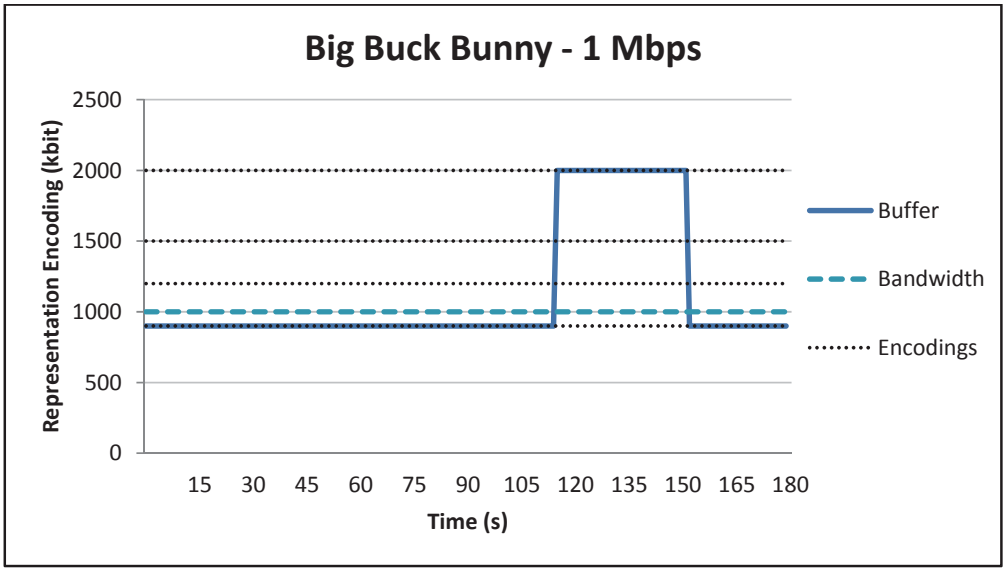


Figure 5.2 - "Buffer Algorithm" streaming "Big Buck Bunny" over 1 Mbps

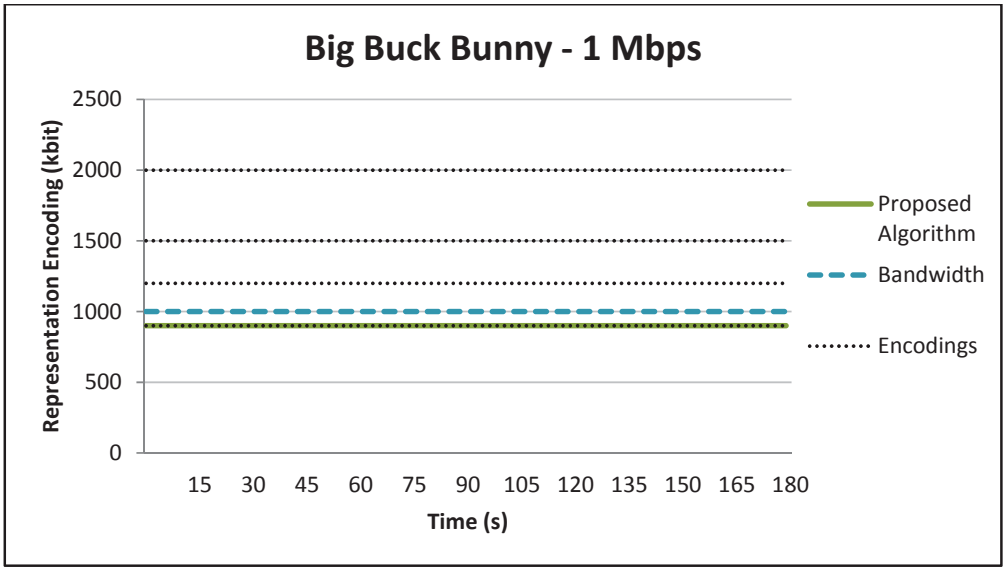


Figure 5.3 - "□roposed Algorithm" streaming "Big Buck Bunny" over 1 Mbps

High Bandwidth

The high bandwidth emulations of the "Big Buck Bunny" were done over 2 Mbps. Accordingly, the packet shaper bandwidth was set to 2 Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □□, Figure □□and Figure □.□□for the Original, Buffer and □roposed algorithm respectively.

The "Big Buck Bunny" experiments under 2 Mbps show a similar performance for the "Original", "Buffer" and "□roposed Algorithm" implementations with a stable encoding of 2 Mbps as shown in Figure □□, figure □□and Figure □□ However, the "Original" and "□roposed Algorithm" reached the 2 Mbps faster than the "Buffer" algorithm. This delay in the "Buffer" approach reflects the buffer status at that time and supposedly at a level less than □0□. The three algorithms provided good performance.

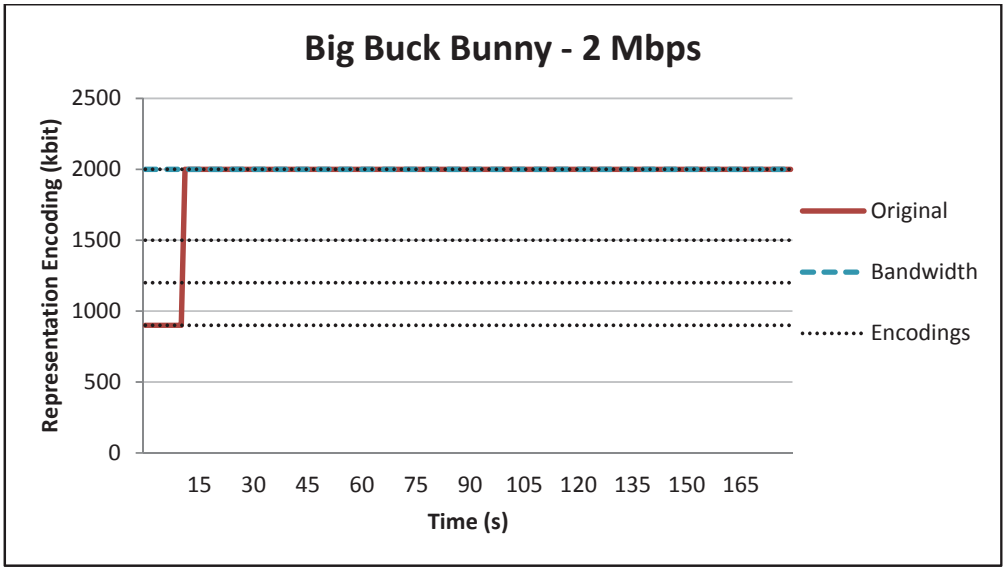


Figure 5.4 - "Original Algorithm" streaming "Big Buck Bunny" over 2 Mbps

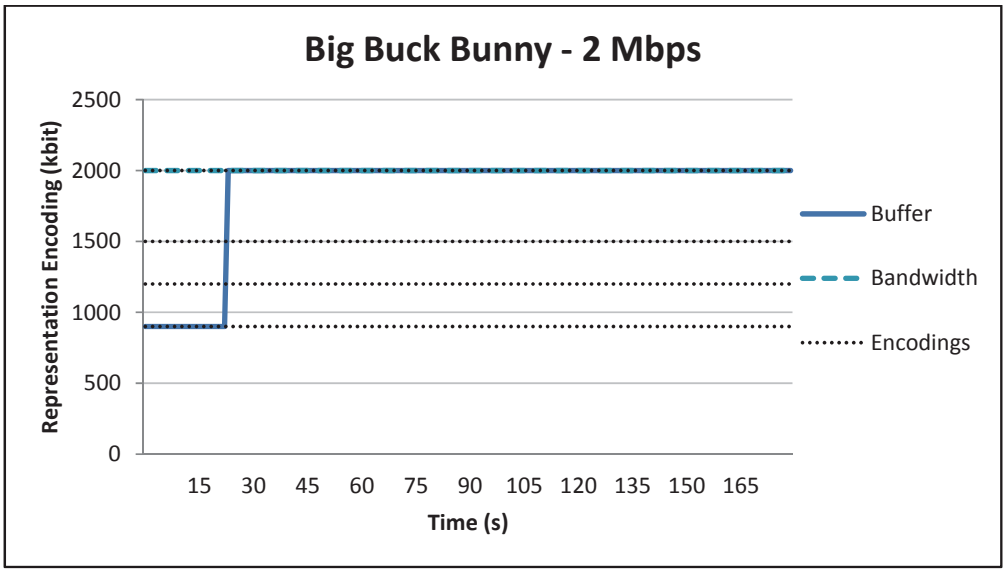


Figure 5.5- "Buffer Algorithm" streaming "Big Buck Bunny" over 2 Mbps

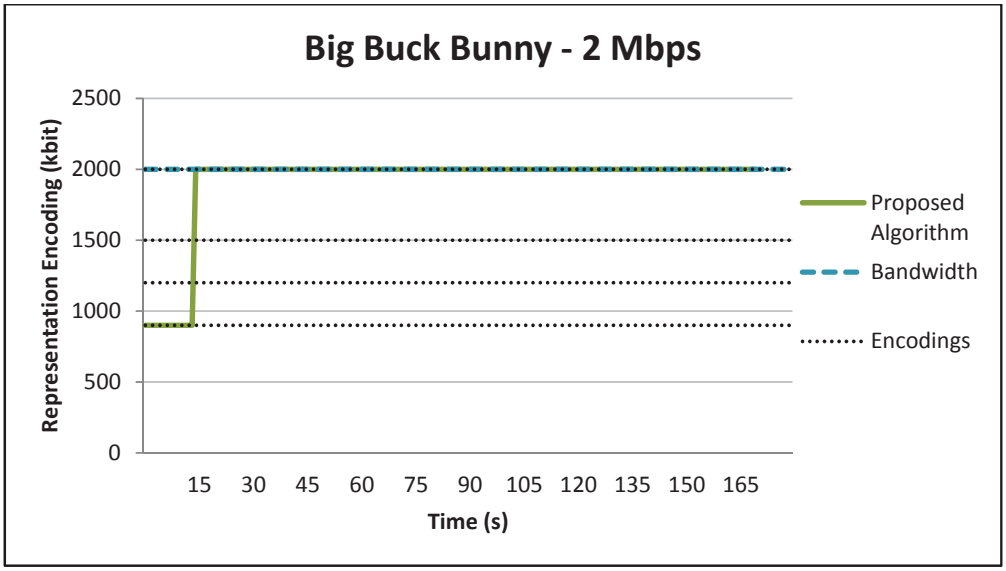


Figure 5.6 - "□roposed Algorithm" streaming "Big Buck Bunny" over 2 Mbps

Average Bandwidth

The average bandwidth emulations of the "Big Buck Bunny" were done over 1.□ Mbps. Accordingly, the packet shaper bandwidth was set to 1.□Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □□, Figure □.□and Figure □.□for the Original, Buffer and □roposed algorithm respectively.

The "Big Buck Bunny" experiments under 1.□Mbps show a similar performance for the "Original" and the "Buffer" implementations as shown in Figure □.□and Figure □.□At first the lowest encoding was selected since the buffer level was less than 30□. The "Original" approach selects the lowest encoding when the buffer level is less than 30□ and the "Buffer approach" also decreases the selected encoding by 30□ or □0□ of the measured bandwidth when the buffer level is less than 1□□ or 3□□ respectively. □ ith the increase in the buffer level, the 2 Mbps encoding was selected. Again at the □□th second, the buffer level decreased and accordingly the lowest encoding was selected. At □1st second, the 2 Mbps encoding was restored as a result of the increase in the buffer level. On the other hand, Figure □.□presents the "□roposed Algorithm" results where the gradual switching of representations is noticed starting by the □00 Kbps and moving up to the 1.□Mbps to finally reach the 2 Mbps. Consequently, the "□roposed Algorithm" provided better performance.

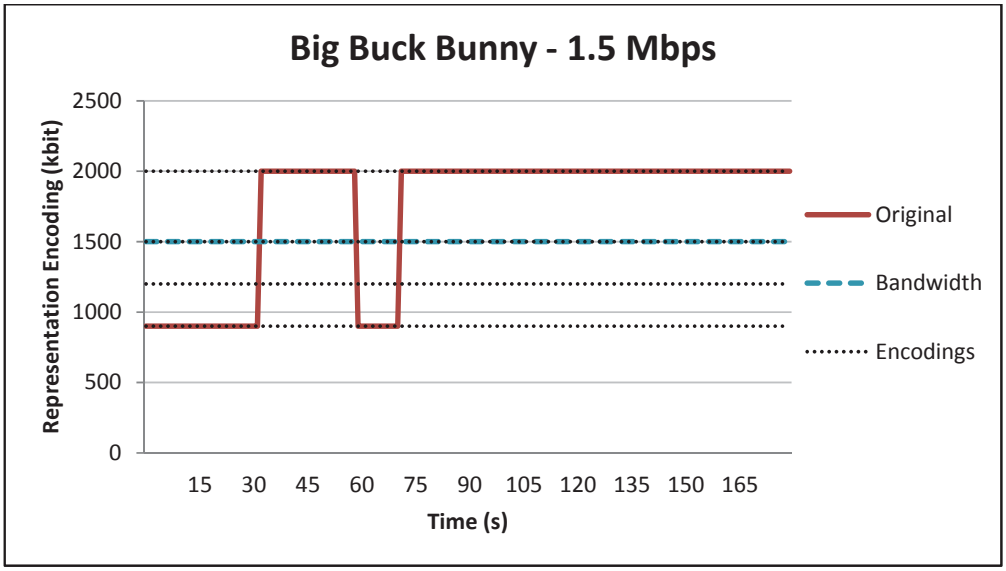


Figure 5.7 - "Original Algorithm" streaming "Big Buck Bunny" over 1.□Mbps

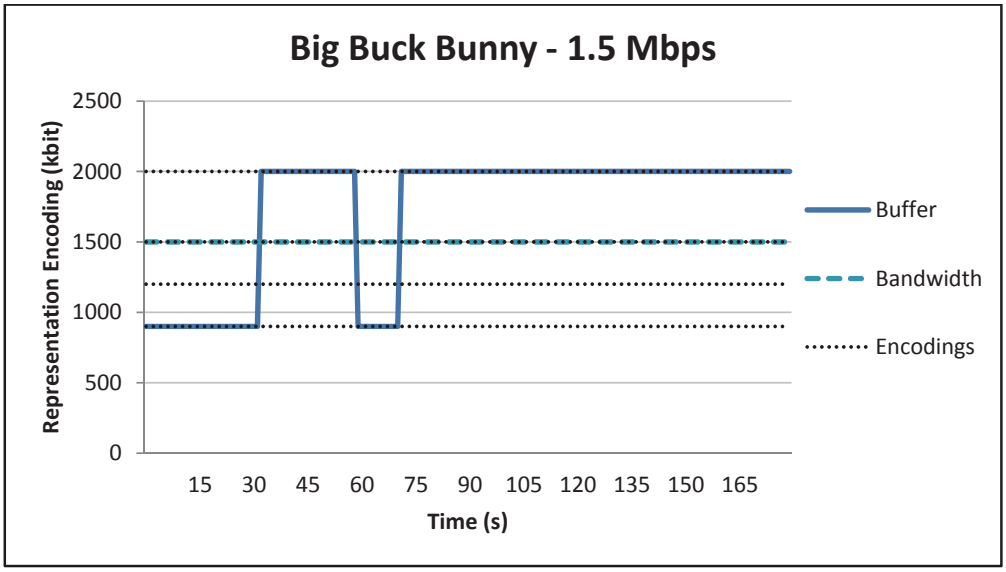


Figure 5.8 - "Buffer Algorithm" streaming "Big Buck Bunny" over 1.□Mbps

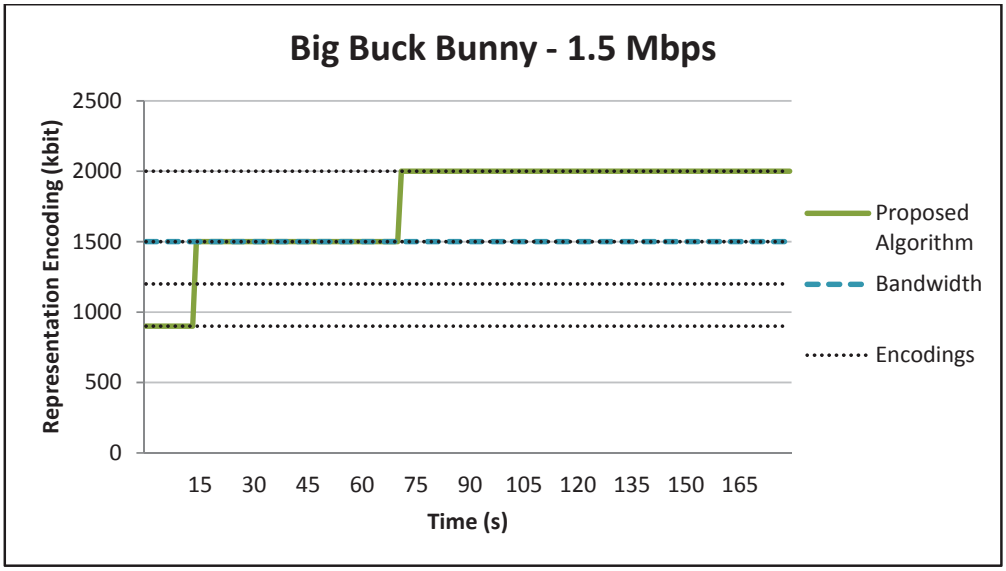


Figure 5.9 - "□roposed Algorithm" streaming "Big Buck Bunny" over 1.□Mbps

Variable Rate Bandwidth

The variable rate bandwidth emulations of the "Big Buck Bunny" were done as follows. First, the bandwidth was set to 1 Mbps for 2□seconds, and then increased to 1.□Mbps for 30 seconds and then to 2 Mbps for another 30 seconds. Afterwards, the bandwidth was decreased to 1.2 Mbps for 30 seconds and then to 0.□ Mbps for a duration of □0 seconds. Finally, the bandwidth is again increased back to 1.□Mbps. The test duration was set to 200 seconds. The results are shown in Figure □10, Figure □11 and Figure □12 for the Original, Buffer and □roposed algorithm respectively.

The "Big Buck Bunny" experiments under variable rate bandwidth show different results for the three implementations.

Figure □10, shows the results of the "Original" approach where the algorithm adapted positively to the increase in bandwidth and in a stepwise approach. However, when the bandwidth was dropped the requested encodings stayed high as a result of the high average bitrate and then dropped at the 1□□th second due to the

decrease in the buffer level below 30□. And with the increase in the bandwidth and of course the increase in the Buffer level, the selected encoding was also increased to 1.□Mbps and 2 Mbps on the 1□0th and 1□2nd second respectively.

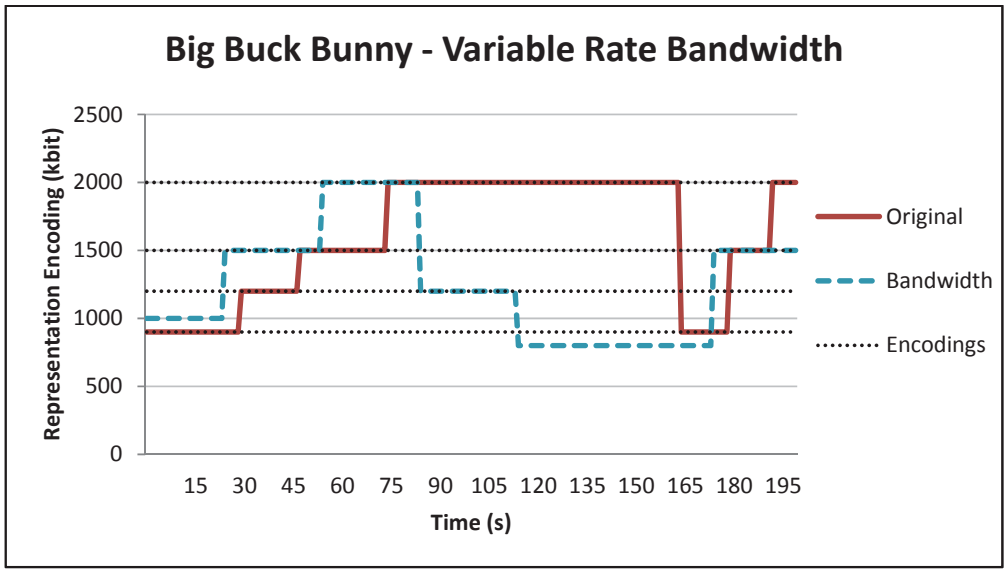


Figure 5.10 - "Original Algorithm" streaming "Big Buck Bunny" over variable rate

Figure □11, presents the results of the "Buffer" approach. This implementation does not provide a stepwise transition whether the bitrate increases or decreases. Instead, the transition is dependent on the buffer level and that is why there was a direct increase to the maximum available encoding on the □□th second and then a direct decrease to lowest encoding on the 1□□th second which reflects the status of the buffer when it was full and then when it was below 1□□.

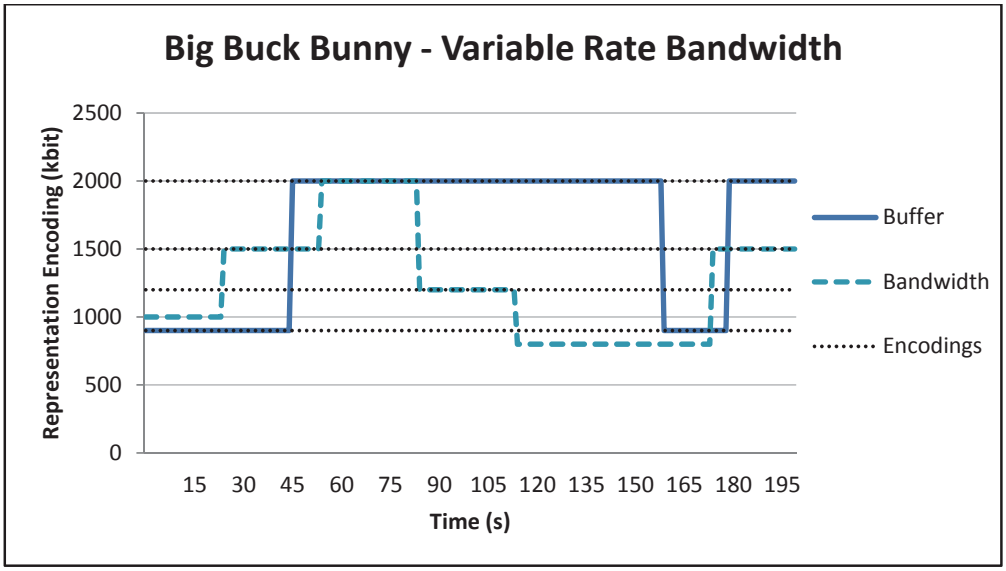


Figure 5.11 - "Buffer Algorithm" streaming "Big Buck Bunny" over variable rate

Figure □12, presents the results of the "□roposed Algorithm". The adaptation process takes place gradually on the increase and decrease in the available bitrate in a stepwise manner reflecting the underlying network conditions. The selected encoding increased from □00 Kbps to 1.□ Mbps on the □0th second and to 2 Mbps on the □□th second. Afterwards, it decreased back to 1.□ Mbps on the 121st second and to □00 Kbps on the 1□□th second to finally increase back to 1.□ Mbps with the increase in the bandwidth to 1.□ Mbps. Accordingly, the "□roposed Algorithm" provided better performance.

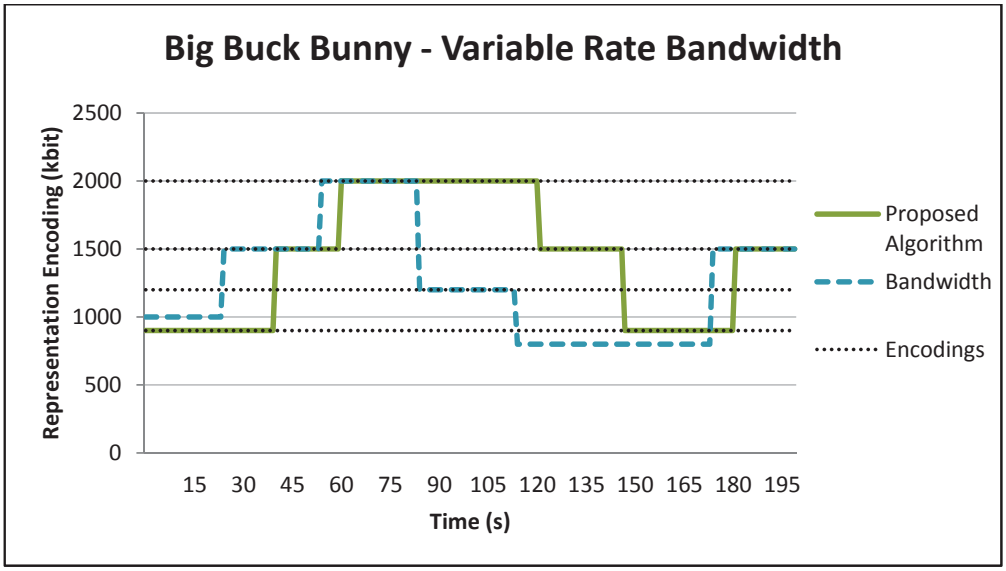


Figure 5.12 - "□roposed Algorithm" streaming "Big Buck Bunny" over variable rate

5.2 Of Forest and Men Experiments

Low Bandwidth

The low bandwidth emulations of the "Of Forest and Men" were done over 1 Mbps. Accordingly, the packet shaper bandwidth was set to 1 Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □13, Figure □1□ and Figure □1□for the Original, Buffer and □roposed algorithm respectively.

The "Of Forest and Men" experiments under 1 Mbps show again a similar performance for the "Original" and the "□roposed Algorithm" implementations with a stable selection of the □00 Kbps encoding as shown in Figure □13 and Figure □1□ However, in the "Buffer" approach, Figure □1□, the selected encoding increased to the maximum available encoding at the 1□□th second, and that is □Mbps. This is the result of having a full buffer. However, at the 1□0th second a drop again to the lowest encoding, □00 Kbps, occurred as result of having a low buffer level. Consequently, the "Original" and the "□roposed Algorithm" provided better performance

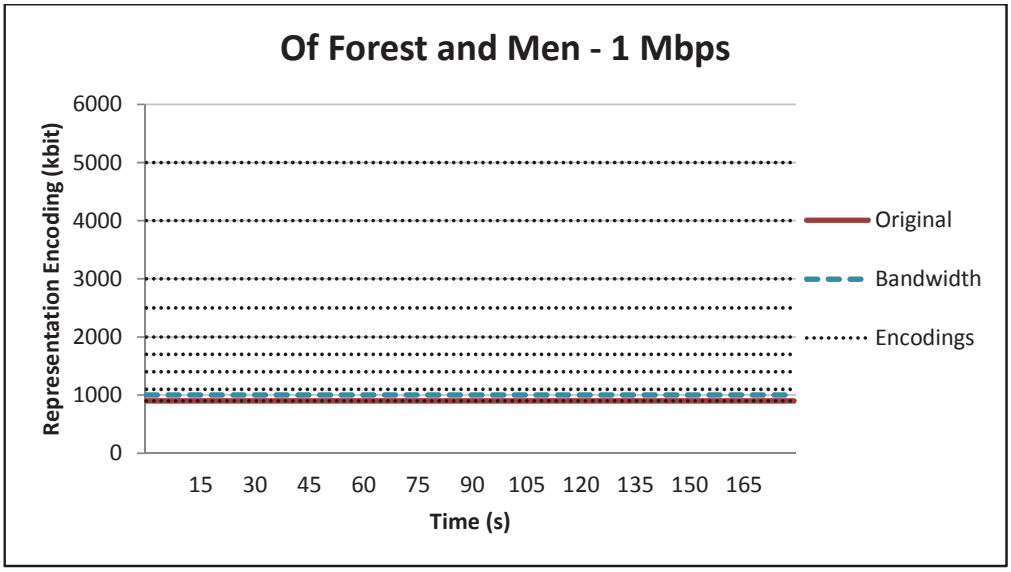


Figure 5.13 - "Original Algorithm" streaming "Of Forest and Men" over 1 Mbps

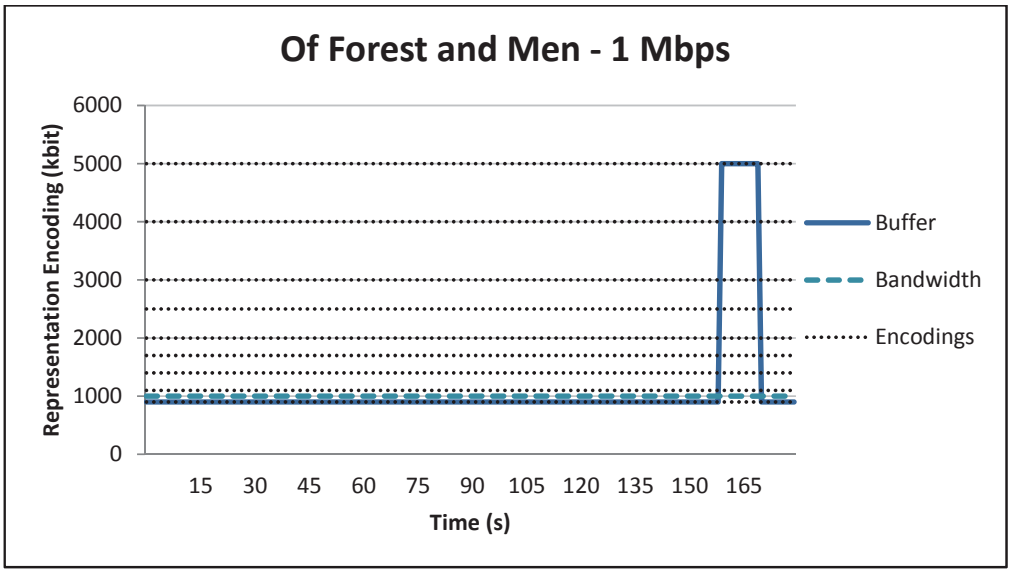


Figure 5.14- "Buffer Algorithm" streaming "Of Forest and Men" over 1 Mbps

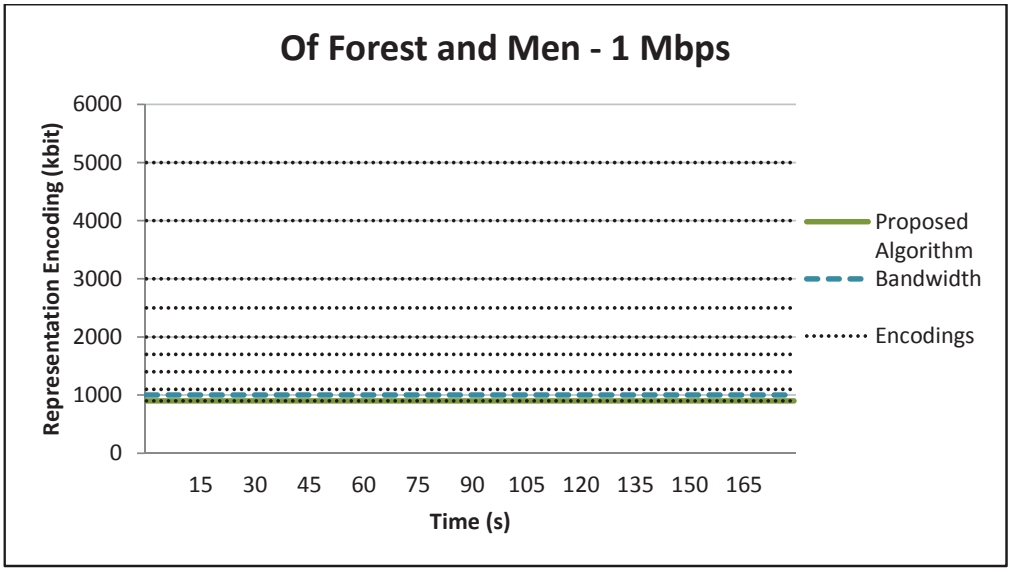


Figure 5.15 - "□roposed Algorithm" streaming "Of Forest and Men" over 1 Mbps

High Bandwidth

The high bandwidth emulations of the "Of Forest and Men" were done over □Mbps. Accordingly, the packet shaper bandwidth was set to □Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □1□, Figure □1□ and Figure □1□for the Original, Buffer and □roposed algorithm respectively.

The "Of Forest and Men" experiments under □Mbps show a similar performance for the "Original" and "Buffer" implementations as shown in Figure □1□ and Figure □1□. A drop in the selected encoding is detected to □00 Kbps at the 1□□th second in the "Original approach and to 3 Mbps at the 13□th second in the buffer approach. However, in the "Buffer approach this decrease was gradual as the drop was first to 3 Mbps and then to 2.□Mbps at the 1□0th second. This drop is of course the result of the low buffer level. □onetheless, with the increase in the buffer level, the selected encoding values were restored to □ Mbps in both approaches. On the other hand, Figure □1□presents the results of the "□roposed Algorithm" that show the stepwise move from □00 Kbps to □ Mbps and then to □ Mbps at the 22nd second. The "□roposed Algorithm" provided better performance.

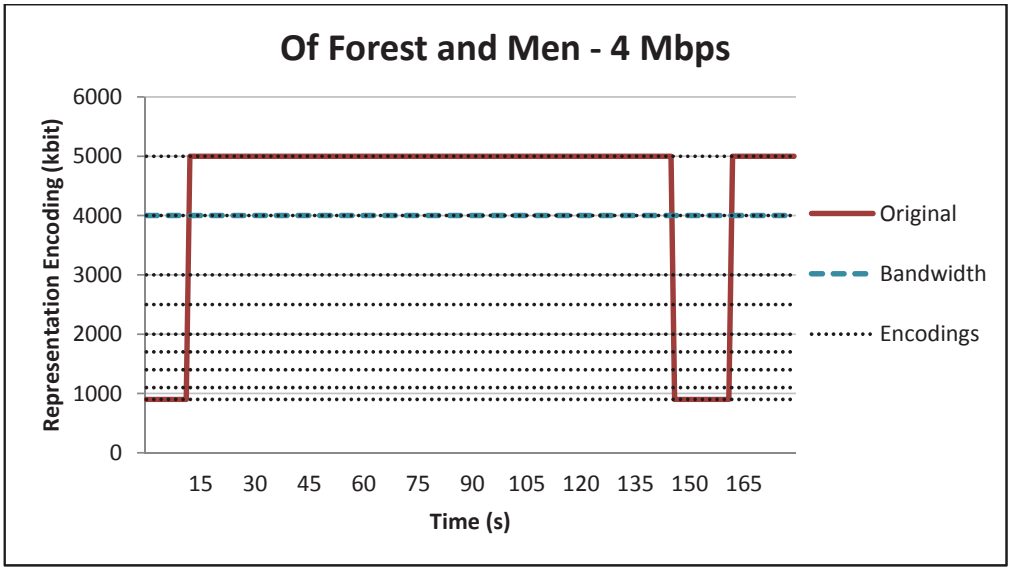


Figure 5.16 - "Original Algorithm" streaming "Of Forest and Men" over □Mbps

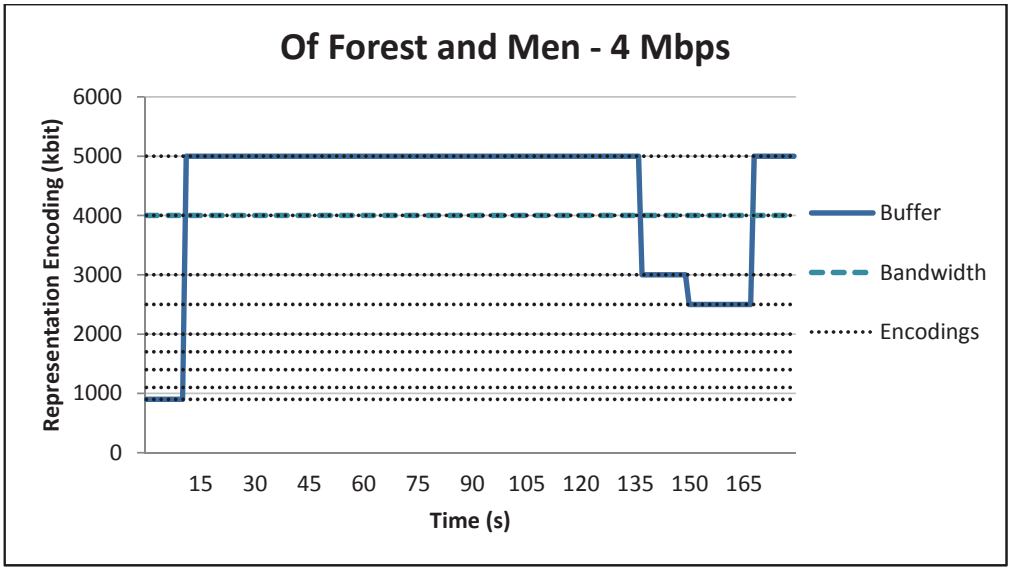


Figure 5.17 - "Buffer Algorithm" streaming "Of Forest and Men" over □Mbps

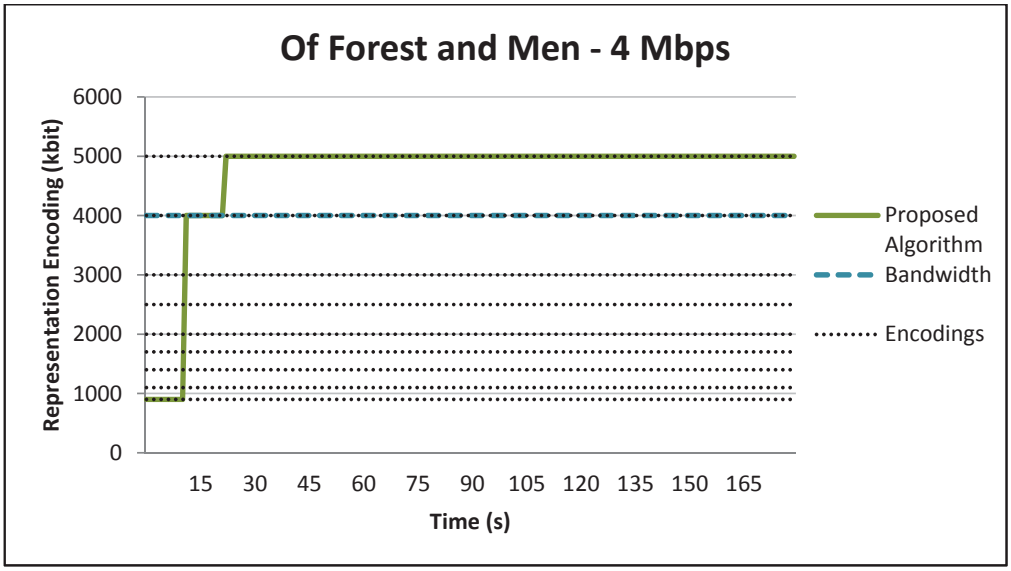


Figure 5.18 - "□roposed Algorithm" streaming "Of Forest and Men" over □Mbps

Average Bandwidth

The average bandwidth emulations of the "Of Forest and Men" were done over 2.□ Mbps. Accordingly, the packet shaper bandwidth was set to 2.□Mbps non-burstable. The test duration was set to 1□0 seconds. The results are shown in Figure □1□, Figure □20 and Figure □21 for the Original, Buffer and □roposed algorithm respectively.

The "Of Forest and Men" experiments under 2.□Mbps for both the "Original" and "Buffer" approaches show an inconsistent performance and a transition in the selected encodings as a result of the Buffer level changes as shown in Figure □1□ and □20. An abrupt drop to the lowest encoding and that is □00 Kbps can be observed twice in both methodologies. However, the "□roposed Algorithm" again provided gradual transition from □00 Kbps to 2.□Mbps and to reach finally 3 Mbps as shown in Figure □21. The "□roposed Algorithm" provided better performance.

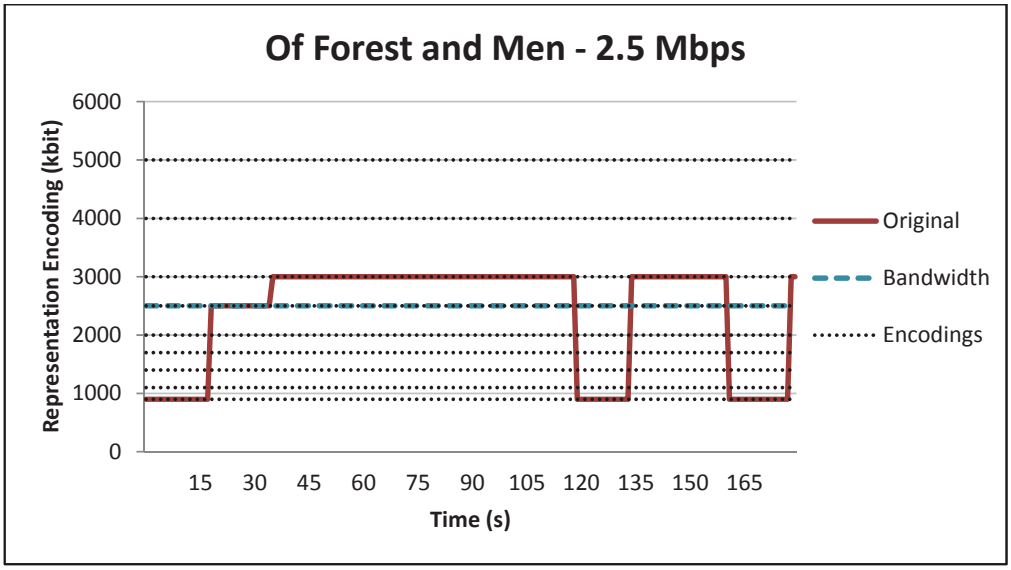


Figure 5.19 - "Original Algorithm" streaming "Of Forest and Men" over 2.□Mbps

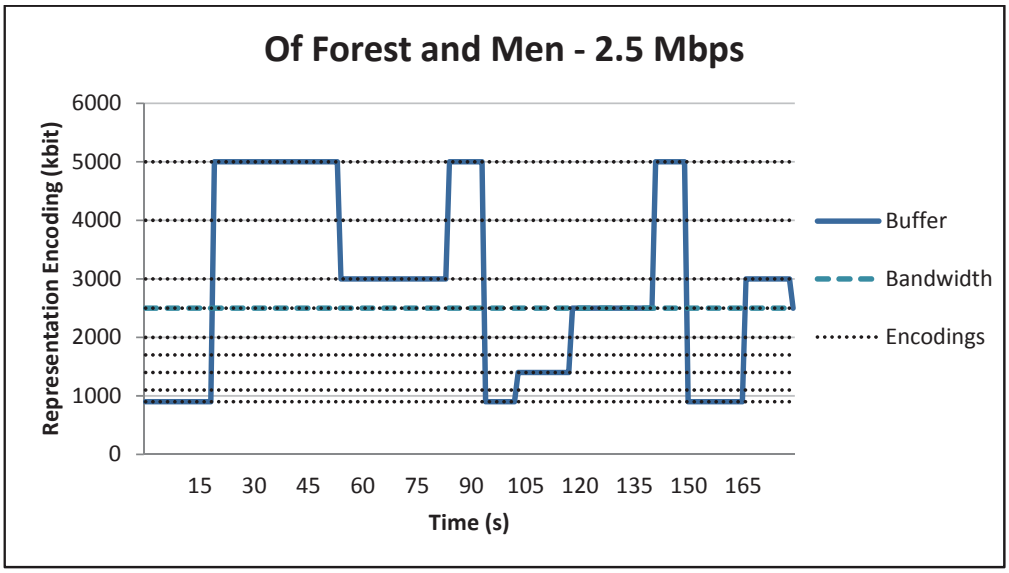


Figure 5.20 - "Buffer Algorithm" streaming "Of Forest and Men" over 2.□Mbps

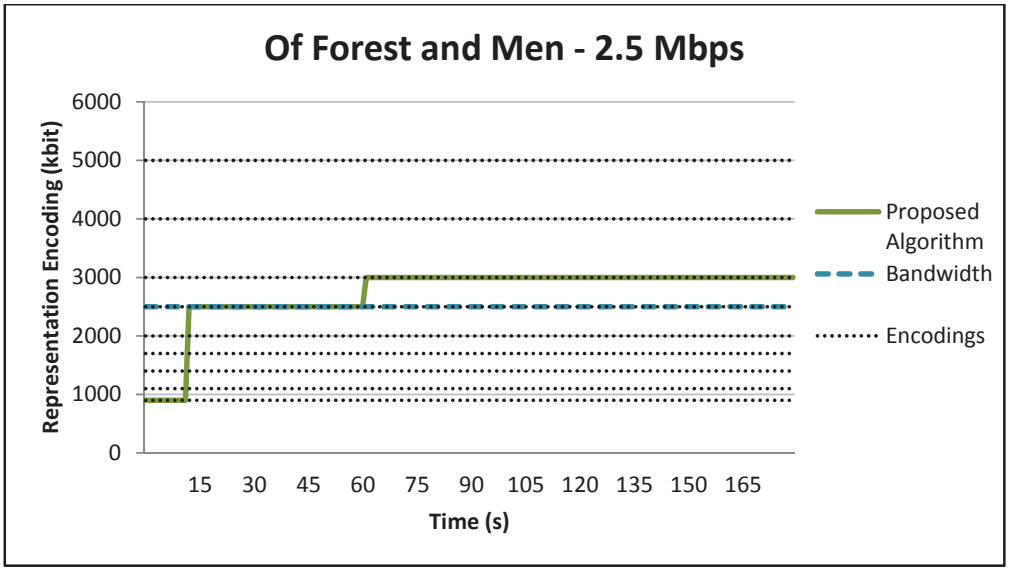


Figure 5.21 - "□roposed Algorithm" streaming "Of Forest and Men" over 2.□Mbps

Variable Rate Bandwidth

The variable rate bandwidth emulations of the "Of Forest and Men" were done as follows. First, the bandwidth was set to 2 Mbps for 2□seconds, and then increased to 3 Mbps for 30 seconds and then to □Mbps for another 30 seconds. Afterwards, the bandwidth was decreased to 2 Mbps for 30 seconds and then to 1 Mbps for a duration of □0 seconds. Finally, the bandwidth is again increased back to 3 Mbps. The test duration was set to 200 seconds. The results are shown in Figure □22, Figure □23 and Figure □2□ for the Original, Buffer and □roposed algorithm respectively.

The "Of Forest and Men" experiments under variable rate bandwidth show different results for the three implementations.

Figure □22, shows the results of the "Original" approach where the algorithm adapted positively to the increase in bandwidth and in a stepwise approach. However, when the bandwidth was dropped the requested encodings stayed high as a result of the high average bitrate and then dropped at the 1□□th second to the lowest

encoding level, □00 Kbps, due to the decrease in the buffer level below 30□. However, when the throughput was increased again, the buffer level also increased and accordingly the selected encoding also increased to 3 Mbps on the 1□0th second.

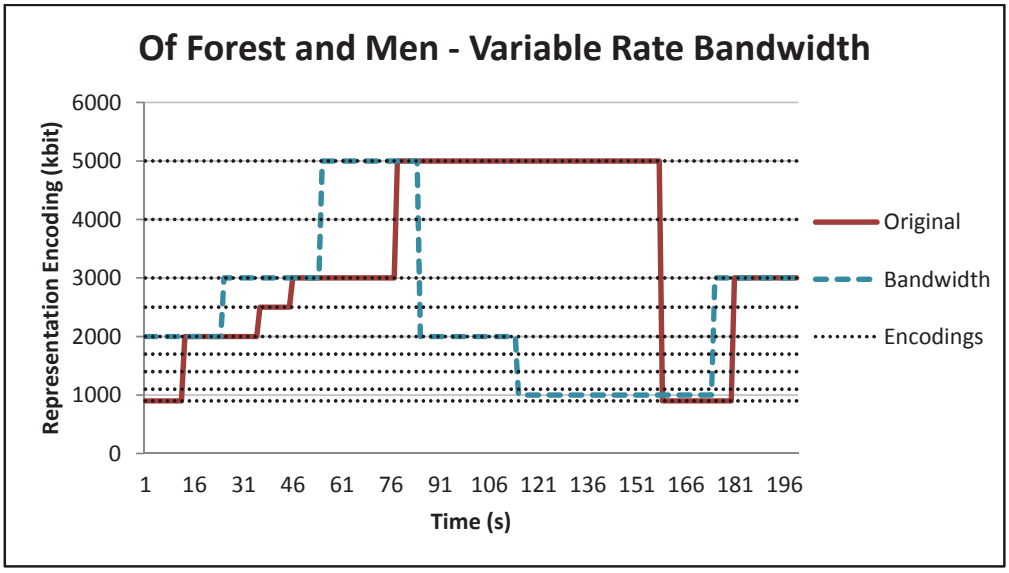


Figure 5.22 - "Original Algorithm" streaming "Of Forest and Men" over variable rate

Figure □23, presents the results of the "Buffer" algorithm. Since this implementation is dependent on the buffer level, the switching between the different encodings is independent on the available throughput and this caused an abrupt increase between the lowest and highest encodings as shown in the figure below.

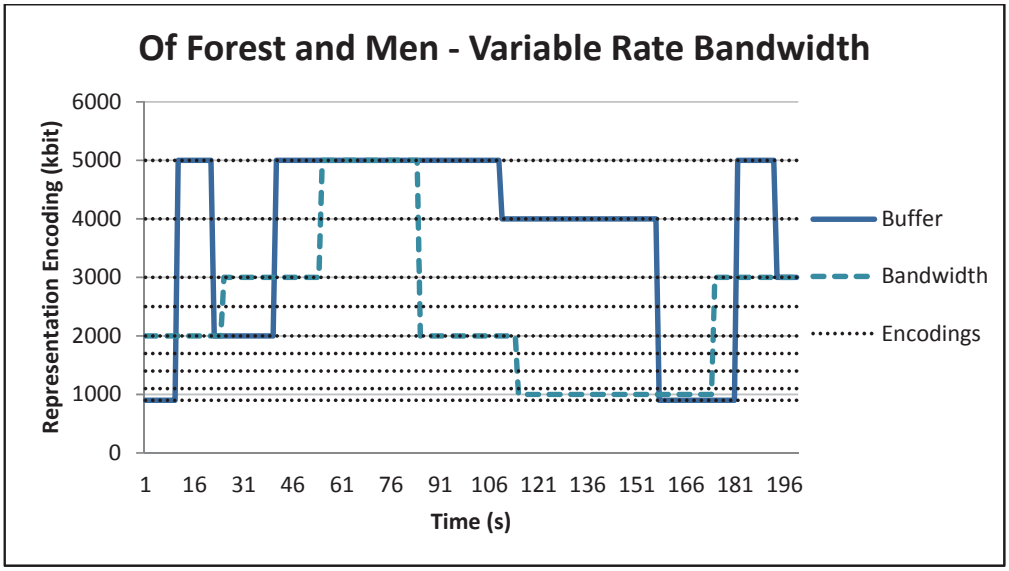


Figure 5.23 - "Buffer Algorithm" streaming "Of Forest and Men" over variable rate

Figure □2□ presents the results of the "□roposed Algorithm". The adaptation takes place gradually on the increase and decrease in the available bitrate in a stepwise manner reflecting the underlying network conditions. The selected encoding increased from □00 Kbps to 2 Mbps on the 10th second, to 3 Mbps on the □0th second and to □Mbps on the □□th second. Afterwards, it decreased back to 2.□Mbps on the 11□th second, to 1.1 Mbps on the 1□0th second and to □00 Kbps on the 1□2nd second to finally increase back to 3 Mbps with the increase in the bandwidth. Consequently, the "□roposed Algorithm" provided better performance.

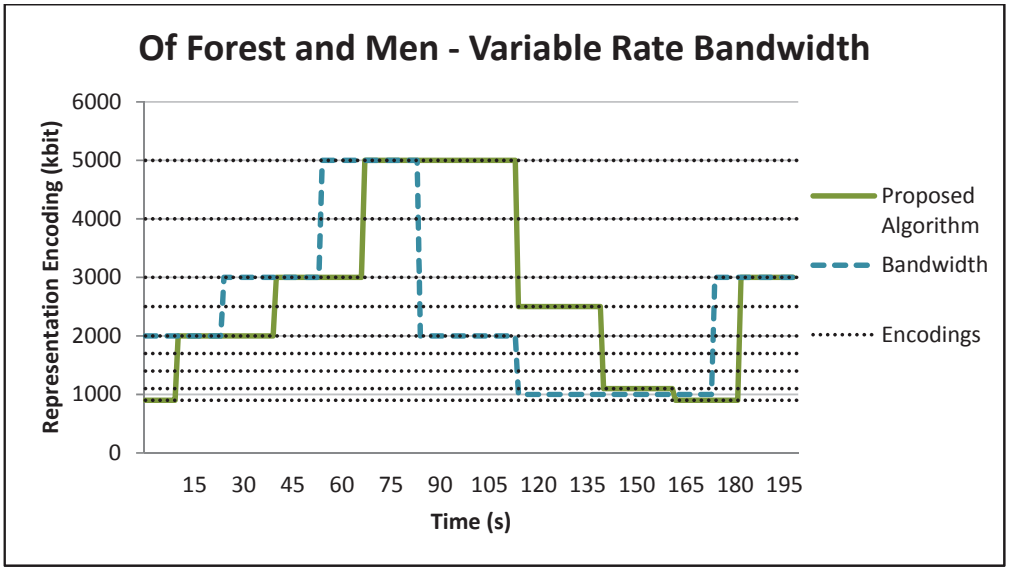


Figure 5.24 - "□roposed Algorithm" streaming "Of Forest and Men" over variable rate

CHAPTER SIX

CONCLUSIONS AND FUTURE WORK

This thesis presents a new M□E□-DASH dynamic adaptation algorithm for mobile devices that provides enhanced quality of experience. A detailed evaluation of the proposed adaptation approach in comparison to two existing M□E□-DASH implementations, under different network simulations, is provided. The results show that the proposed approach provides better results in comparison to the other methodologies. The algorithm ensures a stepwise transition between different video encodings that is consistent with the varying network and client conditions.

Moreover, while finalizing this work, a recent approach was found that presents a smooth incorporation of M□E□-DASH using HTML□video component in the □eb (□ainer, Lederer, Muller, & Timmerer, 2012). A comparison between the adaptation algorithm of this work and the proposed approach in this thesis, under variable rate bandwidth using the "Of Forest and Men" dataset, is presented in Appendix I. The comparison showed similar results for both algorithms. However, in comparison to the proposed approach in this thesis under the same network simulation, the proposed algorithm provided less number of switches and guaranteed a fast response to the variations in the network throughput. Based on these results, it can be identified that the tradeoff is between having high number of gradual switches and between making use of the available bandwidth with minimal number of switches□which is of course worth more investigation in order to decide which approach is better.

Accordingly, and as part of the future work, the focus will be more on improving the adaptation process of the proposed adaptation algorithm in order to maximize the quality of experience. This will comprise identifying the optimal number of switches to be used for video encoding transitions and at the same time to benefit the most from the available bandwidth. Moreover, this work will include integrating buffer

schemes into the proposed approach in order to guarantee the best quality of experience.

REFERENCES

Adxic, V., Kalva, H., & Furht, B. (2012, May). Optimizing video encoding for adaptive streaming over HTTP. *IEEE Transactions on Consumer Electronics*, *58*, 397-403.

Akhshabi, S., Begen, A. C., & Dovrolis, C. (2011, February). An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. *Proceedings of the Second Annual ACM Conference on Multimedia Systems*. Doi: 10.1145/1943552.1943574

Campanelli, P. (2011). The architecture of VLC media framework. *enjoyTheArchitecture*. Retrieved from http://www.enjoythearchitecture.com/vlc-architecture

Computer Information System Company. (2013). *Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017*. Retrieved from Computer Information System Company, Cisco Visual Networking Index website: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html

DASH Industry Forum. (2012*).* DASH identifiers repository. Retrieved from http://dashif.org/identifiers/

Fecheyr-Lippens, A. (2010). A review of HTTP live streaming. *Issuu*. Retrieved from http://issuu.com/andruby/docs/http_live_streaming?viewMode=magazine&mode=embed

International Organization for Standardization. (2012). *ISO/IEC 23009-1:2012: Information technology -- dynamic adaptive streaming over HTTP (DASH) -- part 1: Media presentation description and segment formats*. Retrieved from ISO website: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=57623

Lederer, S., Müller, C., & Timmerer, C. (2012). Dynamic adaptive streaming over HTTP dataset. *Proceedings of the 3rd Multimedia Systems Conference.* Doi: 10.1145/2155555.2155570

Liu, C., Bouazizi, I., & Gabbouj, M. (2011, July). Segment duration for rate adaptation of adaptive HTTP streaming. *IEEE Transactions on Consumer Electronics*. Doi: 10.1109/ICME.2011.6012094

Lohmar, T., Einarsson, T., Fröjdh, P., Gabin, F., & Kampmann, M. (2011, June). Dynamic adaptive HTTP streaming of live content. *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks.* Doi: 10.1109/WoWMoM.2011.5986186

Mok, R. K., Luo, X., Chan, E. W., & Chang, R. K. (2012, February). QDASH: A QoE-aware DASH system. *Proceedings of the 3rd Multimedia Systems Conference*.

Müller, C., Lederer, S., & Timmerer, C. (2012, February). An evaluation of dynamic adaptive streaming over http in vehicular environments. *Proceedings of the 4th Workshop on Mobile Video*. Doi: 10.1145/2151677.2151686

Müller, C., & Timmerer, C. (2011). A test-bed for the dynamic adaptive streaming over HTTP featuring session mobility. *Proceedings of the Second Annual ACM Conference on Multimedia Systems*. Doi: 10.1145/1943552.1943588

Müller, C., & Timmerer, C. (2011). A VLC media player plugin enabling dynamic adaptive streaming over HTTP. *Proceedings of the 19th ACM International Conference on Multimedia.* Doi: 10.1145/2072298.2072429

Ni, P., Eg, R., Eichhorn, A., Griwodz, C., & Halvorsen, P. (2011). Spatial flicker effect in video scaling. *2011 Third International Workshop on Quality of Multimedia Experience.* Doi: 10.1109/QoMEX.2011.6065712

Rainer, B., Lederer, S., Muller, C., & Timmerer, C. (2012, August). A seamless web integration of adaptive HTTP streaming. *2012 Proceedings of the 20th European Signal Processing Conference.* Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6333880&isnumber=6333779

Stockhammer, T. (2011a). Dynamic adaptive streaming over HTTP – Design principles and standards. Retrieved from http://web.cs.wpi.edu/~claypool/mmsys-2011/Day2-1_3GPPPDynamic.pdf

Stockhammer, T. (2011b). Dynamic adaptive streaming over HTTP - standards and design principles. *Proceedings of the Second Annual ACM Conference on Multimedia Systems*. Doi: 10.1145/1943552.1943572

VLC media player. (2013, April). In *Wikipedia, The Free Encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=VLC_media_player&oldid=567337772

Adaptive bitrate streaming. (2011, October). In *Wikipedia, The Free Encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Adaptive_bitrate_streaming&oldid=567824077

Samsung Galaxy S II. (2013, May). In *Wikipedia, The Free Encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Samsung_Galaxy_S_II&oldid=568189063

Streaming Media. (2007, December 14). In *Wikipedia, The Free Encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Streaming_Media&oldid=177931016

APPENDIX

Comparison between a recent approach and the proposed algorithm

A comparison between the adaptation algorithm of a recent approach that integrates MPEG-DASH in the web (Rainer, Lederer, Muller, & Timmerer, 2012) and the proposed algorithm in this thesis is presented below. The proposed algorithm weights were set to $w_1 = 0.35$ and $w_2 = 0.65$ for fair comparison. The experiments were implemented under variable rate bandwidth using the "Of Forest and Men" dataset similar to the experiments presented earlier. The test duration is 200 seconds.

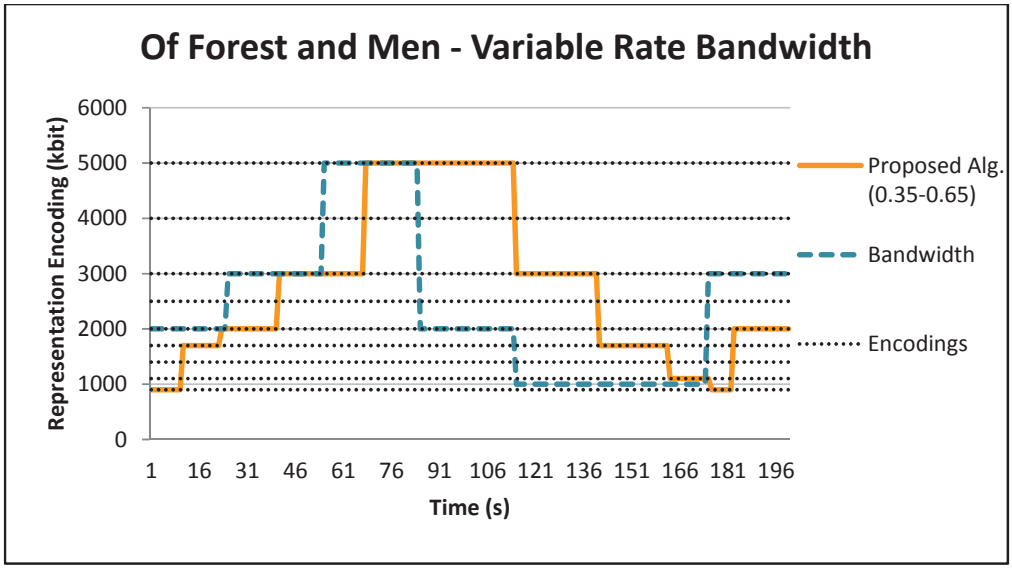


Figure 8.1 - "Proposed Algorithm (0.35-0.65)" streaming "Of Forest and Men" over variable rate

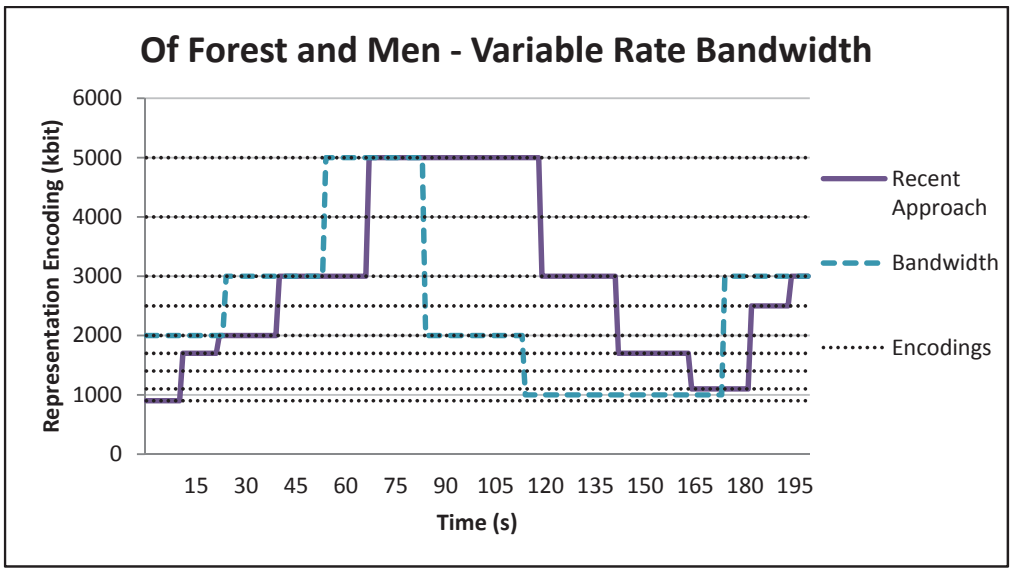


Figure 8.2 - "Recent Approach" streaming "Of Forest and Men" over variable rate