A Common Analysis Framework for Simulated Streaming-Video Networks

Submitted in fulfilment

of the requirements of the degree of

Master of Science

of Rhodes University

Patrick Mulumba

Grahamstown, South Africa

April 2009

Abstract

Distributed media streaming has been driven by the combination of improved media compression techniques and an increase in the availability of bandwidth. This increase has lead to the development of various streaming distribution engines (systems/services), which currently provide the majority of the streaming media available throughout the Internet. This study aimed to analyse a range of existing commercial and open-source streaming media distribution engines, and classify them in such a way as to define a Common Analysis Framework for Simulated Streaming-Video Networks (CAFSS-Net). This common framework was used as the basis for a simulation tool intended to aid in the development and deployment of streaming media networks and predict the performance impacts of both network configuration changes, video features (scene complexity, resolution) and general scaling.

CAFSS-Net consists of six components: the server, the client(s), the network simulator, the video publishing tools, the videos and the evaluation tool-set. Test scenarios are presented consisting of different network configurations, scales and external traffic specifications. From these test scenarios, results were obtained to determine interesting observations attained and to provide an overview of the different test specifications for this study. From these results, an analysis of the system was performed, yielding relationships between the videos, the different bandwidths, the different measurement tools and the different components of CAFSS-Net.

Based on the analysis of the results, the implications for CAFSS-Net highlighted different achievements and proposals for future work for the different components. CAFSS-Net was able to successfully integrate all of its components to evaluate the different streaming scenarios. The streaming server, client and video components accomplished their objectives.

It is noted that although the video publishing tool was able to provide the necessary compression/decompression services, proposals for the implementation of alternative compression/decompression schemes could serve as a suitable extension. The network simulator and evaluation tool-set components were also successful, but future tests (particularly in low bandwidth scenarios) are suggested in order to further improve the accuracy of the framework as a whole. CAFSS-Net is especially successful with analysing high bandwidth connections with the results being similar to those of the physical network tests.

Acknowledgements

I would like to thank my supervisors, George Wells and Peter Clayton, for their remarkable supervision in helping me manage both the content and direction of my work. I am grateful for their recommendations, their guidance and being able to bounce ideas back and forth between them. I am humbled by their constant communication (even across international waters) and timely response to any queries and suggestions that I had and assisting me to become a more independent researcher.

I would like to thank my family, particularly my Mom and Dad, who provided an extra set of eyes on my work. For finding the time to proof read and smoothing out my grammatical errors. I will always remember your continuous motivation (even through difficult times) and I will forever appreciate the opportunity that you given me to produce this thesis.

I would like to thank my fellow colleagues from the Convergence group, who spurred me on with their kind counsel and the weekly meetings that broadened my computer science knowledge. Friends from the Masters Lab, I would like to thank you for your kind suggestions and efficient coding scripts that assisted me in foreign computing environments.

I would also like to thank the members of staff and the rest of the Computer Science department in providing a comfortable and stimulating environment to work in and finally Rhodes University, for where I have spent many great years, I thank you.

The names of various products and systems referred to in this thesis are registered trademarks and trademarks of the companies that have developed them. These trademarks are duly acknowledged.

This work was partly sponsored by the Distributed Multimedia Centre of Excellence at Rhodes University, with sponsorship from Telkom SA, Business Connection, Comverse, Amatole Telecommunication, Mars Technologies, Openvoice, Storetech, Tellabs, THRIP and with financial support from the National Research Foundation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Definition

Distributed multimedia comprises of various distribution engines which supply streaming services across the Internet through different networks [1]. The majority of these services depend on the distribution of video multimedia [2]. Unfortunately, there are situations where many of these streams are transmitted through an assortment of various network bandwidths and networking configurations which can have an adverse effect on the quality of the video stream. Even though most of these conditions may be network specific, some of these difficulties arise due to the nature of the source material (such as the size and the quality of the original source stream). We propose a framework for streaming media which assists in the prediction of performance impacts of various changes to an existing streaming media network. These performance impacts range from modifications in network design to the different video characteristics of the original source stream.

1.2 Motivation

The development of network communication infrastructure has aided the expansion of present video streaming media [3]. However, there is still a considerable portion of the

Internet that has not yet developed the necessary means to conform to this modern network development.

Video streaming media is significantly dependent on the characteristics of the communication medium [4]. Seeing that bandwidth is one of the major determinants of the quality of the media stream [5–8], various streaming optimisation strategies and techniques have been applied by existing streaming media frameworks to ensure successful video stream transmission. Some of these optimisation strategies include utilising video compression techniques, applying caching or buffer strategies, editing the source media to a more economic format and simplifying network dynamics.

However, some of these strategies or techniques restrict any drastic changes to the overall design of the delivery framework. Therefore, we propose a simulation tool to analyse and predict the results of transmitting a video stream through different network configurations. This is an analytical system which will serve as a guideline to aid in the design of video streaming media environments. It is designed to predict the performance of video streaming in relation to different scaled network sizes. This framework has been called "Common Analysis Framework for Simulated Streaming-Video Networks", abbreviated as CAFSS-Net.

1.3 Scope

This study seeks to view a range of existing commercial and open-source streaming media distribution systems/services. The common features derived from these frameworks in relation to media streaming will be to used as a foundation for CAFSS-Net**.** This is to validate the simulation tool against empirical evidence collected from both configuration and engineering extensions to existing systems.

This research investigates various open-source and commercial streaming media distribution engines (systems/services). However, it does not intend to alter or make any changes to these distribution engines (systems/services), but it does intend to identify useful features or concepts that are applicable to CAFSS-Net. The research will use a suitable network simulation tool to be the foundation of the study and to develop the framework accordingly.

Various caching strategies/algorithms are available for video streaming, but CAFSS-Net will only implement a simplified caching solution. Though various video compression/encoding algorithms are available, the research will only investigate a single encoding format, so as to limit the scope of the project. To get a controlled collection of data, CAFSS-Net will be using a specific number of video files as source streams (and not live videos) to provide an accurate comparison between the other streaming distribution systems/services and CAFSS-Net.

1.4 Outline of the Thesis

Chapter 2

This chapter reviews the literature covered within this study. It starts by presenting multimedia streaming along with its advantages, disadvantages and requirements. Section 2.3 introduces some of the common network, video and streaming terminology used throughout this study. This section also describes how streaming media is classified and transmitted. These next two sections review the literature showing how transportation occurs over networks in section 2.4 (subdivided into multicasting and Peer-to-Peer implementation) followed by algorithms and strategies in section 2.5 (subdivided into content distribution, proxy and caching).

Section 2.6 introduces the three existing streaming media distribution systems/services that will be used to provide a foundation for the framework by identifying their common elements. The next section introduces network simulation and provides brief descriptions of existing simulators. The last section (section 2.8) presents the video characteristics relating to this study which include the types of videos, the types of frames and the common video resolutions.

Chapter 3

The methodology chapter starts by evaluating the existing streaming media distribution systems/services reviewed from the previous chapter. Section 3.3 defines the Common Analysis Framework for Simulated Streaming-Video Networks (CAFSS-Net). A short overview of the six components (which fall within three major components) are presented which consist of: the videos, the video publishing tools, the streaming server, the network simulator, the client(s) and the evaluation tool-set. The details of the three major components are presented and are composed of: The network simulator in subsection 3.3.3 (based on the NS-2 network simulator), the evaluation tool-set in subsection 3.3.4 (based on the Evalvid tool-set) and the videos utilised through CAFSS-Net in subsection 3.3.5.

The test bed is presented in section 3.4 which consists of three scenarios. The first scenario (the Peer-to-Peer scenario in subsection 3.4.1) consists of a client-server architecture and is designed to test and evaluate the videos through the simulated network over different bandwidth speeds. The second scenario (the Small Network scenario in subsection 3.4.2) consists of three sets of tests: the physical networking test, the cache comparison test and the node placement test. The last scenario (the Large Network scenario in subsection 3.4.3) introduces external network traffic into the simulated networking environment.

Chapter 4

The results chapter begins by providing details about the data collection process. It introduces TRACE_FILE's which are a principal output for CAFSS-Net's evaluation component and this chapter also shows how some of the various evaluation tools evaluate these files to produce the results. The Peer-to-Peer scenario test results are presented in section 4.3, followed by the results for the Small Network scenario (in section 4.4) and the results for the Large Network scenario in section 4.5. The results are presented graphically with the short descriptions detailing the measurements obtained and the interesting results. This chapter concludes with an overview (in section 4.7) of the test results gathered throughout these test scenarios.

Chapter 5

The discussion chapter first provides the limitations of CAFSS-Net related to the video and network characteristics of the framework. Section 5.3 presents the analysis of the results obtained with reference to the overview section from the previous chapter (section 4.7). This analysis presents the interesting observations and irregularities identified from the results within three bandwidth discussions: the low bandwidth discussion (in subsection 5.3.1.1), the medium bandwidth discussion (in subsection 5.3.1.2) and the high bandwidth discussion (in subsection 5.3.1.3). This analysis then analyses the evaluation tools (subsection 5.3.2) to determine and present the relationships between them. This section also shows that different tools can be combined to further evaluate the result of the video stream transmission.

The implications for CAFSS-Net as a result of the analysis performed is discussed in section 5.4. In this discussion, each of CAFSS-Net's components are evaluated based on both the limitations defined earlier in section 5.2 and the analysis in section 5.3. Proposals for future work and possible extensions are presented where necessary for each of the components within this evaluation.

Chapter 6

The conclusion of this research is presented in this chapter and it begins by first defining the objectives of the study. Section 6.3 identifies the similarities between CAFSS-Net and the existing streaming media distribution systems/services, showing how CAFSS-Net is able to emulate the essential characteristics of these existing systems. The proposals for future work (obtained from the previous chapter in section 5.4) are presented within section 6.4. This is followed by the possible extensions in section 6.5 (obtained from the same analysis section 5.4) and the possible applications for CAFSS-Net in section 6.6. This study concludes by reviewing the accomplishments of this study in relation to the goals of CAFSS-Net (section 6.7), with reference to the limitations discovered as a result of the findings.

Chapter 2

Literature Review

2.1 Introduction

In order to gain an understanding of our intended research field, this chapter provides an overview of the literature on streaming media. Section 2.2 defines the concept of streaming, its advantages/disadvantages and also the general requirements needed to manage a media stream. The next section 2.3 outlines the common networking, video and streaming terminology that will be referenced throughout this document.

The next two sections highlight the related work in the streaming media field. The network transportation section 2.4 summarises the transportation and distribution of the media stream with specific reference to both Peer-to-Peer networks and Multicasting networks. Section 2.5 introduces some of the algorithms and strategies for streaming media with reference to content distribution, caching and a proxy.

Section 2.6 provides an overview of the existing streaming media systems/services. This section focuses on three common distributions: Microsoft Streaming Services, Apple Darwin Streaming Service and RealNetworks Helix Server/Proxy systems. The next section 2.7 introduces the concept of network simulation with some of its advantages and disadvantages. This section also provides examples with short summaries of existing network simulators. This chapter concludes with the section 2.8 which presents further literature on video and provides the information on the following: video types, resolutions/aspect ratios and video frame types.

2.2 Multimedia Streaming

The Internet has become an expanding communication tool, with various formats of media available. With the ever increasing demand for efficient means of communication, both audio and video are becoming primary sources of relaying information to peers, the public, and other recipients. Traditional downloading was the common form of distributing this media. However, with the increasing requirements for easy content control, provision of real-time video/audio and prevention of piracy, an alternative means of distribution was needed to get this media to the public.

Streaming media is a suitable alternative to traditional downloading. It enables real-time access to media via the Internet or on an intranet [9]. It is currently the most popular form of providing live broadcasts on the Internet [10]. Amongst the given advantages, no file is downloaded during the broadcast. In order for streaming to take place, the media is generally played by client software. In almost all streaming cases, a media server is required to provide the streaming [10]. Below are the common advantages and disadvantages of streaming media.

Advantages

- No need to wait for the complete file to be downloaded [9]
- No information is stored to the clients' local disc [10]
- Provides a sense of copyright protection [11]
- Excellent for live broadcasts [10]
- Adapts to different connection speeds depending on the provider [12]

Disadvantages

- Internet is not designed for real-time streaming
- Bandwidth constraints [10] [13]
- Transmission latency [12]
- Packet loss and noise [7]

Requirements

Establishing a basic stream requires certain software, hardware, and specifications. The following are the essentials needed to provide a media stream:

- **Video Encoder(s)** convert the source content into streaming media format [10]
- **Stream Server(s)** transmit the streaming media content after it has been encoded for the audience. Multiple servers are generally a good practise, as they provide redundancy and serve as backups [9]
- **Players** are used by clients to view the streaming media content. Different players allow different formats to be viewed and depending on the source type of the original stream, the players can offer full control of the stream (play, stop, fast-forward, and rewind). Some popular players are Real Player [14], Quicktime [15] and Windows Media Player [16].
- **Authoring** is the intermediate process between the stream servers and the players. This is the section where the presentation of the media stream is decided upon, be it simple links on a web-page or a complete front-end with an extensive interface [10].

2.3 Terminology

2.3.1 Network Terminology

The distribution of multimedia can not exist without the presence of some networking infrastructure, be it a large network or a simple connection. The following terminology is needed to further understand the fundamental concepts of networking within the streaming media context:

Cache - A local repository of data used to decrease the "cost" of retrieving the same information again [17]

Packet - A single unit representation of network data used to transmit information across the network

Latency - The end to end delay of a packet transmission (also the time taken for a packet to get from the source to the recipient) [18]

Node - A network device that can both receive or transmit network packets [19]

Bandwidth - The total amount of data that can be transmitted between two network nodes in a given time

Jitter - The fluctuation of delay from packet to packet (It is also the out-of-time packets/frames, decoded at the same rate) [20]

Peer-to-Peer - With this networking configuration, there is no server, and computers simply connect with each other [21]

Client-Server - Similar to peer-to-peer with emphasis set on one node functioning as a server [21]

Topology - The physical layout of a given network [22]

Routing - The process of defining the path that the packets will travel through the network [10]

Broadcasting - Transmitting the same network signal to all nodes on the network [10]

Multicasting - An efficient means of broadcasting on the network by using a multicast address [10]

Broadband - Large bandwidth sizes that allow for smooth network operation [23]

Proxy - A filter or service that forwards requests from a network device to other network devices on the network [24]

2.3.2 Video Terminology

Even though this research investigates streaming video media, it is important to first represent the concepts behind standard video before we explore the advanced streaming concepts. The following terminology represents the basic video terms:

Bit Rate - For each unit of time, the total number of bits that can be processed within that time frame [25]

CBR - **C**ontrolled **B**it **R**ate where each unit of time is constant (controlled)

Compression - The process of decreasing the size of a video file, by utilising an appropriate algorithm that matches the required compressed specification [26]

Decompression - The process of increasing the size of a video file, by utilising an appropriate algorithm that matches the required decompressed specification

Codec - (**co**mpress or **dec**ompress) Allows the playback of the compressed video file within the environment of the operating system's media player [27]

Resolution - The number of picture elements that can be displayed on the horizontal and vertical planes. (eg: 640X480) [28]

2.3.3 Streaming Terminology

Video streaming is the process of delivering a source video as a network stream through a network. The following describes the basic terminology used in video streaming. More detailed concepts will be provided at a later stage, but for now we describe the basic terms for streaming within this thesis:

Buffer - The temporary storage of video data until there is sufficient information to produce the video stream [29]

Transcoding - The express conversion from one digital codec/format (occasionally lossy) to another digital codec/format [30]

Videoconferencing - The collection of video equipment that facilitates multiple locations to interact via two-way or multi-way video communication [31]

Webcast - A streaming format which consists of a live broadcast which is transmitted over the World Wide Web [32]

2.3.4 Streaming Media General Classification

Streaming media can be classified according to three different schemes. Each scheme has benefits and problems for both the client(s) and the server(s). The three common forms are:

- **Live broadcast** across the Internet involves encoding content straight from an event 'live' to a server to be streamed [10]. However, depending on the scale of the event, more servers would alleviate the stress on a single server and at the same time provide some redundancy as backup.
- **On-demand** files are streamed files which are encoded such that they can be accessed at a later stage. The advantage of on-demand content is that access is granted simply by applying links to the content. The same file can be used by multiple clients with full playback control (play, stop, fast-forward, and rewind) [10].

- **Progressive download** consists of a mixture of traditional downloading and streaming. It enables content to be viewed whilst being downloaded, by utilising temporary memory as a buffer. This streaming method is generally used in conditions where the content is short. However, this style is sometimes not perceived as streaming [10].

2.3.5 Streaming Media Transmission Methods

Streaming media is also classified by the format in which the stream is distributed, be it in a simple one to one manner or to a wider audience. According to the *Streaming Media Bible* [10], streaming media is delivered in three common transmission methods which are unicasting, broadcasting and multicasting.

Unicasting

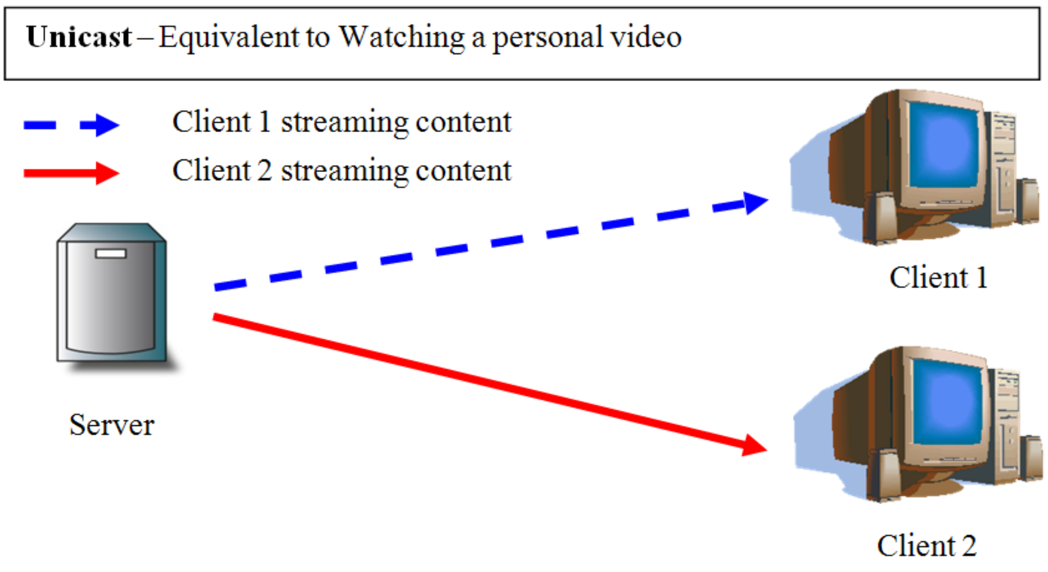

Unicast – Equivalent to Watching a personal video
Client 1 streaming content
Client 2 streaming content
Server
Client 1
Client 2

Figure 2.1: Unicasting: [10]

Figure 2.1 shows unicasting whereby each client has his/her own personal connection to the server. Some systems allow the option to manipulate playback (play, pause, rewind, next, previous).

Broadcasting

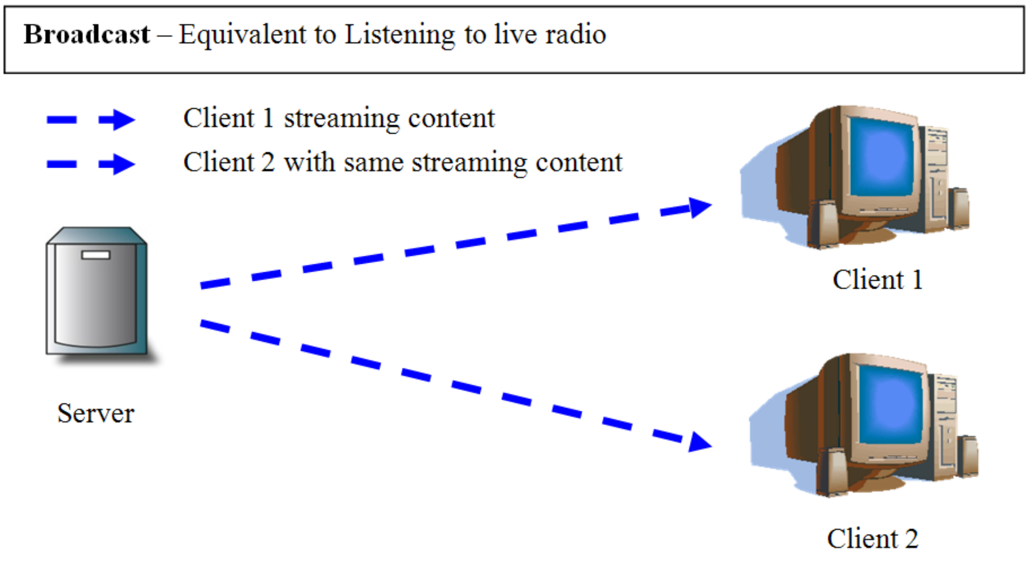

Broadcast – Equivalent to Listening to live radio
Client 1 streaming content
Client 2 with same streaming content
Server
Client 1
Client 2

Figure 2.2: Broadcasting: [10]

Broadcasting is the common streaming scenario, where the stream can not be manipulated. The server sends out a single stream which all the clients can connect to. This is illustrated in Figure 2.2.

Multicasting

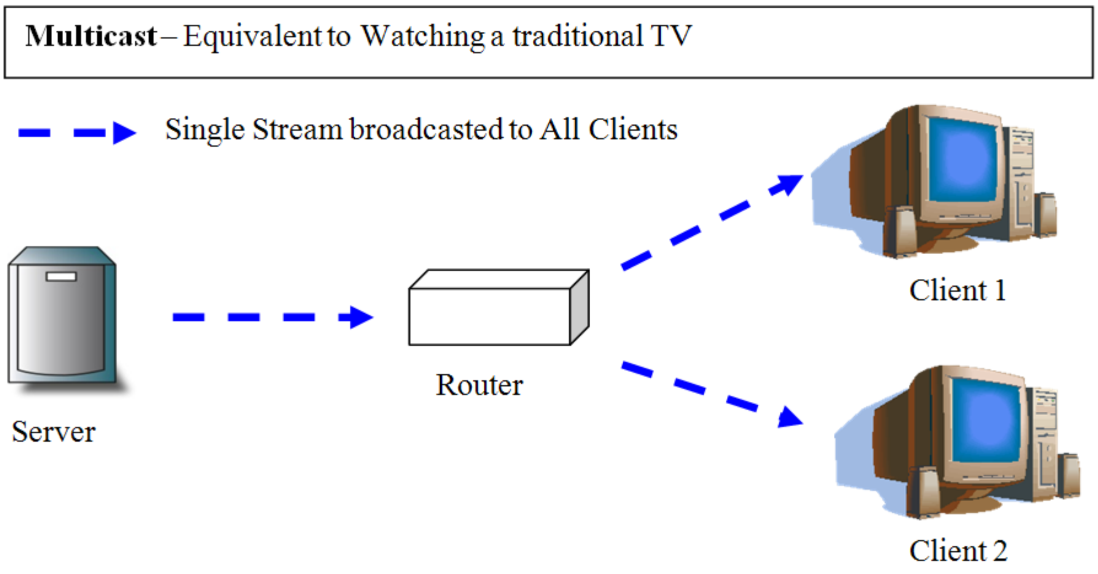

Multicast – Equivalent to Watching a traditional TV
Single Stream broadcasted to All Clients
Server
Router
Client 1
Client 2

Figure 2.3: Multicasting: [10]

The third transmission method is a more efficient form of broadcasting and is referred to as multicasting. In this form of transmission, a single stream is sent through a special multicast IP-address [10] onto a multicast network. Any of the clients on the network who join the broadcast can view this transmission stream. Another important factor is that in the multicast scenario, only a single stream is sent by the server to the router(s).

On the contrary, in the broadcast scenario each client has a direct connection to the server (flooding the network). The arrangement is shown in the Figure 2.3. Another modern form streaming, similar to multicasting, is called relaying and illustrated in Figure 2.4.

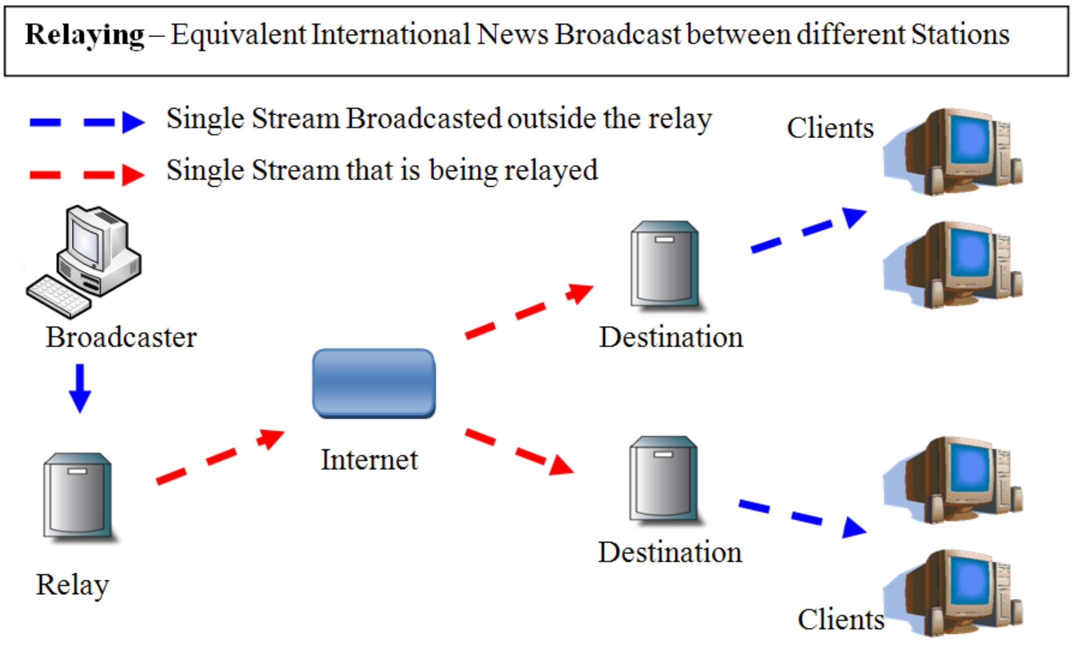

Relaying – Equivalent International News Broadcast between different Stations
Single Stream Broadcasted outside the relay
Single Stream that is being relayed
Clients
Broadcaster
Destination
Internet
Relay
Destination
Clients

Figure 2.4: Relaying: [15]

2.4 Network Transportation

This section reviews the literature related to the two main network transportation concepts expressed throughout this thesis. Even though both multicasting and peer-to-peer terms have been defined in the previous section 2.3, this section provides previous and existing applications of mulitcasting and peer-to-peer concepts which can aid with the development of CAFSS-Net.

2.4.1 Multicasting

Chu et al. provides a report on the deployment of an “operational Internet broadcast system founded on Overlay Multicast” [33]. It demonstrates that the system has been running as a suitable “cost-effective” solution for Internet broadcasts and has been issued to more than 3000 users across the globe in various environments. *Chu et al.* details the various challenges involved in setting up such a large system [33], particularly issues that are not clearly visible during the design stage. An important factor that needs mentioning is that “deployment of IP Multicast” over the Internet [33], can only take place if every router used in the transmission has IP multicast enabled [10]. This research paper [33] provides a unique perspective that will provide great insight into the development of a complete system generated from the design stage to real world application, coupled with a testing scheme which includes a benchmark.

This research system demonstrates its real world deployment in which it has been “used in more than 20 real broadcasting events” such as video conferences, meetings, sports. *Chu et al.* provides a table which represents a summary of the major broadcasts as well as the data used for the suitable benchmark [33]. *Chu et al.* presents a controlled design that demonstrates the various obstacles and decisions made to produce a successful streaming system based on Overlay Multicast [33]. A parallel concept has been developed for commercial use in respect to IPTV and can also demonstrate the similar multicasting concepts discussed earlier but to a greater scale. It exhibits a "benchmarking scheme that scales to one million subscribers" [34].

Kellerer et al. presented a similar benchmarking paper [1] which incorporated a complete test bed designed to provide a “real-time LAN-based” investigation into the behaviour of streaming media applications. This test bed was developed using Linux Divert Sockets which allow for “IP-Packet interception and packet injection” into systems within a network. The investigation looked into the performance of commercial applications on both wire-line and wireless networks. This test bed examined the flow control performance of modern streaming implementations over wireless networks, and confirmed that “flow control algorithms that view lost packets as network congestion in wire-line” are not well adapted for the wireless networks.

Krassic focuses on a general design strategy for streaming media applications in an environment of best-effort computing and networking [35]. This research paper is entitled "A Framework for Quality-Adaptive Media Streaming: Encode once, Stream everywhere". This is accomplished by applying an algorithmic process known as "adapting by Priority drop", (commonly known as adaptation by priority data dropping) [36].

Priority drop is best evaluated through "video and networking domains". Within the video domain, the "common compression formats can be expanded to support priority drop, which in turn makes them more friendly for streaming. For the networking domain, the author presents a "real-time best-effort streaming protocol" which in its basic form provides classic "unicast streaming for video on demand style applications" [35]. There is also a more advanced version of this protocol, which provides "efficient broadcast style streaming, through a multicast overlay" and exhibits multi-rate properties.

The research from this paper shows that priority drop provides a powerful source of streaming, where a single video source can be streamed across an extensive range of network bandwidths while still maintaining quality. The protocol specification presented within this paper will serve as a possible extension to enhance the multicasting capability of CAFSS-Net.

One of the techniques utilised to increase the efficiency of multicasting is to provide better encoding standards for the media to be streamed. *Krassic* demonstrated the "priority drop" adaptation with media streams over a wide range of rates and with fine granularity [35]. Fine Granularity Scaling (FGS) is a MPEG-4 video coding standard particularly designed for media streaming and is the focus of another research paper that "evaluates the Streaming of FGS-Encoded Video with Rate-Distortion Traces [13]. The evaluation of this encoding technique produced in this paper was the benefit of providing 'scene optimisation' as opposed to 'frame optimisation', which in turn shows that it greatly decreases the computational effort. This paper investigated features applicable to both video scene and video frames which our study will also investigate.

The literature review thus far has examined streaming media with multicast support. With the advent of greater availability of bandwidth and improved widespread communication, some modern streaming techniques have begun utilising modernised alternative means of distributing content. The discussion so far involves the broadcasting of streams from a source to various clients in various formats. A modern adaptation of content distribution focuses on direct Peer-to-Peer implementations.

2.4.2 Peer-to-Peer Implementation

One of the popular content distribution protocols at the moment is BitTorrent. BitTorrent is a "protocol that enables fast downloading of large files by utilising minimal Internet bandwidth" [37]. This is accomplished by obtaining pieces of the file requested simultaneously from various sources. The success of this protocol lies in the large collection of peers with the same pieces of content, increasing the availability of sources (also known as a swarm). *Shapiro* provides a proposal that highlights the efficacy of BitTorrent and how it helps alleviate the load on a server by having multiple parties provide the same source for other parties whilst simultaneously downloading the files themselves [38]. This resource iterates both the advantages (sharing and minimising costs) and possible network community benefits of this protocol.

Carnegie Mellon University presented an application of a Peer-to-Peer sharing system proposed to speed up movie and music downloads [39]. They developed a research system called Similarity-Enhanced Transfer (SET) which not only looks for similar files, but similar chunks of files. This demonstration provides an insight into a community driven adaptation of peer-to-peer technologies to better serve their society. However in this example, no media streaming was explored. This article merely references an example of peer-to-peer networks and their applications.

A comprehensive research paper by *Xu et al.* [40], details a specific peer-to-peer streaming media system which exhibits a collection of characteristics imposed on general peer-to-peer systems. These characteristics are specified to analyse interesting problems faced with streaming media in general as follows [40]:

1. How to assign media data to multiple supplying peers in a single streaming session?
2. How to quickly enhance the system's total streaming capacity?

The author presents an investigative solution to the first question in the form of an algorithm, particularly an "optimal media data assignment algorithm". For the second question, a protocol was specified yielding a "distributed differentiated admission control protocol". It demonstrates that peers who offer their available bandwidth help achieve "quicker system capacity amplification", which effectively shortens the waiting time taken by peers. A similar approach was utilised by *Padmanabham et al.* [41] to provide "resilient peer-to-peer streaming". Here the focus was to provide robustness to alleviate peer transience (brief user participation within a particular network) by applying redundancy to both the data and the network paths.

Although BitTorrent has been intended for file distribution in general, an interesting research article explains how this protocol can be adapted to improve media streaming. According to *Shah and Paris*, they proposed a peer-to-peer streaming media solution based on the content-distribution nature of the BitTorrent protocol [42]. The research also shows that peer-to-peer distributions have "provided some benefits over the traditional client-server architectures" seen in most streaming media implementations. These include [42]:

- Handle very large and unexpected surges of content demand
- Surpass the limited bandwidth availability of the server
- Need not require any special support from a network
- Runs over IP Multicast and other content distribution schemes

However even with all these benefits, BitTorrent is not acceptable as a streaming media solution because it can not support real-time requirements of streaming applications [42]. This is a result of peers not obtaining pieces of the files in sequence and the current aggregation of peers being forced to wait too long before entering the swarm [42]. This research suggested the following two adjustments to the protocol: forcing the peers to obtain the first pieces of the file that is to be viewed and decreasing the wait time of the other peers wishing to enter the swarm. The outcome of this investigation showed that both these modifications yielded better streaming quality.

The research from [37–39, 41] demonstrated different strategies for peer-to-peer networks and their benefits which they provide to their particular environments. However, *Xu et al.*, *Shah and Paris* focused their peer efforts on streaming media [40, 42]. While *Xu et al.* [40] yielded a more detailed investigative approach to peer-to-peer streaming, *Shah and Paris* [42] provided an insight into peer-to-peer media streaming using BitTorrent. Our research will incorporate a peer-to-peer network test for CAFSS-Net.

During the process of streaming, whether it be a client-server scenario or peer-to-peer scenario, bandwidth is always utilised by the transmission. According to *Ho and Lee* , the quality of service control between the receiver and sender is difficult due to the Internet's inability to always guarantee the availability of bandwidth between the two ends [7]. *Ho and Lee* tackled the phenomenon by employing a predictive buffering algorithm which is not utilised from a single sender, but by many senders [7]. The research concluded that the overall difference between their buffering prediction algorithm and the actual 'efficient buffer time' within best-effort networks was within a satisfactory range. This research showed an advanced adaptable implementation of video streaming over the Internet where some of these concepts (buffer prediction) will serve as a suitable extension for CAFSS-Net.

2.5 Algorithms and Strategies

Even though some of the following algorithms and strategies have been reviewed in the previous section 2.4, the literature on the algorithms and strategies reviewed in this first subsection 2.5.1 are focused on the distribution of content. The subsequent subsections review the proxy and caching algorithms/strategies.

2.5.1 Content Distribution

Whether there is a single source providing a media stream or multiple servers, managing both the distribution and the diversity of the stream is part of the foundation of any streaming environment. *Conklin et al.* discusses the concept of video coding for the delivery of streaming media specifically over the Internet [43]. The primary focus consists of three sections:

1. The delivery mechanisms: which can be unicasted or alternatively multicasted.
2. The optimisation criteria: which consists of minimising the distortion for the connection rate or minimising the traffic.
3. The distribution model of the stream: which can be on-demand or alternatively streamed live.

These three sections from *Conklin et al.*'s discussion provided useful concepts for the specification of CAFSS-Net [43]. They demonstrated the core components needed to transmit a video stream over the Internet which CAFSS-Net will utilise in its definition. The reviewed literature within this section serves as background to the streaming media algorithms and strategies available.

Multiple Description Coding (MDC) is a popular optimisation strategy employed by most common streaming environments which can be applied across multiple paths [44]. Since path diversity represents the "streaming over multiple paths to overcome both the loss rate and the delay", the following research by *Apostolopoulos and Trott* [45] (which also acknowledged the use of Multiple Descriptions) proposed the following path diversity architectures as suitable measurements for overcoming these problems:

- "A single Description versus Multiple Description versus Scalable stream" where the choice depends on the context

- "A single-Sender Path Diversity" versus "Multiple-Sender Path Diversity" where the use of Content Delivery networks (CDN) provide Path Diversity to Multiple-Sender architectures
- "A path diversity architecture over Wireless Networks" where for 802.11 networks, exploitation of different access points provides more source availability.

Another research paper focused on a larger population of hosts. *Padmanabhan et al.* proposed the situation in which a "large and highly dynamic population" of hosts are potential viewers of live streaming [46]. The intention in this investigation was to "provide robustness" by incorporating redundancy within the network paths and in the data itself. To achieve this, they presented an efficient tree management algorithm that manages "diverse distribution trees to accommodate redundancy in the paths of the network". In order to facilitate the management of the load, they also proposed an "adaptation framework" to manage the Multiple Description Coding (MDC) [13,44,47] which accomplishes the redundancy in the data.

MDC uses multiple descriptions which are "separate bit-streams containing complementary descriptions of a signal stream" which provides redundancy for the media stream [45]. This approach is considerably different to both *Ho and Lee* [7] and *Nguyen and Zakhor* [48] because of the precise representation of both these cooperative algorithms. The research by *Padmanabhan et al.* [46] focused on the actual "Peer-to-Peer content distribution scheme" CoopNet in which this scheme is evaluated by testing traces "gathered from MSNBC during the September 11, 2001" event.

2.5.2 Proxy

As mentioned earlier on, a proxy is a filter or service that forwards requests from and to other network devices on the network [24]. A commercial framework that assists in streaming media proxy development is the RealNetworks Helix Proxy [49]. This is a commercial delivery platform for digital media which incorporates various formats. It is specifically designed to reduce bandwidth costs by alleviating redundant client-server requests which decreases bandwidth usage. Some large institutions such as our own (Rhodes University) utilise their own proxy systems that monitor/filter requests between users and external servers to help manage content and decrease bandwidth utilisation. Figure 2.5 represents the existing configuration:

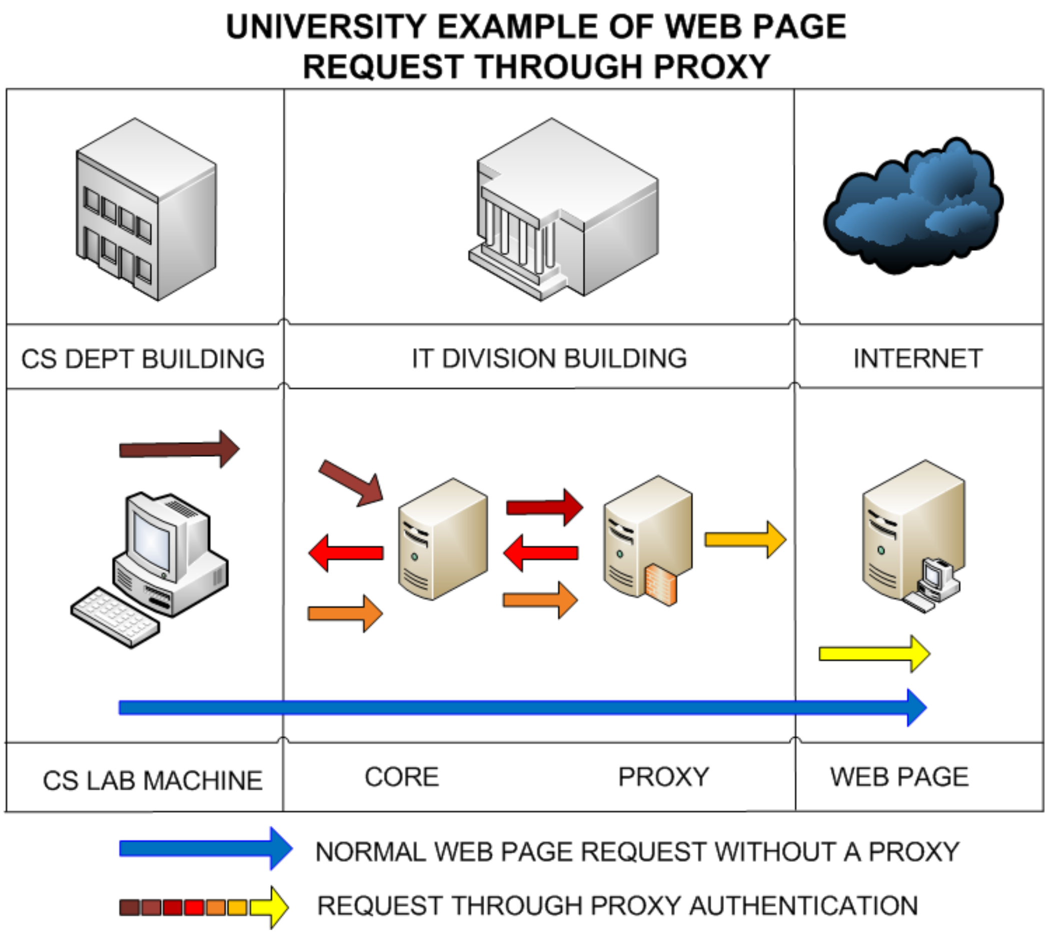

UNIVERSITY EXAMPLE OF WEB PAGE REQUEST THROUGH PROXY
CS DEPT BUILDING
IT DIVISION BUILDING
INTERNET
CS LAB MACHINE
CORE
PROXY
WEB PAGE
NORMAL WEB PAGE REQUEST WITHOUT A PROXY
REQUEST THROUGH PROXY AUTHENTICATION

Figure 2.5: Web request through a proxy vs a standard request

Figure 2.5 illustrates a client machine (which resides in the Computer Science Department) requesting a normal web-page from the Internet which is external to the University network. The alternative is the same web-page being accessed by a client through a proxy which intercepts the request and in this scenario, requests the user enter their username and password to validate their access to the network. This validation is effective as it manages the content, applies quota restrictions to users, adds a certain level of security and provides user access control. The coloured arrows demonstrate the sequence of events during the authentication followed by the rest of the request. This proxy example is specific to this university environment.

A comprehensive strategic paper by *Sen et al.* expresses a solution to high loss rates and latency of streaming media in the Internet by introducing a technique that incorporates a proxy which stores initial frames of popular clips [50]. It requests the remaining frames from the server upon a client's request for the playback of that particular clip. What is interesting is that not only does it decrease the delay and throughput between the various end points, but can aid in the application of Workahead Smoothing which minimises the peak bandwidth of the bit-rate stream. This peak bandwidth usually arises from a variable known as *burstiness* (the network traffic where the short-term bandwidth exceeds the average bandwidth) [51]. This filter/service also provides test examples of the proxy application and demonstrates that a "few megabytes of buffer space on the proxy can significantly reduce bandwidth".

Proxy systems have been synonymously associated with caching in many instances within this review of literature. This proxy service [50] works in conjunction with the technique which is a form of caching designed to assist in streaming media distribution. As a result, caching shall be discussed in the next section. The incorporation of a proxy (particularly the initial frame storage concepts from the [50] research paper) into CAFSS-Net will serve as a suitable extension to the framework.

2.5.3 Caching

Caching is the process of storing copies of the streaming data on the local network for future use by other users on the local network [10]. Numerous caching strategies and algorithms have been implemented in networks, especially in the streaming media environment. As mentioned in the proxy representation above [50], caching was implemented in conjunction with a proxy system to form a proxy prefix caching scheme. The proxy prefix caching scheme utilises the proxy frame storage policy that stores a fixed set of frames at the start of each of these popular video clips [50]. These frame sets are the designated prefix. The intended aim of this prefix is to avoid the storage of the entire resource which is normal in traditional caching systems. It demonstrates an in-depth specific streaming media implementation aimed at decreasing latency by applying the proxy cache system to video frames.

Almeida et al. examines the concept of "full file caching" which outperforms the traditional prefix caching when the context is specified in system design space at the cost of considerably more storage [52]. The research also provides certain cost-models which aim to determine the benefits of the various streaming systems investigated by the study. Since numerous caching strategies are used for various streaming configurations, *Shen et al.* investigated caching strategies in transcoding-enabled proxy systems for distribution networks based on streaming media [53]. The investigation probes transcoded-enabled caching (TeC) between wireless and wire-line environments. They employ two types of network measurement strategies: synthesised traces (similar to simulated network traffic) and server logs. This research also utilises proxy caching and focuses its TeC strategies between wire-line and wireless environments.

As previously mentioned at the end of the previous subsection, proxies and caching have been synonymously expressed in literature throughout the media streaming field. The previous citation showed a test system modelled in a simulation environment. *Hofmann et al.* and *Houtzager and Williamson* [54, 55] both present simulation models to help optimise caching schemes (these simulations shall be discussed within the next chapter). The first group presented different caching techniques for streaming multimedia over the Internet [54], whilst the latter proposed a simulation study at a packet-level for the optimal placement of a web proxy cache [55]. *Hofmann et al.* provided a complete caching architecture that evaluated the efficiency of the various cache techniques [54]. However, the technicality and definition of these schemes fell outside the scope of our investigation.

2.6 Exploring Existing Systems

2.6.1 Microsoft Streaming Services

Microsoft Windows Server 2003 is a powerful network management operating system that serves numerous businesses and organisations around the world. An interesting feature of this system suite is Microsoft's Windows Media Streaming Service 9 (WMS-9) [16]. This service provides support for live media streaming or on-demand facilities across the Internet. The service comprises of tutorials and tools that allow a media stream to be configured within the Windows Server environment.

The installation requires a running version of Windows Server 2003 (Server 2K3. later versions are also available) with administrative privileges available to install the tool-set [56]. The interface provides support for managing the various details of the stream, such as name, content, playlists and user-space. This service provides locations or web/network "publishing points" which manage or control access to the streaming media from the client side. Other features are available such as dynamic content, extensibility and also large industrial distribution. Even though a later release was available after our initial testing phase, a decision was made to experiment with version 9 due to its well-known tried-and-tested infrastructure.

A companion to this service was also available which assists in obtaining video sources that could serve as input video for the Windows Media Services framework. The extension of this service consists of the Windows Media Encoder 9 (WME-9) which presents the process of capturing video for content distribution [57]. Figure 2.6 illustrates the stream flow from the source video to the clients. With this streaming scenario, various options are available:

- Server 2003 can broadcast pre-encoded videos through WMS-9
- WME-9 either pushes the video to Server 2003 or Server 2003 pulls it from WME-9
- Also if the hardware is available on the Server, WME-9 can be setup on the Server 2003 machine

The standard tutorials assisted in the construction of the video broadcast from the server to other clients on the network as illustrated in Figure 2.6. Our initial examination focused on the various video qualities available, and the necessary features that were required to maintain a standard video broadcast. However we noticed that there was no multicast support in this particular version of this service.

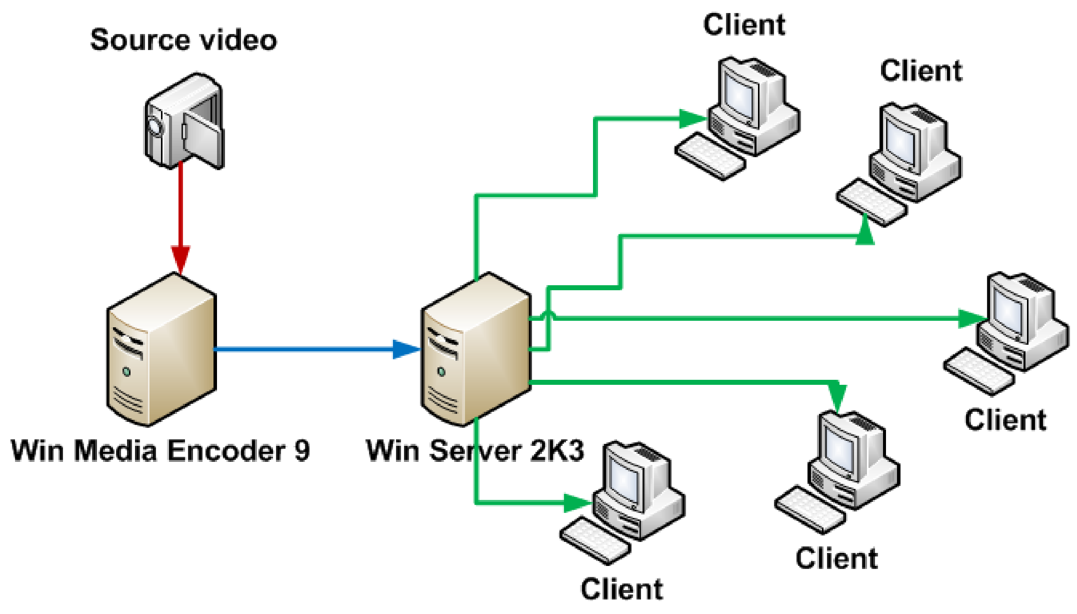


Figure 2.6: Microsoft Windows Media Services Example

2.6.2 Apple Darwin Streaming Service

Apple provides excellent MAC OS-X facilities and software solutions for video media. Apple presents their unique Quicktime movie format (MOV) with their Quicktime movie player [58]. However, this is just a media player and Apple's streaming service is the Darwin Streaming Server [59]. This server provides only the on-demand streams, but to use live streaming, it can be accompanied by the Quicktime Broadcaster. This is the equivalent of the previous WME-9. In this instance, the streaming server reflects or relays the broadcasting stream to its clients.

Darwin Streaming Server is an Open-Source streaming server which has continuous development which extends its functionality. This streaming server is primarily designed for Apple's Operating System MAC-OS X [59], but various installations are available for Linux, Solaris and Windows Server. The Windows variant was chosen because the existing Microsoft Server 2003 with the WMS-9 was already implemented. A difficulty that was encountered consisted of the console server application failing to load because only one streaming service can operate within the same Microsoft Server 2003 machine (this is a result of the conflicting services trying to communicate on the same network ports).

The interface is accessed through a Web-based Streaming server Administrators space. It is accessed in much the same way as a traditional web-address but instead of providing a complete URL, the administrator specifies the network address of the server machine. The installation requires the implementation of usernames and passwords to be set for various sections of the streaming administration. The administration section allows for the following [59]:

- viewing of logs (errors, access history)
- Management of port settings
- Remote administration of other servers

The last feature is interesting as it demonstrates a complete integration of the various services working to provide a distributed streaming platform.

Figure 2.7 demonstrates that the Darwin Streaming Server operates in the similar manner to the Microsoft equivalent. However, the difference in this instance is the feature to remotely access and administer the Quicktime Broadcaster from the server machine. This is illustrated by the blue 2-way communication arrow in Figure 2.7. Another advantage over the Microsoft equivalent is the availability of multicast support.

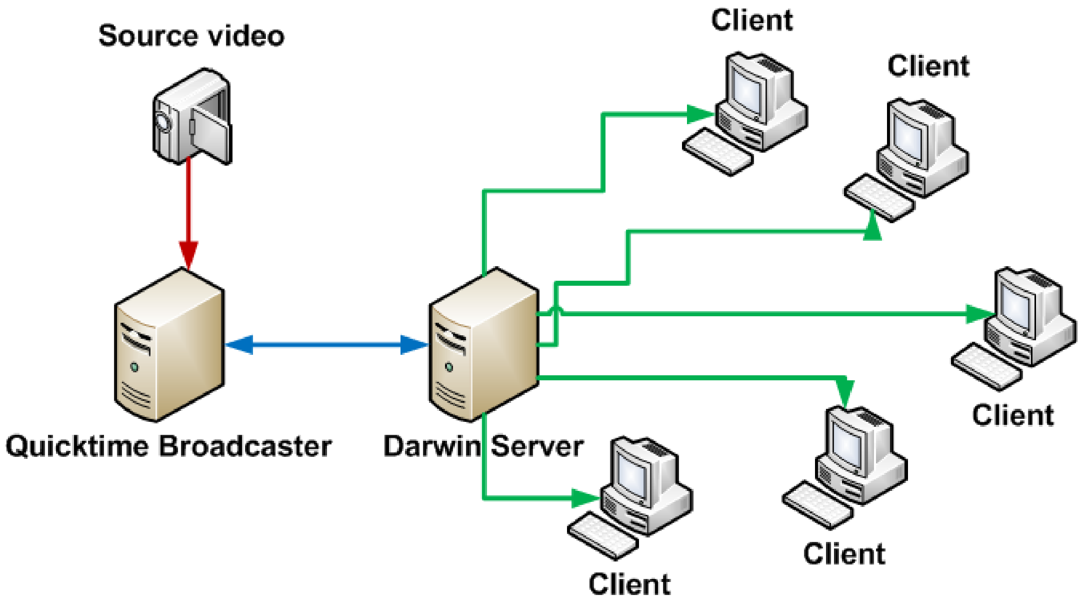


Figure 2.7: Darwin Streaming Server Example

2.6.3 RealNetworks Helix Server and Helix Proxy

RealNetworks supply a suite of streaming programs that provide excellent alternatives to the previous Microsoft and Apple equivalents. RealNetworks provides a streaming media delivery platform known as the Helix Server [60]. This streaming platform once again provides the standard streaming capabilities such as both on-demand and live video. However what makes this streaming platform unique is the multi-service support for different codecs (such as Windows Media, Quicktime, MPEG-4 and Macromedia Flash).

The multi-service support is very useful as it provides streaming distribution to various media players. The Microsoft and Apple service/system required their specific media players in order to view their streaming content (Microsoft required its Windows Media Player and Apple required its Quicktime Movie Player). The Helix Server not only distributes to its RealNetworks Real Player [60] but also to the given alternatives available. Since different media players operate through different communication ports, this server is able to provide the streams across different protocols to better optimise the throughput to the media player.

The installation operates in similar manner to the Darwin Streaming Server with username and password specifications. It also runs as a console service with the administration based through a web-based interface. The web-based interface has a wide variety of customisable options with tutorials to provide test-demo streams for the various playing formats. There are also other features that provide support for this streaming platform:

- Simple Network Monitoring Support (SNMP) allows monitoring of the service on any networked host with SNMP support
- Supports publishing from WME-9
- Internal cache and proxy support

RealNetworks has a sophisticated companion for the Helix Server which assists in optimising performance and minimising bandwidth for the server's content distribution. RealNetworks provides a proxy service called Helix Proxy which sits between the server and the clients as illustrated in Figure 2.8. Figure 2.8 also shows the similar network structure to the Darwin and WMS-9 services and the proxy provides powerful caching capabilities for the server. However, it is recommended not to place both the Helix Proxy and Helix Server on the same machine as they will both compete for the machines processing power.

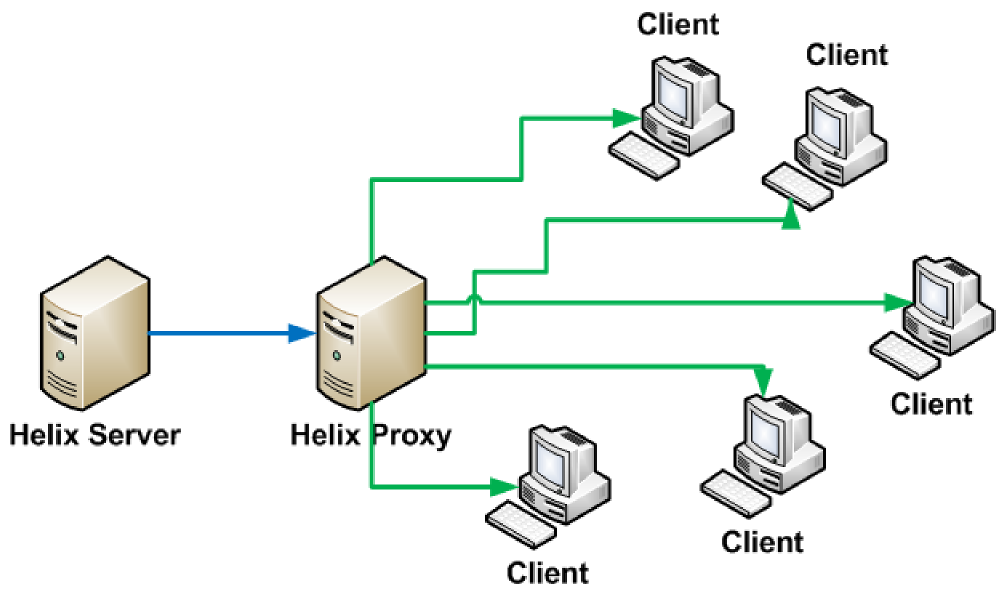


Figure 2.8: Helix Server Streaming Platform Example

There are various other streaming systems available (such as Macromedia Flash) that provide powerful streaming functionality. These selected systems/services represent a significant portion of the current streaming media distribution engines used today. We present some citations for each of the streaming system/services in which it has further enhanced the media delivery for a given company/organisation:

- Darwin Streaming Server: Youtube [61], Akamai [62]
- Helix Streaming Server: Harvard University [63], State of Washington Department of Information Services [64], a previously cited paper on caching [65]
- Microsoft Streaming Services: MTV URGE (Microsoft & MTV) [3], Beuna Vista Internet Marketing Division, [66]

2.7 Network Simulation

2.7.1 Definition

Simulations are pre-defined experiments performed on an artificial representation of a known scenario. A network simulator is a software based tool that represents a physical network [67]. There are various types and collections of network simulators because various sizes and types of physical networks are available. Networks can be modelled from a simple client-server scenario (such as a Local Area Network: LAN) all the way to large international networks (or Wide Area Networks: WAN). Some network simulators are only able to scale to small sized networks whilst others can represent complex WANs accurately.

Since networks may encompass a range of smaller sub-networks with various configurations and topologies, physically modelling these conditions may prove to be an expensive task [68]. The alternative to network simulation is the process of network emulation [69]. The main advantage of network simulation over network emulation is that "time can be represented artificially" [69]. However, network simulation represents network characteristics according to models [68] and this unfortunately limits its accuracy. Various other third parties (such as human error, foreign or external network traffic) can have an adverse affect on the results of the physical network test. There are also disadvantages for simulations in general. The following presents the details [70]:

Advantages

- Controlled environment rules out external factors like:
  - Human error
  - Network congestion
  - Poor connectivity
  - Foreign network traffic

- Effectively manage time by:
  - Avoiding the physical construction of a network
  - Easily locating a specific aspect of a network
  - Saving simulation data for future use or further study
- Efficient use of space by:
  - Representing an entire network inside a software application
  - Saving a simulation to a format for other parties to examine
- Observe different viewpoints of a network such as:
  - Packet transmission
  - General network traffic
  - Specific network traffic (such as TCP or HTTP)
  - Physical representation such as node activity
  - Network visualisation (types of packets, lost, overflows)
- Cost effective by avoiding:
  - The purchasing of physical hardware
  - Repairs to equipment in the event of a hardware failure
  - Bandwidth usage (a significant factor to WANs)

Disadvantages

- Computational intensity:
  - The more nodes, the more processing power required
  - Complexity between the nodes can decrease performance
  - Modelling the dynamic nature of the Internet

- Physical Accuracy:
    - To increase performance, abstraction is applied which loses accuracy
    - Most simulations are based on models which are abstractions of real world entities

Despite the simulation disadvantage of computational intensity, modern computing and advanced hardware are allowing for more complex models and network abstractions that can increase the performance of network simulators [70].

2.7.2 Existing Network Simulators

There are various network simulators available to the public. The majority of network simulators are discrete event based simulators [69, 71] which utilise a scheduler to apply a particular event at a specific time (it is analogous to an athletics meeting where each sports event takes place at a particular time for a given time period according to a program). The following provides a brief overview of some of the popular network simulators currently available [67]:

CSIM-19 This is a process-oriented discrete event simulator implemented in C/C++ [72] and developed primarily for the modelling of complete systems [73]. This toolkit is not only a simulation engine, but also contains a collection of classes and methods that represent various routines of complex operations. This is a commercial product which provides a wide distribution base spanning from educational institutions to large global corporations [73].

OPNET OPNET Modeler is one of the primary facilitators for network modelling and simulation [74] and is designed in an object-oriented manner. This platform is primarily used in the development of communication networks, networking technologies and protocols by the world's leading networking engineers. The focus of this service is on the accurate realistic deployment of simulated strategies with detailed specifications for various network vendor operating services.

Modeler is currently the leading network simulation service provider on the market [74].

NetSim This simulation tool is developed by Tetcos and provides performance metrics which are detailed and conform to the various international standards (such as IEEE, ITU, IETF) [75]. NetSim also supports various network protocols as opposed to the traditional Ethernet, such as Token Bus, Token Ring and Frame Relay. This simulation tool is supported by numerous educational environments to provide a scalable infrastructure for educational networks. NetSim develops its models through the use of objects and provides extensive reports.

QualNet This network modelling suite provides both network simulation and network emulation. QualNet provides an efficient, portable and scalable environment for the deployment of virtual networks [76]. Developed by Scalable Network Technologies, a host of extensions founded on this network modelling suite are also available from the same developer. The definitive feature of this application suite is its capacity to accurately represent a digital clone of the physical network condition [77].

NS-2 This is a discrete event scheduler developed with the TCL scripting language for the front-end and a C++ representation for the back-end [78]. This simulator is also an object-oriented based solution and is developed for UNIX systems with the availability of porting it to Windows [8]. This simulation tool has been used extensively in academic research environments. Its primary advantage over its competition is its open-source development which makes it available to the public for free, with no need for licensing [79]. NS-2 is also very extensible with access to the tool's kernel source, which allows users to both alter existing protocols or create new ones. This tool's extensive academic research yields another important feature which is its online documentation [80].

2.8 Videos

Video is a very intensive data medium especially within networks. With modern video advancing to a wider device distribution (cell-phones, MP3 players) and achieving higher resolutions, modern video media development has generated new video technologies [81, 82]. IPTV is a video on demand distribution format layered over IP [83] with advanced consumer functionality such as customised advertising and content management. Test beds have already been deployed to explore these technologies [34].

Another technology evolution which is gathering momentum is the next generation delivery of high quality, high resolution video. High Definition video (HD) is the next generation of multimedia distribution. Both traditional television broadcasting and the video entertainment industry have adopted this development [84, 85]. In order to further explore video relevant to this study, we will start by examining the different types of video available then delve into other video specifics.

2.8.1 Types

Video has numerous formats and classifications and can either be digital (DVD, satellite television) or analogue (traditional television broadcasts, VHS cassette tapes). For this research, we will explore only the digital variant because the analogue variant is becoming a legacy system and CAFSS-Net operates within a digital domain. Digital video can be represented in its uncompressed raw format or in a compressed (possibly lossy) format [5] where the significance in this instance is the trade-off between quality and storage space. Various algorithms are available for compressing/decompressing videos (also known as codecs) with their unique features and quality outcomes. We shall explore only the types applicable to CAFSS-Net.

YUV YUV is the uncompressed raw intermediate format for video which uses a colour representation scheme equivalent to human perception [86]. YUV can represent both digital and analogue variants, but we shall only explore the digital equivalent. It is commonly represented in two different interfacing formats used to standardise horizontal and vertical resolutions: Common Interface Format (CIF) and Quarter Common Interface Format (QCIF). The characteristics of YUV are as follows [86]:

Y	Represents the luminance - the brightness (defined by black and white) of the still frame
U	Represents one of the chrominance values (colour) used to provide the colour spectrum of the video. In most cases U refers to the blue spectrum
V	Represents the other chrominance value which in most cases refers to the red spectrum

MPEG Moving Picture Experts Group (MPEG) is a video media compression standard used to encode traditional analogue video [87]. This was the first video compression standard which encompasses many of the existing compression algorithms of today's video media. Formed in 1988, MPEG has produced various international video standards by incorporating two mediums of compression, SPATIAL and TEMPORAL. SPATIAL compression (or spatial redundancy) takes advantage of neighbouring pixels which are similar. TEMPORAL compression (or redundancy) takes advantage of the similarity of successive frames. The popular standards for the delivery of modern video are:

MPEG-2	Modern DVD, SVCD encoding (**S**uper **V**ideo **C**ompact **D**isc = predecessor to DVD)
MPEG-4	Provide low bit-rate encoding to achieve greater compression which in turn minimises storage space.

2.8.2 Resolution, Aspect Ratio and Frame Rate

With digital video, not only is the quality of compression a factor, but also the resolution (the dimensions of the pixels of the video), the frame rate (the number of frames per second) and aspect ratio (the dimensions of the video screen/viewer) [28]. A pixel represents a picture element defined by a colour. An image is a collection of pixels organised in a visual matrix. The greater the number of pixels, the higher the quality of the image. Videos are successive images played sequentially and follow the same structure. Another aspect of video resolution is the refreshing of the video which is normally applicable to narrow bandwidth television technologies. This can either be interlaced (refresh odd and even lines separately) or progressive (refresh all the lines on the screen).

Resolution is defined as the horizontal collection of pixels versus the vertical collection of pixels (an example: 640x480 which is 640 picture elements horizontally by 480 picture elements vertically). The larger the resolution, the larger the storage space required because of the necessity to store each pixel. However, the display resolution can also determine the optimum aspect ratio for viewing the video. The aspect ratio is the dimensions of the screen/viewer and it is a measure of the viewing screen size. Unlike the resolution, this measurement is in the form of a ratio (an example: 4:3 implies the ratio between the horizontal length and vertical length is 4 to 3). Table 2.1 shows some of the popular resolutions accompanied by their aspect ratios, *Terms* and examples of applications [88]:

A comparative research paper explored the relationship between the bandwidth utilised (or quality in this instance) and the density of movement within the video scene (scene motion) [8]. It was observed that there is a significant relationship between the two such that the greater the change of motion within a scene, the more bandwidth will be utilised during the streaming of that video. These results have been a factor for the incorporation of scene motion in the study and are presented in section 3.3.5.

Resolution	Aspect-Ratio	Term	Example Application
320x200	8:5	CGA	mp3 players, cellphones
320x240	4:3	QVGA	PDA's
640x460	4:3	VGA	Portable Game Console
720x480	3:2	NTSC	American TV Monitor
768x576	4:3	PAL	European TV Monitor
800x600	4:3	SVGA	Old 15"Monitor
1280x720	16:9	HD 720	HD TV Screen
1280x800	8:5	WXGA	Wide Screen Monitor
1280x1024	5:4	SXGA	Standard 19" LCD Monitor
1680x1050	8:5	WSXGA+	HI-RES Monitor
1920x1080	16:9	HD 1080	Full HD TV

Table 2.1: Resolution and Aspect Ratio examples

2.8.3 Video Frame Types

With digital video, particularly MPEG video, there are different types of frames used to represent video. MPEG-4 utilises three distinct video frame types to represent compressed videos. A brief description of these frame types are presented as follows [89]:

I-Frame Intra-frames are commonly known as keyframes because they can be decoded independently from other frames. They are also synonymous with simple JPEG images.

P-Frame Predicted-frames are also known as forward-predicted frames. They function by improving the compression of video by storing only the difference in the image between two frames (either I or P). A P-Frame is also a type of inter-frame because it can not be decoded independently from other frames.

B-Frame Bidirectional-frames are also known as backward-predicted frames. They are similar to P-Frames except that they can make predictions both forward and backward (hence bidirectionally). On the other hand, an I-Frame or a P-Frame must be decoded sequentially after a B-Frame for the B-Frame to be displayed, hence making the B-Frames computationally expensive to decode and encode. B-Frames are also an inter-frame type.

2.9 Summary

This chapter discussed the various literature presented within the streaming media field. Streaming is an excellent means of providing content without downloading and serves as a sufficient alternative to traditional downloading. Although streaming media does require both video encoders and a streaming source of some kind, it does offer a strong support for live broadcasts. This chapter highlighted the terminology and jargon for this research before it presented the literature pertaining to the streaming media field.

Both the characteristics of the network, video and the topology play a significant part in the performance of the streaming environment. Advances in multicasting approaches and peer-to-peer distribution schemes have yielded some interesting developments for the modern streaming age. Many streaming algorithms and strategies have been devised for streaming media to combat the various complexities of the modern Internet. Concepts ranging from packet-level simulations to complete caching architectures present viewpoints of optimising streaming media. CAFSS-Net thus far seeks to incorporate a multicasting networking architecture, that can incorporate various networking conditions to model a holistic overview of streaming media in general. Certain concepts were briefly mentioned in the literature, particularly references to simulations and video characteristics. These shall be discussed in the next chapter.

Chapter 3

Methodology

3.1 Introduction

Streaming media consists of various components that work logically to provide a continuous flow of data from a source (or sources) to one or more end-points. This chapter begins with the evaluation of existing systems, the formulation of an evaluation table and the subsequent utilisation of this table to construct CAFSS-Net. This will be followed by the definition of CAFSS-Net with its three major elements.

The first element of CAFSS-Net consists of the network simulation specification where we explored the various network simulators available in Chapter 2 and then the specifics of our chosen simulator NS-2 in section 3.3.3. The second element consists of the tools needed to evaluate the video streaming available through CAFSS-Net. The third element consists of the videos which represent the source video streams that will be transmitted through the network. This chapter finally concludes with the definition of the testing scenarios (the test bed) which elaborates on the configuration for the testing of our framework.

3.2 Evaluation of Existing Systems

There are various aspects in the different streaming media distribution systems that are comparable to one another. Such an example is the licensing for each of these systems. Darwin Streaming Service is an open-source distribution and therefore requires no licensing for the distribution of its services. Windows Media Services is a free application service for the Windows Server 2003 operating System. Real Networks provides a 30 day evaluation on their Helix Server and Helix Proxy systems, but upon completion, recommend contacting their sales office for the purchasing of the product.

An overview of the three streaming media distribution systems was done and common features relating to media streaming in the three systems were identified. Some of these common features were compared to aid in the formulation of CAFSS-Net. The features are summarised in Table 3.2.

Features	**Darwin**	**Helix**	**WMS-9**
Version	5.5.4	11.1.1	9.0
Available OS	MAC-OS X, Solaris 8 RedHat, Windows NT	UNIX, Win Server 2K3	Win Server 2K8, Win Server 2K3
License	Free (Open Source)	30 Day Trial	Free (App Service)
Unicasting	Yes	Yes	Yes
Multicasting	Yes	Yes	Yes
On-Demand	Yes	Yes	Yes
Cache	No	Yes	Yes
Proxy	No	Yes	No
Interface	Web-Based	Web-Based	Server 2K3 Service
Extensions	Quicktime Broadcaster	Helix Proxy	WME-9

Table 3.2: Features of Streaming Media Systems

Of the features reflected in Table 3.2, the following were used to formulate the common analysis framework: Available OS, Unicasting, Multicasting, proxy and Cache. The next section defines the common analysis framework and the subsequent sections highlight the major components of CAFSS-Net.

3.3 Defining Our Framework

3.3.1 Motivation

Modern streaming video consists of distribution engines that provide streaming services through networks over the Internet [9]. However, many of these streams are transported through a collection of various network bandwidths and networking structures which can affect the quality of the video transmission. The improvements in network bandwidth have assisted in the growth of modern video streaming media [8]. However, since video streaming media is significantly dependent on the characteristics of the communication medium [4], various streaming optimisation strategies and techniques have been applied by existing streaming media distribution systems to ensure a successful video stream transmission.

We propose a streaming media common analysis framework that provides guidelines which assist with the prediction of performance impacts of various changes to an existing streaming media network. The Common Analysis Framework for Simulated Streaming-Video Networks (CAFSS-Net) is intended to predict the performance impacts of configuration changes, new features, and general scaling of these strategies and techniques. CAFSS-Net will serve as an aid in the design of video streaming media environments.

All the streaming media distribution systems discussed in the first section consisted of a streaming server, clients and a network connection medium between the two. We examined the different network streaming distribution formats that are available for streaming media. Since networks can have a variety of configurations and physical characteristics, designing a common analysis framework that can encapsulate the majority of these possibilities would be extremely difficult and prohibitively expensive in a physical network domain. For this reason, we implemented the networking analysis based on a network simulator. The next subsections provide the details to the following components of CAFSS-Net:

3.3.2 Components of CAFSS-Net

- **Streaming Server** - Sends the video stream
- **Client(s)** - Receives the video stream
- **Network Simulator** - Models the physical network
- **Video Publishing Tool(s)** - Converts raw videos to streaming format
- **Videos** - The videos to be streamed
- **Evaluation Tool-set** - Compares the server input with the client's output

With all the network simulation services available, determining an optimal solution for choosing a network simulator came down to the open-source cost effective solution of NS-2. Another factor for determining this optimal network simulator is the academic extensible nature of this network simulation tool which provides substantial online documentation. This is also enhanced by the comparative research paper [80] comparing NS-2 to OPNET (the leading network simulation service provider on the market) [74] and showing that from a researchers point of view, NS-2 provided similar results to the paper's network test bed as opposed to the OPNET system. The next section provides details of this tool.

3.3.3 NS-2 - The Network Simulator

As mentioned in the previous chapter, most simulation tools are discrete event simulators and NS-2 is one of the object-oriented versions [72]. To provide further insight into the discrete event simulation process, Figure 3.1 represents two media stream requests between a client and a server. The top part of the diagram shows a client requesting two video streams from a server sequentially. It illustrates the physical network communication between the two ends with the blue arrow representing the playing of the first stream and the request of the second stream. The grid below represents the same communication performed by a discrete event simulation. Here we note that each row represents a particular task denoted by a task ID and a time-stamp at which the event takes place (eg: ID=02 starts at 0.030 seconds into the simulation). This diagram is only an abstraction of the specific events, but it does highlight the sequence of events.

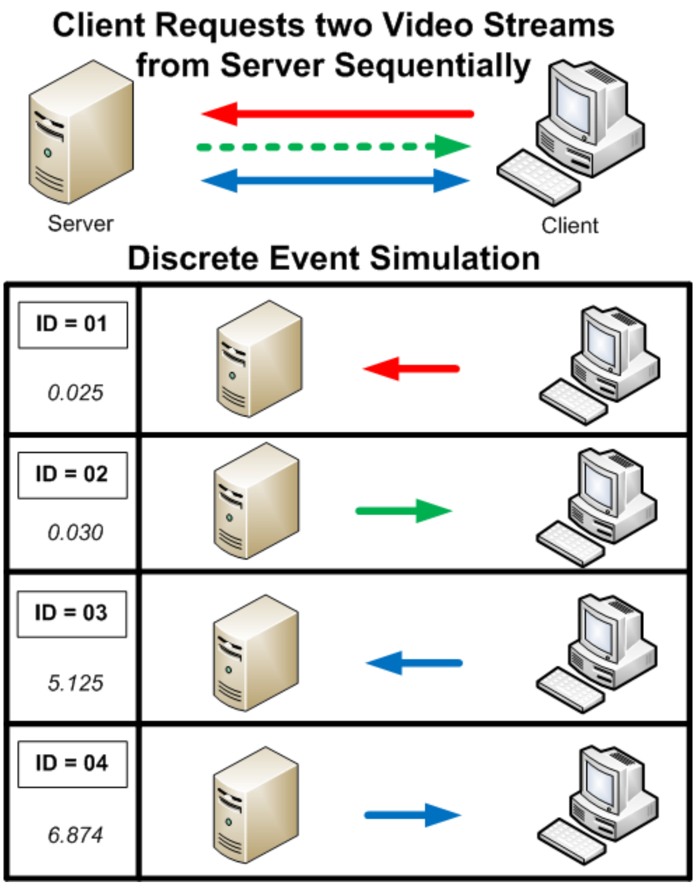

Client Requests two Video Streams from Server Sequentially
Server
Client
Discrete Event Simulation
ID = 01
0.025
ID = 02
0.030
ID = 03
5.125
ID = 04
6.874

Figure 3.1: Discrete Event Simulation Example

NS-2 was developed for UNIX systems but can be ported to Windows systems through Cygwin [90]. NS-2 contains a class hierarchy that it utilises to create simulations which are objects created through an interpreter. NS-2 comprises of two languages: C++ and TCL. The advantage of using two languages is that this allows for the quick manipulation of algorithms and packets through C++ whilst the object oriented version of TCL scripting (OTcl) is very efficient for making changes to the simulation configuration [79].

3.3.3.1 Environment

NS-2 contains a class hierarchy which defines the structure in which objects are instantiated. As stated before, OTcl defines the scripting for the simulation configuration and represents the front-end to NS-2. As a result, in most cases, this is represented as a parent object in the NS-2 class hierarchy in the form of a TclObject [78]. However, there is also a standard class hierarchy (or library) for the C++ object code for the back-end (as NS-2 contains both a front-end component and a back-end component). Figure 3.2 represents an abstract overview of the class hierarchy for the simulation objects.

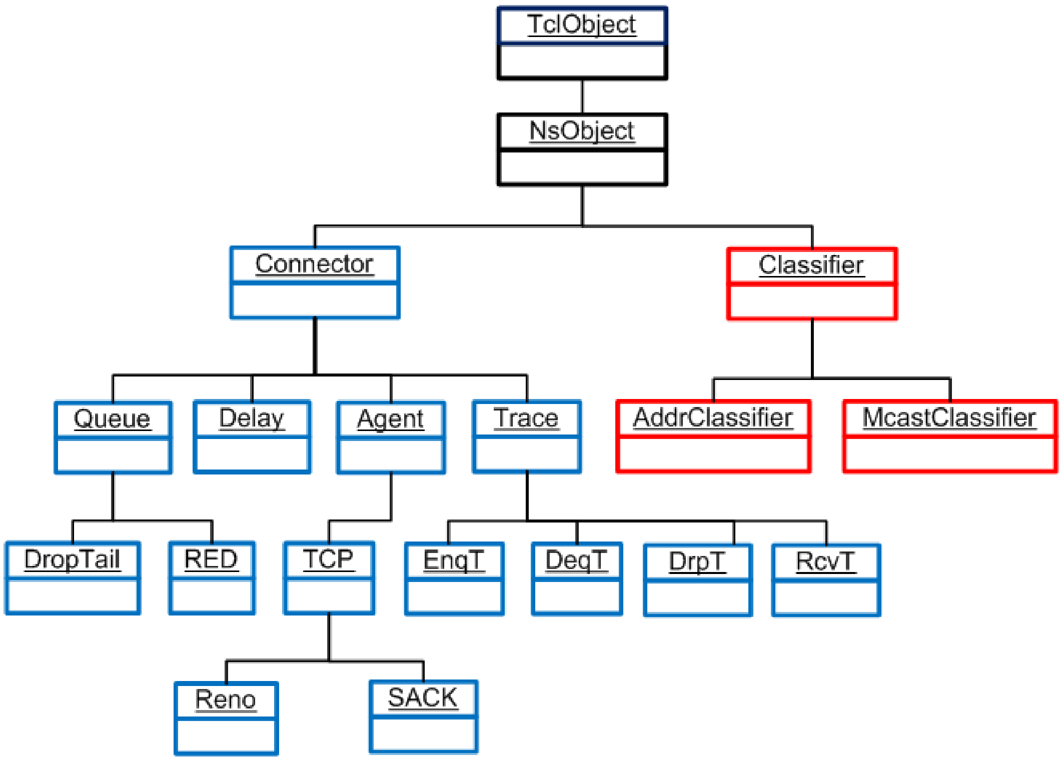


Figure 3.2: NS-2 front-end class hierarchy extended from [78]

The class hierarchy is segmented into two colour schemes. The blue represents the objects used in our investigation whilst the red constitute objects that are used for advanced modelling [79]. These red objects were not necessary for our investigation. Table 3.3 represents a brief summary of the blue highlighted objects and their definitions/purposes [91].

Object	Definition / Purpose
Connector	Receives packets from another connector then forwards/drops them
Queue	Models an output buffer attached to the link like a node buffer
Delay	Time taken by a packet to navigate a link
Agent	Endpoints which construct/consume network-layer packets
Trace	Used to write trace details or network events to a file
DropTail	First in First Out (FIFO) Scheduling queue management scheme
RED	Random Early Discard - detects congestion before queue buffer is full
TCP	Network protocol that provides reliable & dynamic congestion control
EnqT	Trace Element that references packets which enter the queue
DeqT	Trace Element that references packets which leave the queue
DrpT	Trace Element that references packets which drop from the queue
RcvT	Trace Element that references packets which the next node receives
Reno	1-Way TCP sender with Fast recovery (Inflated congestion window)
SACK	Selective repeat based on Acknowledge scheme (RFC2018) [92]

Table 3.3: Details for OTcl Class Hierarchy

The information thus far describes the structure of the NS-2 environment. The advantage of the NS-2 tool is the dual nature of the programming languages. The front-end maintains the simulation setup whilst the back-end maintains the core functionality. Generating a simulation is usually implemented in the front-end through the production of a Tcl script [93].

Generation of the Event scheduler

Generation of the event scheduler initiates the entire simulation process. The first stage was to define the initiation of the simulation by creating the event scheduler as follows:

set ns [new Simulator]

The next stage was the scheduling of the events such as *$ns at <time> <event>* in which an event could be any acceptable ns or Tcl command. A complete example is *$ns at 5.0 "finish"* which essentially states "stop the simulation at 5 seconds". The final stage was to start the scheduler by issuing *$ns run.* This was usually the last command executed within the script and located closer to the end.

Activating the Tracing Feature

The tracing feature allowed the network traffic information to be saved to a file which could be further examined to gather information such as packet loss, latency and other network statistics. To allow tracing the following command

$ns namtrace-all [open tracefilename.dat w]

saves all the network data to the designated TRACE_FILE (more information about the TRACE_FILE will be described later in this section). Another important feature of tracing is the concept of animating the simulation whereby the trace file serves as the input for the animation. This concept will be described later in this section.

Designing the Network

The design of the network represented the physical topology of the network which consisted of network nodes and the links between the nodes. The network nodes were defined as *set node0 [$ns node]* for the first node and *set node1 [$ns node]* for the second node. With reference to the class hierarchy, *queue*s were extensions to network links that emulated buffering. Hence, the links were defined as

$ns <link_type> $node0 $node1 <bandwidth> <delay> <queue_type>

where *link_type* could be a *duplex-link* network connection (traffic can go in both directions) or a *simplex-link* (a single direction network link) and the *queue_type* could be *Droptail*, *RED* and others.

Setting up Routing

Routing represented the flow of traffic amongst the nodes and this usually refers to unicasting or multicasting. A unicast route was defined as follows *$ns rtproto <route>* where the *route* could be Static, Session or DV multi-path. Multicasting was enabled by appending a command to the creation of the scheduler defined as *set ns [new Simulator* ***-multicast on****]*

and routing was defined as *$ns mrtproto <route>* where the *route* could be Dense Mode (DM), Shared Tree mode (ST) and others.

Introducing Errors

Since the simulations are in an independent environment from the traditional networks, these simulations usually exhibit perfect network transmission. Error generation consists of defining a new Error Module and specifying its details as follows:

```
set new_error_model [new ErrorModel]
$new_error_model set rate_ 0.02
$new_error_model unit pkt
$new_error_model randomVar [ new RandomVariable/Uniform]
$new_error_model drop-target [new Agent/Null]
```

Finally the module was inserted into the network link as follows:

```
$ns new_error_model $node0 node1
```

Generation of the Transport Connection

Network traffic is transported along a network link through a network protocol. In this study, the focus was on the two network transmission mediums which consisted of the TCP connection and UDP connection. TCP connections are characteristically reliable, dynamic and ensure accurate delivery [94]. TCP connections required the use of *TCP Agent* and *TCP Sink Agent* which essentially represented the transmitter and receiver. A single TCP example is as follows:

```
set tcp_sender [new Agent/TCP]
set tcpsink_receiver [new Agent/TCPSink]
```

```
$ns attach-agent $node0 $tcp_sender
$ns attach-agent $node1 $tcp_receiver
$ns connect $tcp_sender $tcp_receiver
```

With TCP as the transport layer, two representations of application network traffic could then passed over this connection, which were FTP [95] traffic and Telnet traffic [96]. FTP traffic was defined as follows

```
set ftp_connection [new Application/FTP]
$ftp_connection attach-agent $tcp_sender
```

Telnet traffic was defined as follows

```
set telnet_connection [new Application/Telnet]
$telnet_connection attach-agent $tcp_sender
```

The alternative to the TCP option is the UDP connection. UDP packets do not guarantee connections [97]. However, they are an excellent option for receiving bulk files where continuous information is specified. Unlike the TCP connections, the UDP equivalents required the use of the *UDP Agent* and a *NULL Agent* objects which represented the transmitter and the receiver. A single UDP example is as follows:

```
set udp_sender [new Agent/UDP]
set udp_receiver [new Agent/NULL]
$ns attach-agent $node0 $udp_sender
$ns attach-agent $node1 $udp_receiver
$ns connect $udp_sender $udp_receiver
```

With UDP as the transport layer, three representations of application network traffic can then be passed over this connection. They are CBR traffic, Exponential traffic or Pareto (on-off) traffic. The traffic is defined as follows:

```
set source [new Application/Traffic/<type>]
```

whereby *type* can be CBR, Exponential or Pareto. A detailed discussion of these traffic generation types is presented in the next subsection.

Generation of the Network Traffic

There are four traffic generation objects for NS-2 which are *exponential*, *Pareto*, *CBR* and *traffic trace*. Exponential, Pareto and CBR have parameters that define the manner in which each of these objects are configured. The traffic trace object receives its data from a TRACE_FILE. This TRACE_FILE represents the network traffic generated from an external source which NS-2 can utilise to create the simulation. Initiating the generation of this traffic was done as follows:

set trace_file [new Tracefile]

$trace_file filename example-trace

Attaching the TRACE_FILE to the source was done as follows:

set source [new Application/Traffic/Trace]

$source attach-tracefile $trace_file

Exponential objects produce on/off traffic where the on and off periods are represented as an exponential distribution. Its parameters are *PacketSize_, burst_time, idle_time, rate_*. Pareto objects also produce on/off traffic but according to the Pareto distribution [98]. Pareto objects have the same parameters as exponential but an additional parameter *shape_* specifies the shape of the Pareto distribution. CBR objects generate traffic at a constant bit rate with the following parameters governing the transmission: *Packet_size_, rate_, interval_, random_ (option to introduce random noise in scheduled times for departure. This setting is either on or off with the default set at off), maxpkts_.*

Attaching the *generator* to the source and then starting the transmission at a specific time (2.0 seconds into simulation) was done as follows:

$generator attach-agent $source

$ns_ at 2.0 "$generator start"

3.3.3.2 Trace Files

When NS-2 executes a simulation with tracing enabled, it produces a TRACE_FILE which presents the network information produced during the simulation process. This represents a detailed log file of the simulation with various headers and details stored. The file contains the packet trace information such as the packet type, packet ID and the type of action performed by the simulator.

As an example to illustrate this information, an extract from a TRACE_FILE (which is part of a packet transmission within a simulation) is shown below*:*

r 0.1 1 2 cbr 1000 - - - - - - - - 2 1.0 3.1 0 0

Table 3.4 assigns variables to each of these trace values, followed by another table that specifies what each value represents in the simulator.

Table 3.4 shows that the traced_line from the simulation represents the receiving of a CBR packet 0.1 seconds into the simulation. This transmission originated at node 1 and was sent to node 2 with a packet size of 1000 bytes of data. The identification number 0 shows that this was the first packet in the simulation. This trace file is instantiated from the simulation script by issuing the following commands sequentially:

set new_trace_file [open out.tr w] and *$ns trace-all $new_trace_file.*

3.3.3.3 The Network Animator (NAM)

NAM is an extension of the NS-2 environment, that uses TRACE_FILE*s* to generate a visual representation of the simulation. NAM produces its own TRACE_FILE which provides a graphical representation of the network simulation. This in turn can provide visual insight into the entire network process. This TRACE_FILE is instantiated from the simulation script by issuing similar sequential commands to the standard simulation TRACE_FILE as follows: *set new_namtrace_file [open out.nam w]* and *$ns namtrace-all $new_namtrace_file.* The first command creates a file and the second command dumps the network animation trace information to the newly created file.

TRACE	r	0.1	1	2	cbr	1000	- - - - - - - -	2	1.0	3.1	0	0
KEY	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$10	\$11	\$12

KEY	VALUE	DEFINITION	TYPE
\$1	action	***r***(Receive),***d***(Drop),***e***(Error),*+*(Enqueue), **-**(Dequeue)	Char
\$2	time	Real-time Value inside of network events	Double
\$3	from	Source Node	Int
\$4	to	Destination Node	Int
\$5	type	Packet Type (CBR,TCP,ACK)	String
\$6	pktsize	Packet Size	Int
\$7	[FLAGS]	8 Advanced flag values (normally not needed)	Char []
\$8	flow_id	ID for the flow of network traffic over a network link	Int
\$9	src	Source Address	Double
\$10	dst	Destination Address	Double
\$11	seq_no	Sequence Number	Int
\$12	packet_id	Unique Packet ID	Int

Table 3.4: Standard Simulation TRACE_FILE Interpretation

These are the different event headers available in NAM along with their respective parameters and their definitions [79].

Initialisation

Certain events take place before NAM can begin animating the simulation. These are denoted by the *-t ** flag within the line (this generally consists of node placements). The first requirement of NAM is to state the minimum **version** (V-flag) of NAM required to animate the simulation. The parameter representation for this line in the TRACE_FILE is as follows:

Example *V -t * -v 1.0a5 -a 0*

FLAG	***V***	*-t*	*-v*	*-a*

FLAG	**Value**	**Example**	**Definition**
-t	<time>	*	Do before simulation begins
-v	<version>	1.0a5	Minimum version of NAM needed for animation
-a	<attr>	0	Extra functionality executed at beginning of NAM

Table 3.5: Version definition for NAM TRACE_FILE

The second requirement of NAM is to state the **hierarchy** (A-flag) addressing scheme to be used by the Animator. There are two settings available: the *default* and the *specific*. This address representation deals with the nature in which the bits are represented in the node structure of the simulation (e.g. multicasting) which can be flat or **n**-level tiers. The parameter representation for this line in the TRACE_FILE is as follows:

Example *A -t * -h 1 -m 2147 -s 0*

FLAG	***A***	*-t*	*-n*	*-o*	*-c*	*-a*	*-h*	*-m*	*-s*

FLAG	**Value**	**Example**	**Definition**
-t	<time>	*	Do before simulation begins
-n	<levels>	1	Addressing levels (in this case flat)
-o	<address-space size>		Total No. of bits used for addressing
-c	<mcastshift>		not used
-a	<mcastmask>		not used
-h	<nth level>	1	Level of the address hierarchy
-m	<mask in nth level>	2147	Determines the address mask
-s	<shift in nth level>	0	the bit shift of determined level in tree

Table 3.6: Hierarchy definition for NAM TRACE_FILE

The third requirement of NAM is to run the **colour table entry** (c-flag) which allows the colouring of packets used by the Animator. This is represented through a colour scheme which must conform to colour name standards used by the standard UNIX environment. The parameter representation for this line in the TRACE_FILE is as follows:

Example *c -t * -i 1 -n purple*

FLAG	**c**	*-t*	*-i*	*-n*

FLAG	**Value**	**Example**	**Definition**
-t	<time>	*	Do before simulation begins
-i	<colour id>	1	Colour ID used to differentiate network traffic
-n	<colour name>	purple	Sets the network traffic to this colour

Table 3.7: Colour Table Entry definition for NAM TRACE_FILE

Nodes

The construction of the nodes within the NAM environment is described. This description specifies how the nodes are placed in the Animation window that includes parameters such as orientation, colour and shape. The parameter representation for this line in the TRACE_FILE is as follows:

Example *n -t * -a 2 -s 2 -S UP -v circle -c orange -i orange*

FLAG	**n**	-*t*	-*a*	-*s*	-*S*	-*v*	-*c*	-*i*

FLAG	**Value**	**Example**	**Definition**
-*t*	<time>	*	Do before simulation begins
-*a*	<src-addr>	2	address of node within the simulation
-*s*	<src-id>	2	ID of node within the simulation
-*S*	<state>	UP	UP or DOWN (down = failure)
-*v*	<shape>	circle	options (box, hexagon) shape of node
-*c*	<colour>	orange	Shape colour of orange
-*i*	<colour>	orange	label colour of orange

Table 3.8: Node definition for NAM TRACE_FILE

Links

This event header describes the construction of the links within the NAM environment and determines how the links between the nodes are specified. An important feature that can assist in the visualisation of the network is the orientation of the nodes. The orientation is determined by the angle between the link and the horizon. However, the orientation is an optional feature so in most cases is not present. Once again, parameters such as colour and shape are also present. The parameter representation for this line in the TRACE_FILE is as follows:

Example *l -t * -s 0 -d 5 -S UP -r 1000000 -D 0.01 -c black*

FLAG	**l**	*-t*	*-s*	*-d*	*-S*	*-r*	*-D*	*-c*

FLAG	**Value**	**Example**	**Definition**
-t	<time>	*	Do before simulation begins
-s	<src>	0	source ID of node in the simulation
-d	<dst>	5	destination ID of node in the simulation
-S	<state>	UP	UP or DOWN (down = failure)
-r	<bw>	1000000	Bandwidth in bps (this one is 1Mb/s)
-D	<delay>	0.01	ms delay for the packet transmission
-c	<colour>	black	colour of this network link

Table 3.9: Link definition for NAM TRACE_FILE

Packets

The_simulated packet trace information is represented in a similar manner to the standard TRACE_FILE. However, there are some changes that need to be specified as NAM needs to maintain specific information such as how the packets are denoted. The majority of the parameters are the same, but new definitions are incorporated for the visual representation of specific network traffic. The parameter representation for this line in the TRACE_FILE is as follows:

Example *h -t 1.01 -s 0 -d 2 -p cbr -e 210 -c 0 -i 1 -a 0 -x {0.0 3.0 -1 - - - - - - - null}*

FLAG	*h*	*-t*	*-e*	*-s*	*-d*	*-p*	*-c*	*-i*	*-a*	*-x*

FLAG	**Value**	**Example**	**Definition**
h	<type>	h	Packet is forwarded to the next hop
-t	<time>	1.01	Time of event occurrence
-e	<extent>	210	Packet size (bytes)
-s	<source id>	0	The source node of the transmission
-d	<destination id>	2	The destination node of the transmission
-c	<conv>	0	conversation ID or Flow ID
-p	<pkt-type>	cbr	Type of Packet (TCP, ACK, SRM)
-i	<id>	1	the Packet ID in the network flow
-a	<attr>	black	colour of this network link
-x	<src-na.pa>	0.0	source node . port address
	<dst-sa.na>	3.0	destination node . port address
	<seq>	-1	sequence number
	<flags>	- - - - - - -	Extra flags for adv-app representation
	<name>	null	colour of this network link

Table 3.10: Packet definition for NAM TRACE_FILE

Queue

This event header represents the number of packets available before the transmission. They are graphically represented as packets stacked on top of each other parallel to the network link. Again, the orientation is available here, but this time it is a mandatory requirement. The parameter representation for this line in the TRACE_FILE is as follows:

Example *q -t * -s 0 -d 1 -a 0.5*

FLAG	***q***	*-t*	*-s*	*-d*	*-a*

FLAG	**Value**	**Example**	**Definition**
-t	<time>	*	Do before simulation begins
-s	<src>	0	Source ID of the node in simulation
-d	<dst>	1	Destination ID of node in simulation
-a	<attr>	0.5	Represents orientation by using (0.5 * pi)

Table 3.11: Queue definition for NAM TRACE_FILE

These tables represent the formatting and parameter specification for NAM to interpret the TRACE_FILE. The interface presents the Animation window which contains various buttons and tools to help represent the animation more accurately:

- Playback buttons to animate the simulation both forward and backward
- Zoom option to either enlarge or decrease the current focus of the network
- Counter representing the time within the simulation
- Step meter to either increase or decrease the speed of the simulation
- Time Ruler representing how far along the simulation the animation has gone
- Buttons to redefine the layout of the topology.

Figure 3.3 illustrates the NAM interface with a simple two node network in the middle of the window. The animation stands at the zero marker, the step meter is set to 2.0 *ms* and the animation has not started yet. The layout of the nodes was pre-defined within the simulation script, hence redefining the layout was not needed. The nodes occupy the majority of the screen, showing how the focus of the network has changed (by using the zoom option).

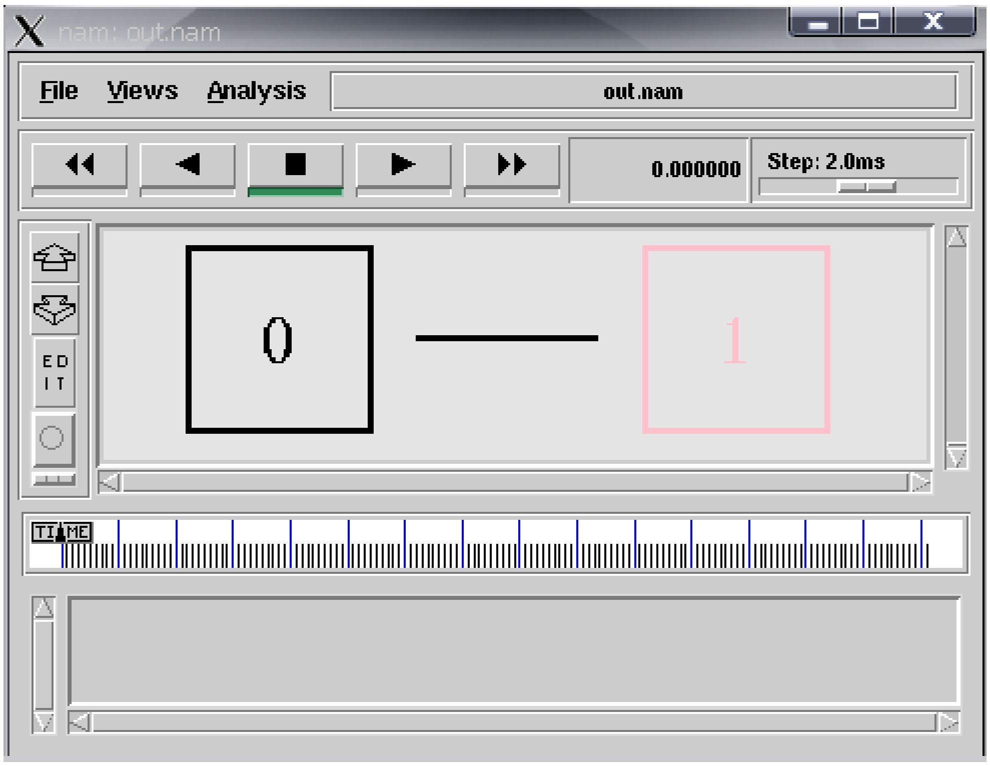


Figure 3.3: NAM window

Now that the features of NS-2 have been elaborated, a complete simulation example that demonstrates how a simulation script is constructed from top to bottom, is provided in Appendix A. Evaluating a simulation tool generally entails the construction of a physical network and comparing the physical trace data against the simulated environment. However, there was no application for video in this NS-2 environment so a comprehensive investigation into video support led us to a tool-set that allows for the evaluation of video over networks.

3.3.4 Evalvid - Evaluation Tool-set and Video Publishing Tools

Evalvid is a tool-set which allows for the evaluation of video quality over both physical and simulated networks [47]. Evalvid has been used extensively in network video quality research. With the capacity to evaluate both simulated and physical networks, Evalvid can be used to bridge the evaluation gap between these two networking environments.

Since NS-2 does not provide any support to evaluate video quality, some publications have investigated and implemented Evalvid into the NS-2 development design for video quality evaluation support [99,100]. The focus of the publication [99] was on the complete tool-set while this second publication [100] focused on the Variable Bit-Rate (VBR) video with less emphasis on the NS-2 implementation.

The definitive feature of Evalvid is the capacity to provide a visual quality evaluation because the tool-set takes in a video source and produces a destination video with all the faults and errors incurred through the network transmission. However, if the video quality depreciation is too insignificant for the casual observer to distinguish, then the accompanying tools from Evalvid provide a means to detect the difference more accurately and to apply various metrics to separate the different video statistics (such as signal noise, packet loss).

The Evalvid tool-set can be subdivided into three sections: the tools for video encoding/decoding (video publishing), the tools for network evaluation, and the tools for video evaluation. Figure 3.4 illustrates the standard Evalvid tool-set with the colour schemes representing the different sections of the tool-set [101]. The blue squares represent the Evalvid specific tools utilised to produce the video quality evaluation. The red section represents the Network area which can either be a physical network or a simulated network [102]. The network representation in this instance is through the use of TRACE_FILEs which are either generated through NS-2 or inserted from an external source such as a physical network.

TCPDUMP (recently changed to Wireshark) is a network traffic analyser capable of interpreting different network protocols and dumping the packet information to a source file [103]. This source file can be used as a source trace file for the network implementation as is shown within the network area Application Programming Interface (API). The green represents trace file transmissions between the various parties. Both the video decoder and encoder are tools used to manipulate the video in order to significantly decrease the overhead on the network, with the resulting video stream shown in yellow varies according to the location of the video stream.

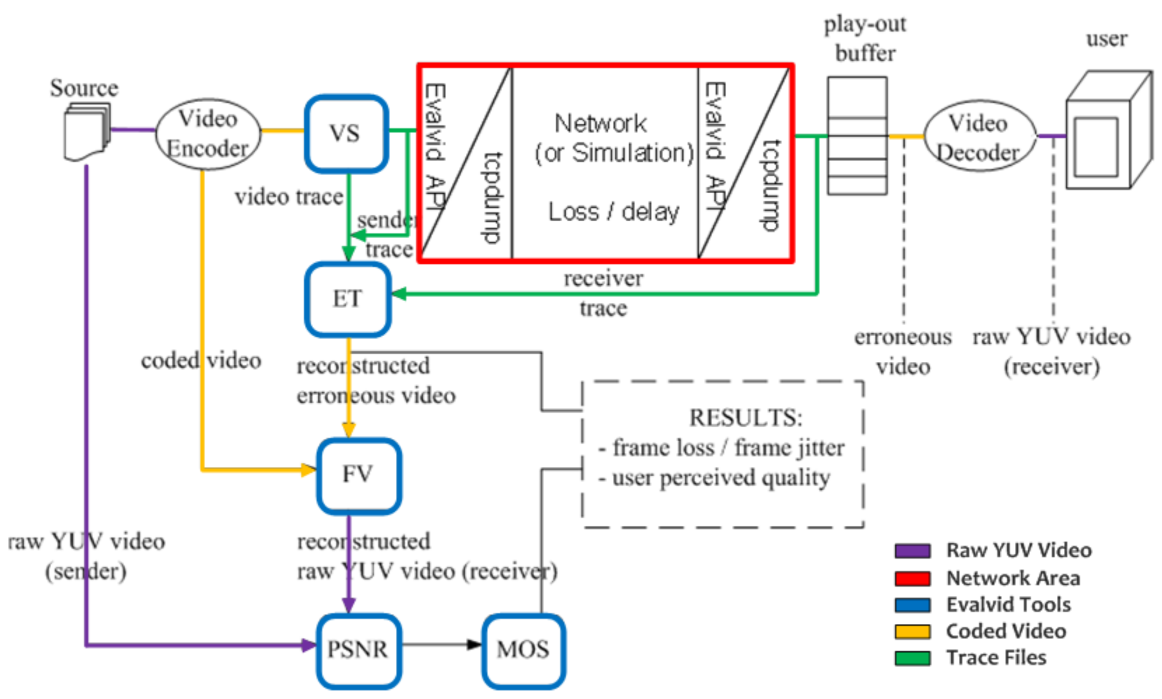


Figure 3.4: Evalvid Tool-set extended from [99]

The *coded video* between the video encoder and the FV tool, is a reference point which can sometimes be used to compare the output from the *play-out buffer* and the video decoder (at this stage, the erroneous video has experienced possible loss). This position is also a good place to evaluate video from a coded point of view because encoding and decoding does impart an element of loss into the output video no matter how accurate the compression is.

Figure 3.5 illustrates a detailed representation of integrating Evalvid with NS-2. The model flow represents a similar direction to Figure 3.4 with a closer view of the NS-2 environment. It also illustrates the various components that are required to interface Evalvid with NS-2. Within the NS-2 environment, there are five objects that are used to interface NS-2 with Evalvid. Of these five, three are new simulation Agents called: *MyTraffic-Trace, MyUDP,* and *MyUDPSink* (located within the NS-2 environment and represented by dot-dashed elements in Figure 3.4) [47, 99, 101]. The other two objects represent the ends of the simulated network that serve as collectors of the network information. They produce the new TRACE_FILEs which are used to generate a new representation of the output video using one of Evalvid's evaluation tools.

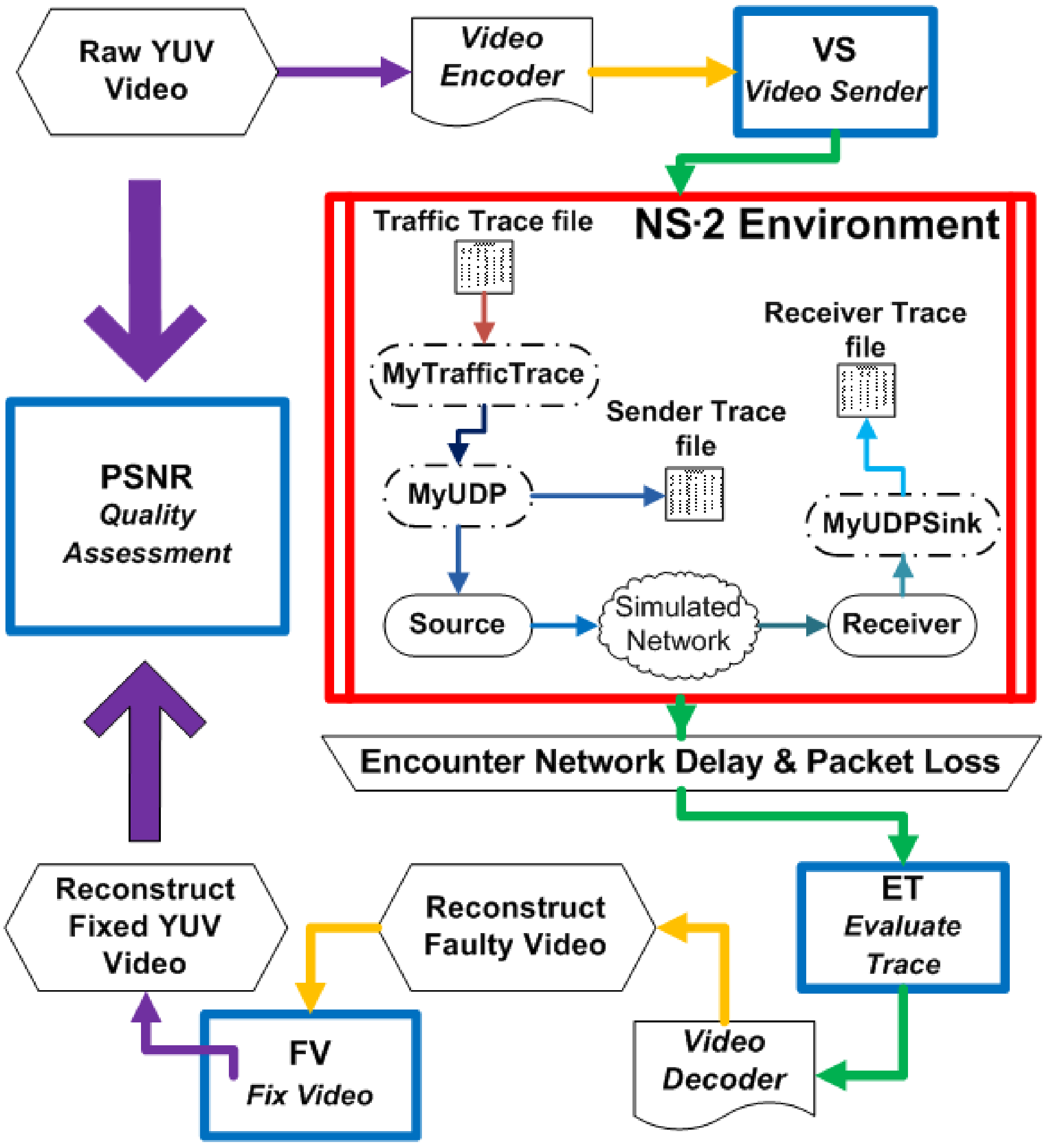


Figure 3.5: Evalvid Integration into NS-2

3.3.4.1 Tools

This section provides a short description of the tools which constitute the Evalvid tool-set.

VS The Video Sender (VS) tool is responsible for transmitting the compressed video across the network (real or simulated) by fragmenting the video into smaller segments. These segments represent packets of video data which this tool then transcribes to a TRACE_FILE which contains details about the transmission (such as *timestamp*, *packetID*, *packet size*) [47]. VS also creates a separate video trace file containing details for each video frame within the video file, which can later be used for evaluation.

ET The Evaluate Trace (ET) is the core component in the Evalvid framework [99]. This tool examines the various trace files gathered from the transmission process (*sender* TRACE_FILE, *receiver* TRACE_FILE, *video* TRACE_FILE) with the encoded video and compares the trace data between the various components (*sender* TRACE_FILE, *receiver* TRACE_FILE, *video* TRACE_FILE). From the comparison, the tool can provide information on lost packets, network delay, jitter and most importantly a reconstructed representation of the compressed transmitted video.

FV *In order to* provide an accurate comparison between two videos, the Fix Video (FV) tool is necessary as it provides the most accurate means to compare the videos frame by frame. However, the ET tool discards frames if they arrive after their designated playback time which effectively designates them as lost frames. These lost frames make it difficult for a comparison to be drawn on a frame by frame basis, hence this tool conceals these difficulties by inserting the last successfully decoded frame in the place of the missing frame [99]. This tool effectively ensures that the final video has the same number of frames as the original video.

PSNR The Peak Signal Noise Ratio (PSNR) is a scientific/mathematical formula used to measure quality of service by determining the signal noise measured in dB (decibels). In most video quality evaluations, the luminance component of the image/frame (Y), the source video frame (S) and destination video frame (D) are common metrics for video evaluations. The following formula is the recommended calculation for this environment which once again focuses on the luminance value Y of the video characteristics [47, 104, 105].

$$PSNR(n)_{\text{dB}} = 20log_{10}\left(\frac{V_{peak}}{\sqrt{\frac{1}{N_{col}N_{row}}\sum_{i=0}^{N_{col}}\sum_{j=0}^{N_{row}}[Y_S(n,i,j)-Y_D(n,i,j)]^2}}\right)$$

$$V_{peak} = 2^k - 1$$

$$n = Video\,frame$$

$$V = Maximum\,possible\;\,pixel\,value$$

$$N_{col} = Number\,of\,column\,pixels\,within\,frame$$

$$N_{row} = Number\,of\,row\,pixels\,within\,frame$$

$$k = number\,of\,bits\,per\,pixel\,(luminance\,component)$$

$$i,j = pixel\,combination\,for\,frame$$

MOS The Mean Opinion Score (MOS) is a subjective measurement specification prescribed by the developers to provide video quality evaluation of the actual video produced [47]. This is a subjective perception with a rating 1-5 with the lower number describing the poorest quality whilst the highest denotes the best quality [104]. The ratio is governed by the average peak signal to noise ratio affecting the final video and is provided as follows:

PSNR	MOS	Verdict
> 37	5	Exceptional
31 - 37	4	Decent
25 - 31	3	Average
20 - 25	2	Mediocre
< 20	1	Unacceptable

Table 3.12: Suggested PSNR to MOS conversion [47]

3.3.4.2 Video Publishing Tools

Several publishing tools are available for the compression of raw YUV formatted videos. Evalvid provided some of these video compression/decompression tools. However, these tools also have to be compatible with the appropriate Evalvid Video Sender tool (VS) which is able to transmit the compressed video through a network (simulated or non-simulated). The following files represent the various publishing tools utilised by CAFSS-Net:

mpeg4Encoder.exe Compresses the raw YUV video to a compressed mpeg4 format

MP4.exe Sends the compressed mpeg4 file through the network

mpeg4Encoder.exe Decompresses the mpeg4 file back to raw YUV video format

A comprehensive online citation provides an informative tutorial to effectively append Evalvid to an existing installation of NS-2 [101]. This is accompanied by another tutorial from the same author that provides the original extension of NS-2 to incorporate the Evalvid API [102]. The second tutorial has previously been explored. However, the first tutorial yielded some recommendations for the integration of Evalvid into the NS-2 environment. One of these recommendations was that Windows be the operating system of choice to better interact with the necessary tools.

Even with all the necessary tools and network specifications, a media stream is meaningless without the source video. The next section details the specifics about the videos and what characteristics governed their selection.

3.3.5 Videos

There are several video compression and decompression tools available in the streaming media environment. CAFSS-Net consists of a limited collection of videos that allow for the testing of different media streaming conditions. Since the quality of streaming is significantly dependent on the medium being transmitted [8], CAFSS-Net aimed to incorporate specific videos to explore majority of the various streaming media conditions. The three videos discussed are presented in a low quality standard (QCIF) and a high quality standard (CIF). Hence there are a total of six videos. The following paragraphs outline the raw YUV videos used in CAFSS-Net:

Akiyo Video

This video represents a news broadcast where the movement within the video is relatively low. This motion within the this video is perceived to be relatively invariant and can be justified by the non dramatic changes from frame to frame. The camera position capturing this scene remains fixed and only the object (the news reporter) within the scene moves slightly.

Foreman Video

This video represents a construction worker supervising a construction site, where the movement in this video was significantly greater than that of the Akiyo video. The camera position capturing this scene is erratic and pans around the construction site. Here, there was a higher degree of change between video frames.

Hall Video

This video represents two employees walking within an office hall. The camera position capturing this scene remains fixed but the two employees provide movement within the scene. This video is a mixture of the significant movement of the objects in scene like the Foreman video and the static nature of the camera position like the Akiyo video.

3.3.6 CAFSS-Net Review

This subsection discusses the various components of CAFSS-Net and also presents the numerous tools that were utilised during the study. The existing streaming media distribution engines explored (RealNetworks [60], Microsoft [56], Apple [59]) are all software-based platforms aimed at streaming media across networks where the network consists of at least two nodes. These streaming media distribution engines either generated their own source videos or streamed pre-recorded videos to their respective counterparts.

CAFSS-Net encompasses both these features (a network of at least two nodes and the streaming of pre-recorded videos) and allows for a more controlled environment where one can examine the results of changes to variable streaming conditions. CAFSS-Net consists of the following components which also serve as the foundation for the test bed Section 3.4:

Videos were not captured live from an external source. They had specific characteristics to allow for greater comparison between various changes to the network configuration. Hence the videos adhered to the following specifications:

- The tests were based on six videos which were:
  - **Akiyo CIF** and **Akiyo QCIF**
  - **Foreman CIF** and **Foreman QCIF**
  - **Hall CIF** and **Hall QCIF**
- The videos were grouped into three degrees of motion as follows:
  - **low motion -** Akiyo CIF and Akiyo QCIF
  - **medium motion -** Hall CIF and Hall QCIF
  - **high motion -** Foreman CIF and Foreman QCIF

- The quality standards were specified as CIF for the high quality video and QCIF for the low quality equivalent
- The quality of the video was directly related to the video resolution.
- All videos were uniform in length with 300 frames
- All videos operated at 30 fps and hence each video was 10 seconds long

NS-2 The initial installation was based on a Linux platform, but was later ported to Windows. The final NS-2 configuration was as follows:

- Wireless network implementation was available in NS-2. However, for the purpose of this study, it was assumed that all wireless nodes exhibited the same characteristics as normal network nodes with the exception that there was greater potential for signal noise and for lost packets
- The NS-allinone version 2.28 package which contains the various components of the simulator was used
- Since NS-2 is UNIX based, Cygwin was used to port NS-2 into the Windows environment
- Any caching mechanism will be incorporated into simulation at run-time, and no external caching/proxy systems shall be implemented

Evalvid The initial tool-set comprised of various tools and applications that allowed for the investigation of video transmitted through networks. Various attempts to integrate Evalvid into the existing NS-2 environment presented difficulties, due to the differences between the versions of applications. The Evalvid resource site presents many codecs and tools for various testing conditions. Even though the Evalvid tool-set uses various video formats for encoding and decoding, we only implemented the MPEG-4 equivalent to interact with CAFSS-Net efficiently. The following tools were utilised from the tutorial package obtained from [101] and hence differ from the original Evalvid sources:

- Myevalvid2 is a set of C++ extensions used to interface Evalvid with NS-2 (objects representing new Agents for the simulation; *MyTrafficTrace, MyUDP, MyUDPSink)*
- *.*par* were parameter files used to compress raw video files into MPEG data format
- **mpeg4encoder** was the encoder used to compress the raw YUV formatted video into a compressed file. It runs in conjunction with the parameter file (**.par*) which governed how the compressed video output should be represented
- **MP4.exe** was the Video Sender (VS) tool from the Evalvid tool-set
- **ET.exe** was the Evaluate Trace (ET) tool from the Evalvid tool-set
- **mpeg4decoder.exe** was the decoder used to decompress the mpeg4 formatted video into a raw YUV video to represent the final video after being transmitted
- **PSNR.exe** was the quality evaluation tool (PSNR) from the Evalvid tool-set
- **MOS** was reproduced through a new script based on the data from the PSNR.exe tool

Figure 3.6 displays the complete overview of CAFSS-Net with the six components highlighted into the three major components using the following colour scheme: **blue** represents the videos, **red** represents NS-2 and **green** represents Evalvid. The complete framework will be evaluated in Chapter 5.

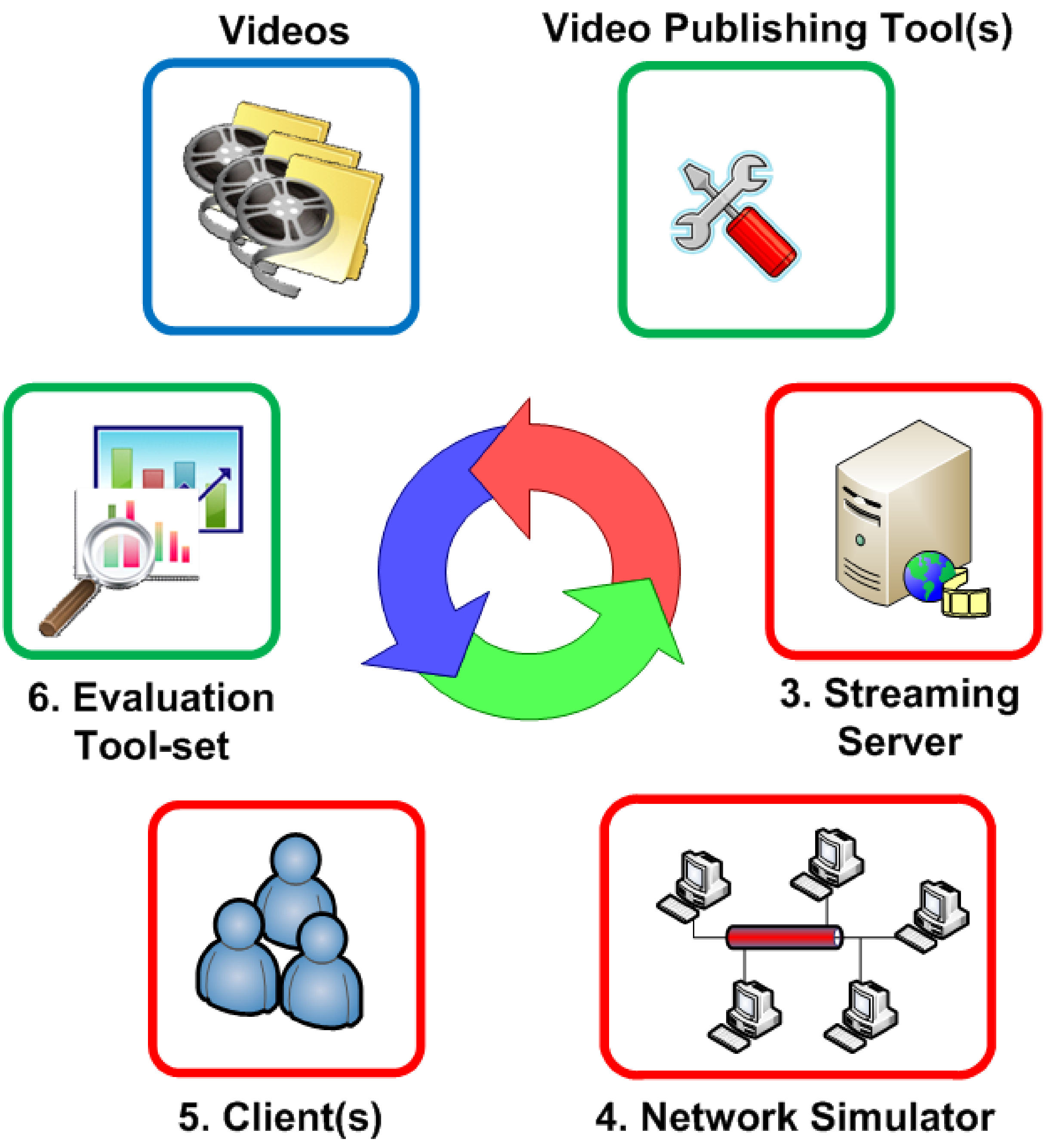


Figure 3.6: CAFSS-Net overview

.

3.4 Testing Scenarios

The testing scenarios represent three types of networks according to their degree of utilisation. The three types of scenarios are: Peer-to-Peer, Small Network and the Large network scenario. These scenarios make up the network configurations which will be simulated by NS-2. In addition, the same three scenarios will be compared with certain prescribed conditions in CAFSS-Net. The prescribed conditions set on all testing scenarios were:

- The server played a single video stream
- The video stream was sent from start to finish with no breaks in between.
- Traffic flowed from the server to the clients

3.4.1 Peer-to-Peer Scenario

This test consisted of a simple client and server scenario where only two nodes are present in the network. The one node transmitted the video stream (the server) whilst the other node received the transmission (the client). This test was designed to mimic a traditional unicasting environment where there was minimal interference from third party network traffic. A physical network representation for this scenario was also implemented in order to further verify the accuracy of CAFSS-Net. This test was also used to calibrate the simulator. The illustration in Figure 3.7 displays the scenario configuration. This scenario is designed to test the following bandwidths represented in Figure 3.7 which are specified in Table 3.13:

Bandwidth Colour	Network Bandwidths
RED	1000Kb/s, 512Kb/s and 384Kb/s
BLUE	256 Kb/s, 128Kb/s and 64Kb/s
GREEN	56Kb/s, 28.8Kb/s and 9.6Kb/s

Table 3.13: Bandwidth colour representation for Figure 3.7

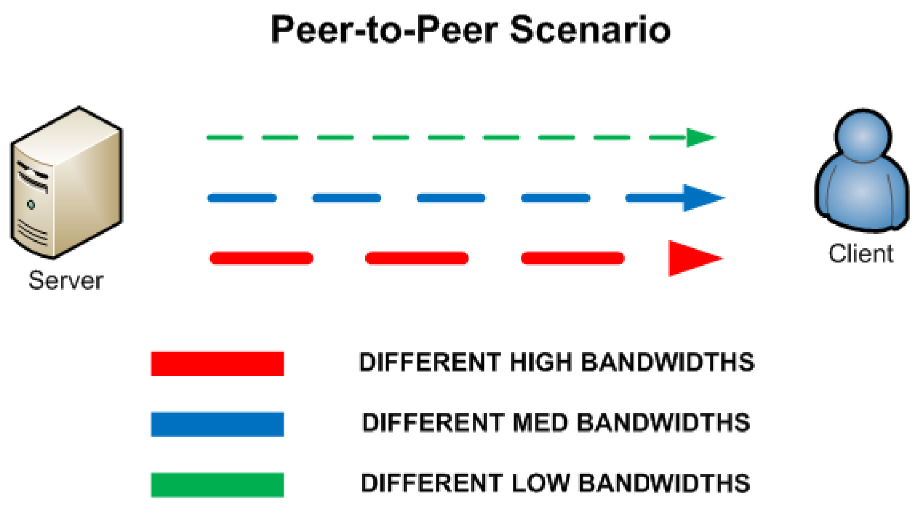


Figure 3.7: Peer-to-Peer Test

3.4.1.1 Test Measurements

The bandwidths in Table 3.13 will be used in this scenario to perform the following tests for the Akiyo CIF, Foreman CIF and Hall CIF Videos:

- The Latency measured in *ms*
- The Frame loss measured as a percentage
- The Jitter measured in *ms*
- The PSNR measured in *dB* with a reference video comparison
- The Average PSNR values with the standard deviation
- The MOS measured with a rating 1-5
- The video Quality showing the final video transmission
- The Cache size measured in *KB*

3.4.2 Small Network Scenario

With reference to Figure 3.8, the normal network connections (illustrated in red) were high bandwidth connections that remained fixed at a comfortable 10 Mb/s rate to ensure that the video stream was not affected. The medium points of interest (illustrated in dashed green) all had at one time, one of the three medium network bandwidth speeds (64Kb/s, 128Kb/s and 256Kb/s). The network connection with the high point of interest operated through the following range of network bandwidth speeds (56Kb/s, 128Kb/s, 384Kb/s, 1000Kb/s). These network connections represent the connection standards available to networking infrastructures (Dial-up, ISDN, ADSL and T1 respectively).

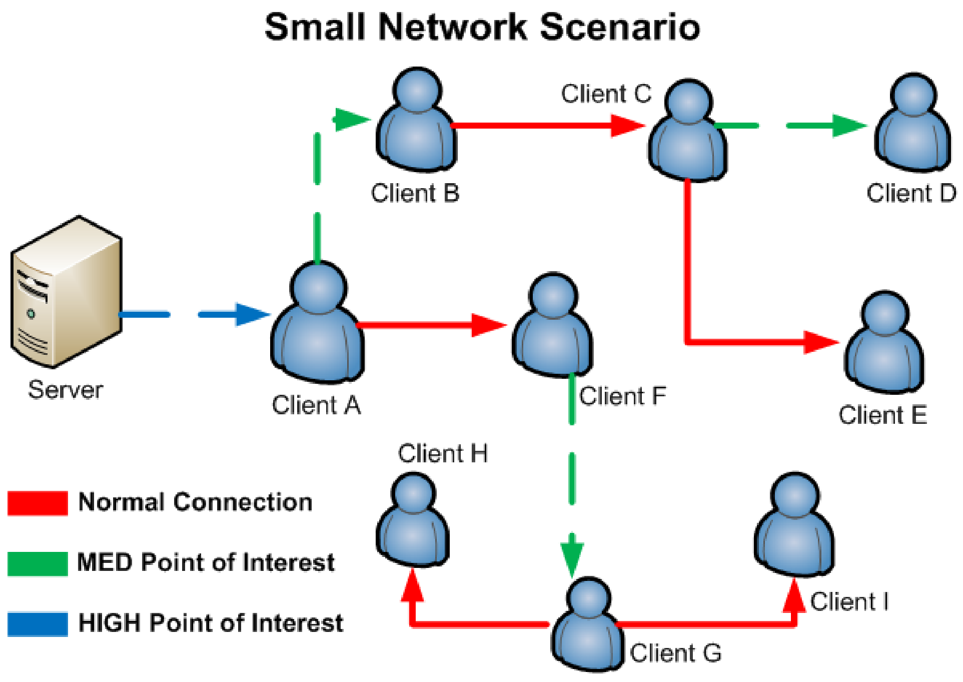
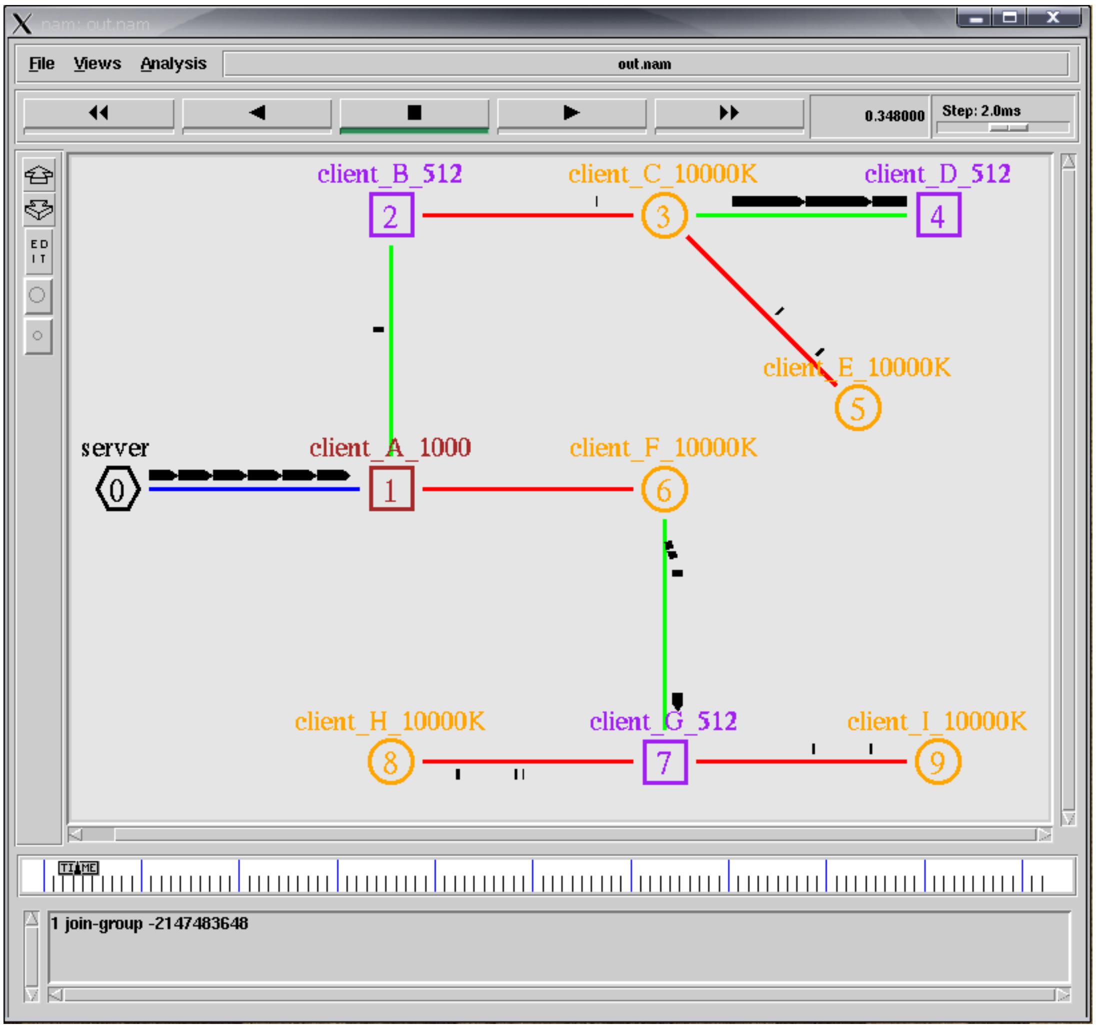


Figure 3.8: Small Network Test

The following Figure 3.9 illustrates the Network Animation Tool (NAM) simulating the small network scenario presented in Figure 3.8. From the diagram, one can see the animation playback options at the top, with the simulation time-stamp (displaying the current simulation time at 0.348000 into the simulation) and speed control located on the right. The simulation distance marker is located at the bottom, with the animation presented in the middle. As previously mentioned, the results of these simulation tests only focus on the squared nodes, with the brown node 1 (client_A) being the primary focus of this investigation and the other three nodes 2, 4, and 7 (clients B, D and G) providing secondary points of interest.



1 join-group -2147483648

Figure 3.9: Small Network Scenario simulation animation

There are three sets of tests within this Small Network scenario as follows:

1. The physical network tests

2. Cache Comparison

3. The Difference in Placement

1. The physical network tests

These tests consist of the Latency, Frame Loss and Jitter tests as previously performed in the Peer-to-Peer scenario with the exception being that the tests are performed on a physical network. Only a small section (the connection from the Server to Client B as shown in Figure 3.8) was examined through a physical network representation to determine the accuracy of CAFSS-Net.

2. Cache Comparison

This test was designed to compare the cache size with respect to the changes of bandwidth between two consecutive nodes within the network. Thus from Figure 3.8, transmission from Server to Client A produces a cache size a. Transmission from Client A to Client B produces a cache size b. What is the difference in the cache sizes between a and b when the bandwidths from the Server to Client A and the bandwidths from Client A to Client B change? Further details for this test are presented in the next chapter.

3. The Difference in Placement

It was designed to mimic small networks where there are numerous nodes receiving traffic from the same source. This is equivalent to a video stream distribution to numerous clients on the network. This test was designed to explore the multicasting capability of CAFSS-Net and to answer the following question:

Does the placement of the node in the network affect the cache size if the links between the nodes are uniform?

The data will be collected from clients A, B, D and G as shown in Figure 3.8.

3.4.2.1 Test Measurements

The data collected in this scenario with its relevant bandwidth specifications was aimed to perform the following tests for the Akiyo CIF, Foreman CIF and Hall CIF Videos:

- The Frame loss for this non-simulated network measured as a percentage
- The Latency for this non-simulated network measured in *ms*
- The Jitter for this non-simulated network measured in *ms*
- The Cache Comparison of the simulated network measured in *KB*
- The Placement Difference of the cache nodes in the simulated network measured in *KB*

3.4.3 Large Network Scenario and Prediction

This final experiment involved a wide distribution where there are three networks and a large number of nodes. Unlike the Small Network tests, this scenario was only implemented through CAFSS-Net. The aim of the study was to test the impact of external (or foreign) network traffic on the video stream over a small, medium and large network. This test involved three clients with one in each network. The first Client resided within a Small Office, the next Client in an Administration Building and the last Client on the Internet. Figure 3.10 provides a representation of the network topology.

This Large Network scenario represented in Figure 3.10 was designed to test the following bandwidths which are as follows: **red** 100 000Kb/s, **green** 1000Kb/s and **blue** 10 000Kb/s.

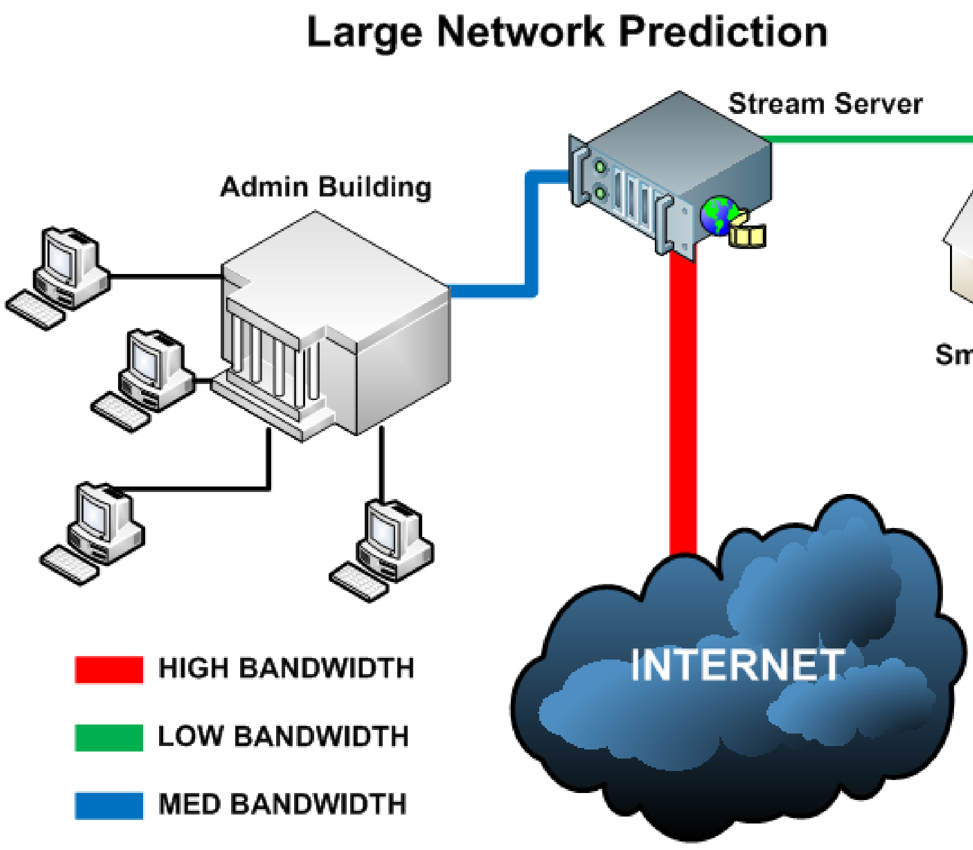


Figure 3.10: Large Network Prediction

The previous two sections consisted of controlled environments where the traffic being transmitted through the network only consisted of the video stream. The Large Network prediction tests consisted of video streams being transmitted in the presence of other external network traffic. In some test instances, the external network traffic is directed to the video stream Client nodes specifically to ascertain the impact on the video transmission.

A topological network representation of the simulation is provided in the Figure 3.11. Each of these categories (office, admin and Internet) represents the degree of external network traffic imposed on the nodes. Each network has a specific node that is used to obtain the results for these tests (Clients A, B and C in Figure 3.11).

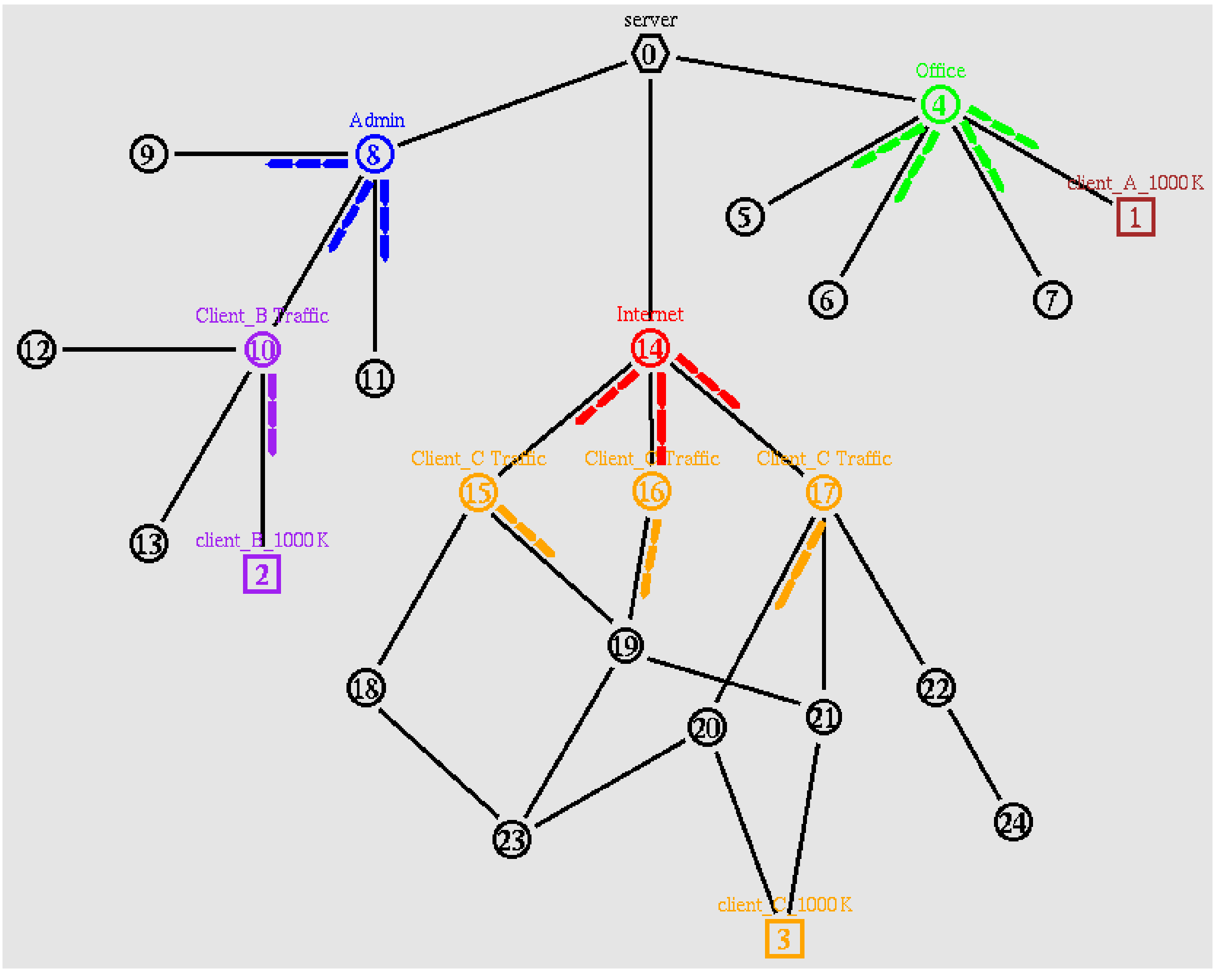


Figure 3.11: Big Network Scenario topology simulation animation

The Small Office network consists of external traffic originating from the office server (node 4 represented in green) with Client A present. The Admin network not only produced external traffic from node 8 (represented in blue), but also contains a specific traffic generator designed to add more noise to the Client B (node 10 represented in purple).

The Internet network consists of a web server (node 14 represented in red) that forwards traffic to its network. The Internet network has three separate traffic generators designed to add noise to Client C (the circle nodes 15, 16 and 17 represented in orange).

3.4.3.1 Test Measurements

The red, blue and green bandwidth representations within this scenario will be applied to the following tests for the Akiyo CIF, Foreman CIF and Hall CIF Videos:

- The Latency with extra network traffic measured in *ms*
- The Frame loss with extra network traffic measured as a percentage
- The Jitter with extra network traffic measured in *ms*
- The Average PSNR values including the standard deviation with extra network traffic
- The MOS with extra network traffic measured with a rating 1-5

3.5 Summary

This chapter evaluated a number of existing streaming media systems and noted that even though there are numerous streaming media systems available, the chosen three present similar strategies and common functionality to effectively provide media streaming to the network. These concepts were used as a foundation for CAFSS-Net. This chapter then delved into CAFSS-Net, describing the major components with their features. The components consisted of the streaming server, the client(s), the network simulator, the video publishing tool(s), videos and the evaluation tool-set. These components were grouped into three sections (videos, NS-2 and Evalvid).

This chapter then provided detailed information to CAFSS-Net's network simulation component which was NS-2. NS-2 was the option of choice due not only to its open-source nature and free licensing, but also to its active community involvement and access to its kernel source code which can be extended to suite user's needs. Two other components of CAFSS-Net are incorporated into the network simulation tool which can represent both the streaming server and the client(s) of the network.

The remaining two major components of CAFSS-Net were then described. These consisted of Evalvid (the evaluation tool-set which included the video publishing tools: the information needed to provide the video streaming distributions) and the videos to be streamed. Section 3.3.6 then provided an overview of CAFSS-Net and highlighted the important details of the various components.

Section 3.4 concludes with the Test Bed design in which three different testing scenarios were proposed. The first scenario was the Peer-to-Peer network and it consisted of two nodes. The second scenario was a Small network consisting of three sets of tests: physical networking , cache comparison and node placement. The third scenario involved a Large network and introduced extra network traffic.

Chapter 4

Results

4.1 Introduction

The research thus far has explored the literature in the field and has also provided a detailed description of the design of CAFSS-Net. The previous chapter consisted of the configuration and specifications for the various components of CAFSS-Net followed by the different scenarios which CAFSS-Net will explore.

This chapter begins by presenting the section on the data collection process. This section describes the details of the network statistical measurements (jitter, packet loss, latency) and a short introduction to the data collected. This chapter then continues to provide results for each of the test scenarios: the simple Peer-to-Peer test, Small Network test and the Large Network prediction.

Section 4.3 provides the results of the Peer-to-Peer scenario which presents the findings of the different bandwidth tests over a client-server architecture. The Small Network scenario is presented in section 4.4. It provides the results of the three sets of tests which are: the physical network tests, the cache comparison tests and the differences in node placement tests. The Large Network results in section 4.5 focus on the effects of external network traffic on specific nodes within their different networking domains. This last section serves as an example of a network prediction because it focuses on the network simulation component of CAFSS-Net and no physical network tests were done.

4.2 Data Collection

4.2.1 CAFSS-Net Output

CAFSS-Net operates in a controlled environment which is customisable to suit various testing conditions. Evalvid provides network traffic analysing tools that can provide information on the performance of the network as well as the quality of video stream. This subsection describes the various tools and also details how they affect the quality of the video stream and the statistics of the network traffic. However, since these tools and graphs are all interpreted from the output data of TRACE_FILEs, details about the different TRACE_FILEs that are generated through CAFSS-Net operations will be provided.

4.2.2 Tracing Data

Subsection 3.3.3 explored the structure of a simulation TRACE_FILE and examined the various flags and features attributed to a single line within this TRACE_FILE. The chapter also demonstrated that various flags represent details that can provide information about the delivery of a packet transmission. By utilising the structure of these TRACE_FILE's, one can analyse the data and produce statistical network analysis based on the collection of the packets transmitted. The following subsections presents the initial attempts at obtaining the statistics for the network traffic analysis from the TRACE_FILES.

4.2.2.1 Packet Loss (Dropped and Lost)

Packet loss consists of either dropped or lost packets. An example of packets being dropped is in a congested network where there may be insufficient bandwidth to cater for all the transmitted packets. General packet loss can occur simply by packets getting lost during transmission (temporary loss of a network connection) and both lost and dropped packets decrease the quality of a network transmission (which can decrease the quality of the video stream). The following four lines were extracted from an example TRACE_FILE and represent the simulated transmission of two packets.

```
+ 0.1 1 2 cbr 1000 ------- 2 1.0 3.1 0 0

- 0.1 1 2 cbr 1000 ------- 2 1.0 3.1 0 0

+ 0.108 1 2 cbr 1000 ------- 2 1.0 3.1 1 1

- 0.108 1 2 cbr 1000 ------- 2 1.0 3.1 1 1
```

By examining the *NS-2 Manual* [79] and interpreting the various flags used in the trace, a simple program was written that runs through the whole TRACE_FILE and identifies lost packets. The program consists of the following steps:

- counts the total number of packets sent
- counts the packets dropped unique to a specific type of packet
- counts the total number of dropped packets.

The NS-2 tutorial recommended that an in-built Linux programming language, which deals specifically with text manipulation (AWK) [106], be used for constructing these tools. Gawk is an open-source text manipulation tool available in most Linux programming environments. It uses a simple programming structure with two optional sections:

- Begin {Sets up all variables and is a place to initialise values} (OPTIONAL)
- {This is the main section, where the language applies the defined functions to each line in the input file. Global variables can be accessed here and manipulated}
- End {This is where one can do the necessary cleanups and any last minute tests } (OPTIONAL)

Algorithm 1 *Packet Loss Algorithm*

```
Set total_Dropped_Packets = 0
Set CBR_Packets = 0
Set TCP_Packets = 0

For each line tracefile do
{
    If TCP_Packet AND dropped flag = TRUE
        ( Increase TCP_Packets AND Increase Dropped_Packets)
    If CBR Packet AND dropped flag = TRUE
        ( Increase CBR_Packets AND Increase Dropped_Packets )
}
Print (TCP_Packets, CBR_Packets, total_Dropped_Packets).
```

Using this tool, Algorithm 1 was devised.

The *gawk* command (a command based AWK implementation that can read AWK scripts) invokes a written application and applies it to a particular input file. The following represents an application of the packet loss script (Appendix B) applied to an example simulation which yielded the following results:

```
$ gawk -f packet_loss.awk out.tr

number of packets sent:550

number of video packets lost:2

number of CBR packets lost:8

number of TCP packets lost:10

number of total packets lost:20

Packet Loss percentage:  3.63%
```

4.2.2.2 Network Latency

This is defined as the time taken for a packet of data to get from a designated node to another [18]. What is needed is to take the difference between the packet's arrival and its departure time, and essentially that would give the latency. Unfortunately the only details within the TRACE_FILE with respect to time are the real-time value of network events for each packet, hence the need to manually determine these values from the real-time events and associate them to start times and end times. Algorithm 2 (similar to the packet loss Algorithm 1) was formulated by utilising another AWK script (Appendix C).

Algorithm 2 Latency Algorithm

```
Set highest_packet_id = 0
For each line tracefile do
{
    Check for the highest_packet_id
    Get new start_time for packet only if it's a CBR packet and not dropped and if It's a receiving packet
    OTHERWISE set a flag to show a packet not to be checked
}
For each of the Collected Packets
Have the End time minus the Start time
If the start time is > end time.. then DON'T print out the packet (from the set flag)
```

The following represents output generated from running the program script (since the list is very long, this output only shows the first 5 packet latencies in seconds:

```
Packet ID: 0, Start time: 0.100000, Latency: 0.038706
Packet ID: 1, Start time: 0.108000, Latency: 0.038706
Packet ID: 2, Start time: 0.116000, Latency: 0.164439
Packet ID: 3, Start time: 0.124000, Latency: 0.224342
Packet ID: 4, Start time: 0.132000, Latency: 0.238706
```

4.2.2.3 Jitter

Jitter has varying definitions particularly with video streaming. One definition has already been given in Chapter 2 where jitter was defined as the fluctuation of delay from packet to packet (or the out of time packets/frames, decoded at same rate) [20]. In this instance a packet/frame arrives ahead of its predecessor or arrives later than its scheduled arrival time. Another definition is taken from the RTP RFC1889 which defines jitter as the smoothed function of delay differences between consecutive packets over time [107]. A popular definition [69, 108–110] states that jitter is the variance in delay. Since there are varying definitions of this feature, there are consequently variable representations of jitter.

Our initial algorithm investigated the representation from [109] which defines jitter as the following:

$$Jitter_i = Delay_i - Delay_{i-1}$$

$$i = network\,packet$$

However, this algorithm may be inaccurate when there is a constant difference between the network packets which would in effect state that there is zero jitter, but in actual fact may just be a constant delayed video. A more comprehensive solution was identified in Algorithm 3. This was implemented as a AWK script and produced the jitter representation [111].

Algorithm 3 Jitter Algorithm

```
Set highest_packet_id = 0
Check for the highest packet ID
Store the Starting time for a reference point for each packet
Store the Sequence # that corresponds to the each packet
Store the Receiving time for the CBR packets
For (each packet till the highest_packet_ID)
{
    If (Start time < end time)
    {
        Compare the sequence # with previous sequence #
        Compare the difference in the delay
        If (No difference in sequence #)
        {
            No Jitter
        }
        Else
        {
            Jitter = Difference in Delay / Difference in Sequence #
        } Print out the Packet Jitter
    }
}
```

However, results obtained using this method did not cater for packets arriving out of time (when start time > end time) which can still cause video jitter. Fortunately, Evalvid provided a relatively versatile tool for calculating jitter using the formula that follows below [99] :

$$Jp = \frac{1}{N}\sum_{i=1}^{N}(it_i - it_N^{\sim})^2$$

$$it_{P_0} = 0$$

$$it_{P_n} = t_{P_n} - t_{P_{n-1}}$$

$$it_P = \textit{inter-packet time}$$

$$it_N^{\sim} = average\, of\, \textit{inter-packet times}$$

$$N = number\, of\, packets$$

$$t_{P_n} = \textit{time-stamp}\, of\, packet\, number\, \mathbf{n}$$

The definition for jitter in this instance provides a better representation for the possible network analysis statistics because it maintains a running average of the jitter representation and caters for packets arriving out of time. The other measurement tools such as PSNR and MOS have already been developed from the Evalvid tool-set and therefore there is no need for constructing necessary scripts.

4.2.3 Frame Results vs Packet Results

All the videos consist of 300 frames. However, when the video is packetized and transmitted across the network, the number of packets transmitted depends on the quality, size and the content of the video. The Evalvid has the ability to produce both network packet results and video frame results. In order to compare these videos more accurately, the results presented within this chapter are based on the video frame results.

4.3 Peer-to-Peer Results

4.3.1 Latency

The latency in Figures 4.1, 4.2 and 4.3 consists of the end to end delay between the video frames. This latency measure is similar to the network packet latency measurement and the Evalvid tools provide measurement traces for both video frames and the network packets. These videos all produced 300 frames of transmitted data as represented by the *x-axis*. The *y-axis* represents the latency in milliseconds *ms* and has been set to 8.0 *ms* for all three videos to illustrate the comparison between them. The different bandwidths represent common connections available within the general networking environment.

These graphs consist of nine different bandwidth tests represented by the different coloured lines. The results for the Akiyo CIF video (Figure 4.1) show the lower bandwidth streams (G, H and I) exhibiting high latency times (as seen by the tall latency peaks) whilst the higher bandwidth streams (A, B and C) exhibit low latency times, remaining close to the *x-axis*. The medium bandwidth streams (D, E and F) present latency times somewhere in between. Though not clearly visible within this graph, the 9,6Kb/s stream experiences significant loss. The tall dark-yellow spikes illustrate the latency for the frames transmitted while the gaps in between these spikes represent the lost packets. The yellow line visible closer to the *x-axis* represents the 56,6Kb/s stream which did not experience as much loss as the 9,6Kb/s stream.

Both the Foreman CIF and Hall CIF videos exhibit similar traits. However, it is interesting to note that the Foreman CIF video presents higher latency peaks of up to 8 *ms* as shown in Figure 4.2. The limited representation of these videos (particularly the Foreman CIF video) indicates lost packets for those streams (9,6Kb/s, 28,8Kb/s, 56Kb/s and 64Kb/s). The Akiyo video illustrated 16 significant peak spikes as opposed to the Foreman CIF video's two peak spikes of 8 *ms* and the Hall CIF video's three peak spikes of 8 *ms* as illustrated in Figure 4.3. This shows that the Akiyo CIF video experienced more latency because the other videos experiencing significant frame loss, even though both the Foreman CIF and Hall CIF produced higher latency peak spikes.

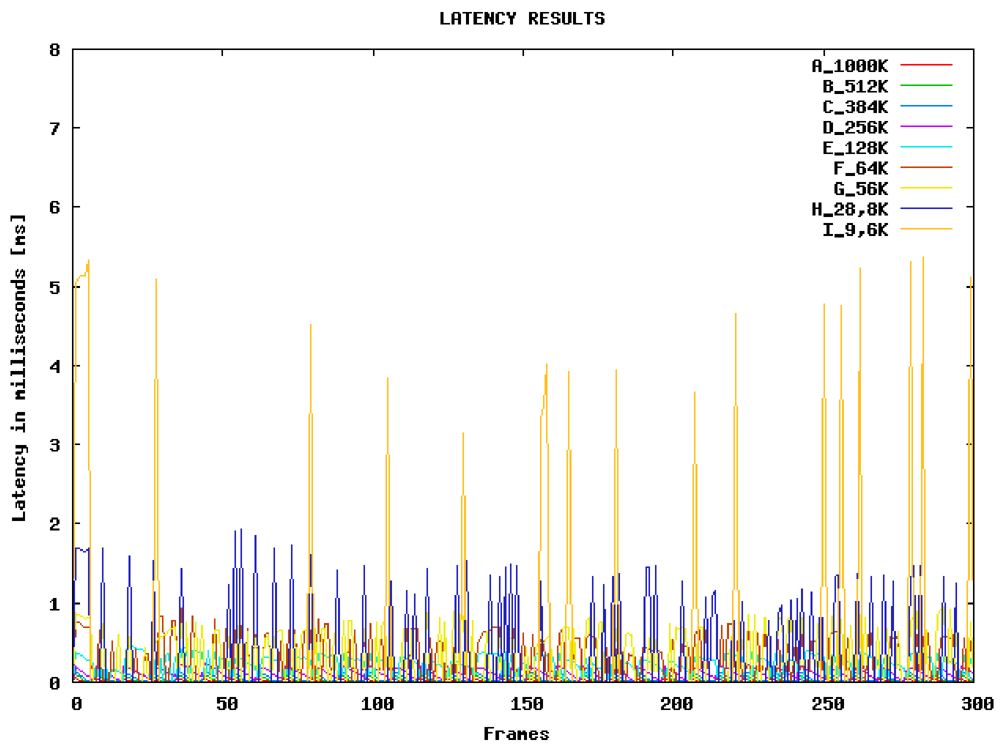


Figure 4.1: Latency for Akiyo CIF Video

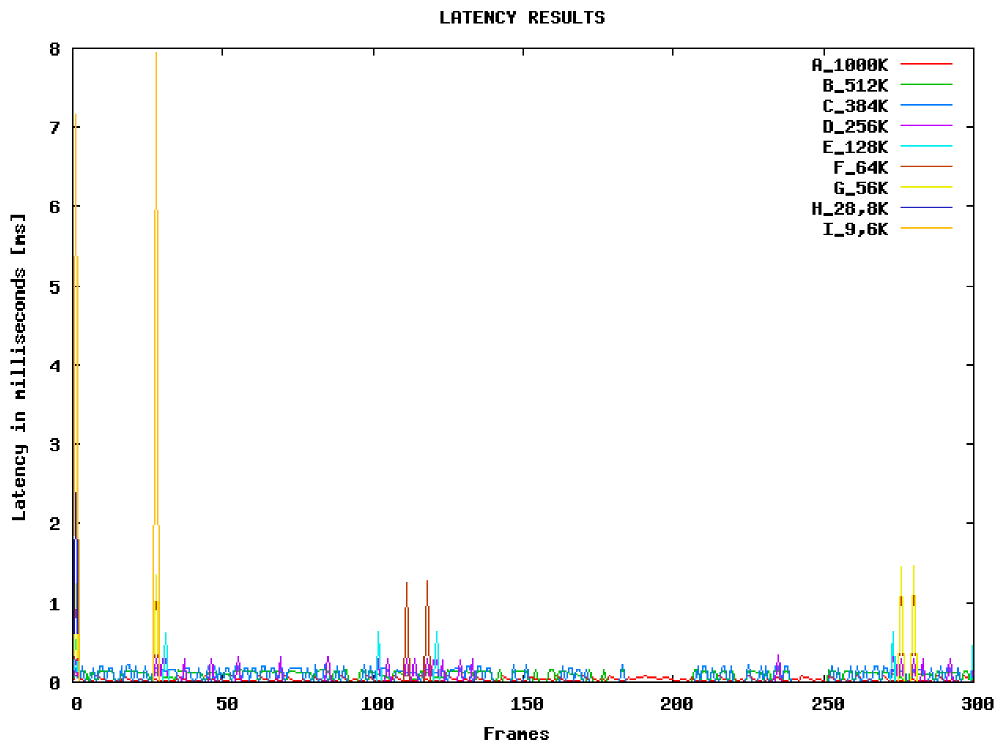


Figure 4.2: Latency for Foreman CIF Video

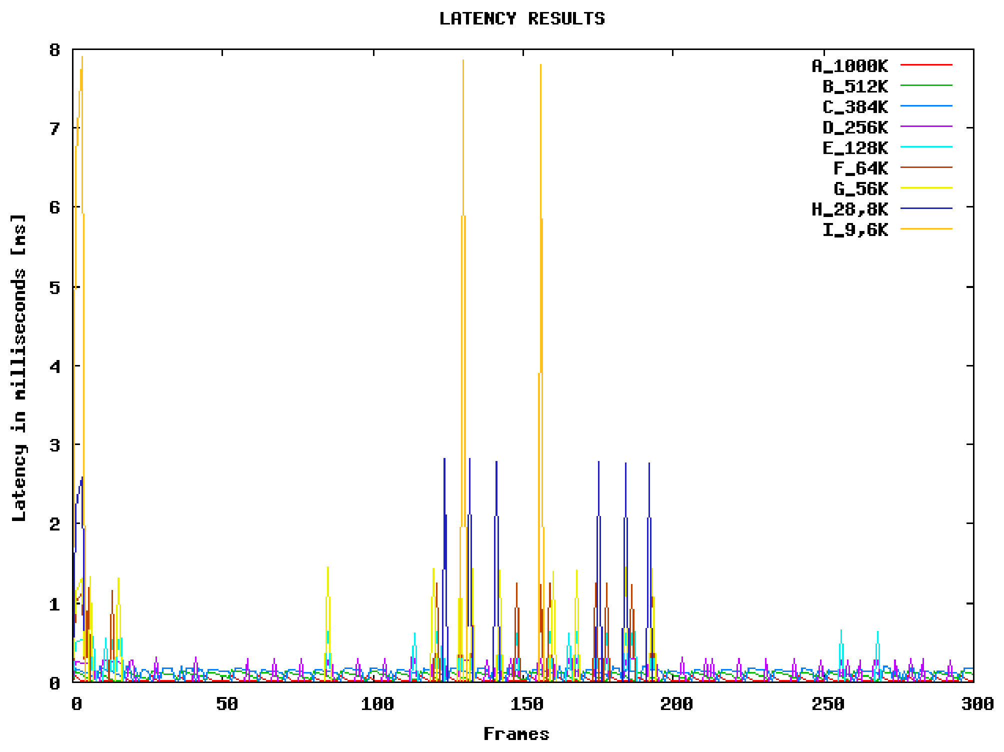


Figure 4.3: Latency for Hall CIF Video

4.3.2 Frame Loss

Figures 4.4, 4.5 and 4.6 are graphs which illustrate the lost video frames. These graphs compare the percentage frame loss for the three video frame types used to represent compressed transmitted MPEG-4 video (I-Frames, P-Frames and B-Frames). They also illustrate the average frame loss for all the frames. The *y-axis* shows the frame loss as a percentage whilst the *x-axis* shows the bandwidths (Kb/s) in descending order of bandwidth size.

From Figure 4.4, the high bandwidth streams present zero frame loss percentages (1000Kb/s down to 384Kb/s streams). The lower end of the medium bandwidth streams (the 128Kb/s and 64Kb/s streams) begin to display frame loss whilst the low bandwidth streams (56Kb/s and lower) experience higher losses.

The Foreman CIF video presents a different illustration as shown Figure 4.5. The majority of the low and medium bandwidths experience frame losses of up to 95% whilst only the

1000Kb/s stream is able to maintain a low frame loss average, close to 5 percent. This is most likely attributed to both the camera position panning and the object movement within the scene.

The Hall CIF video (Figure 4.6) presented a greater degree of frame loss in comparison to the Akiyo video, but not as much frame loss as the Foreman CIF video. The Hall CIF experienced no frame loss with the 1000Kb/s stream. However it did experience slight frame loss with 512Kb/s and a fair frame loss percentage (or neutral from the MOS achieved by this test) with the 384Kb/s stream.

The results demonstrate that the motion of objects (Hall CIF video) and the camera position (Foreman CIF Video) within the scene play a significant part in the quality of the video stream particularly for low bandwidths, as we would expect. With regards to the type of frames lost, in all cases where there are lost frames, the I-Frame always experiences significant loss whilst the P-Frame almost always experiences the least amount of loss when compared to the B and I Frames. This is likely because these the I-Frames are equivalent to JPEG images [89] and are significantly larger than the others.

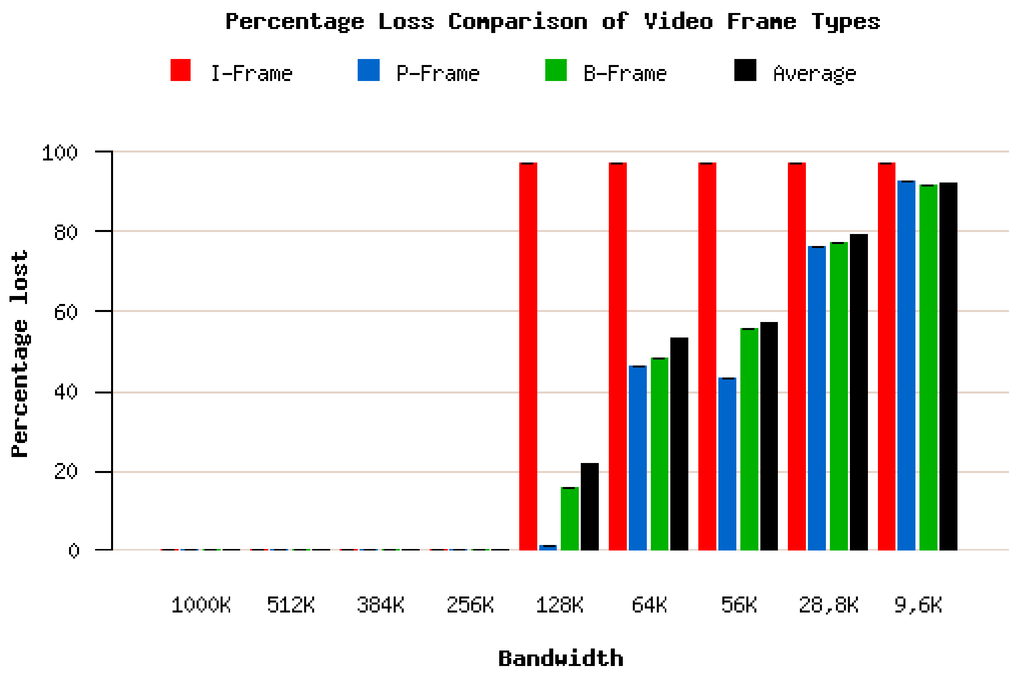


Figure 4.4: Video Frame Percentage Loss for Akiyo CIF Video

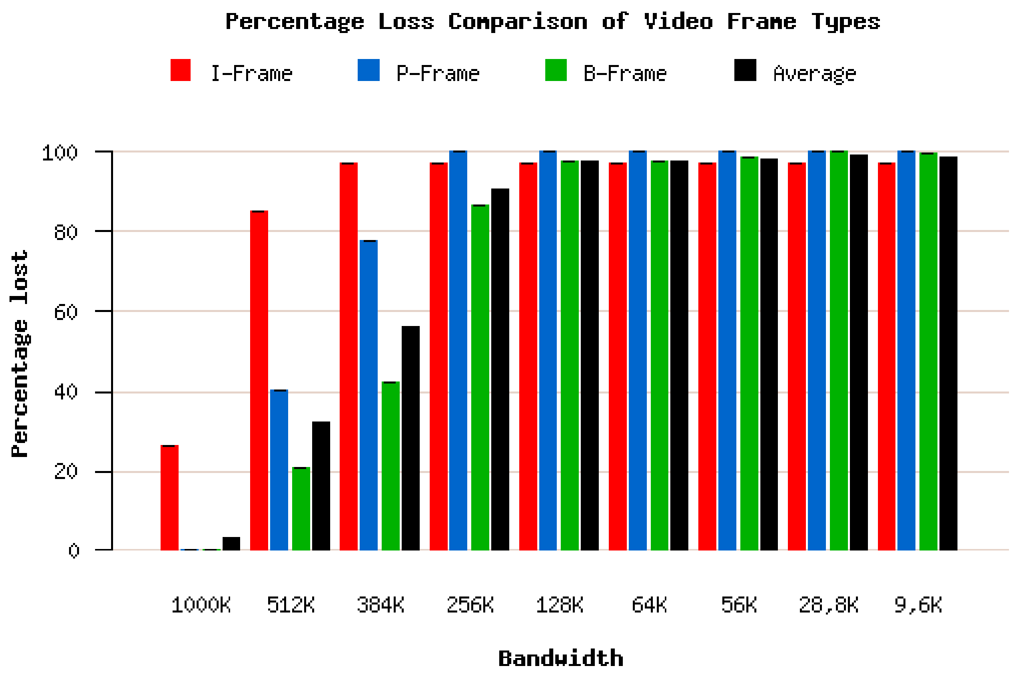


Figure 4.5: Video Frame Percentage Loss for Foreman CIF Video

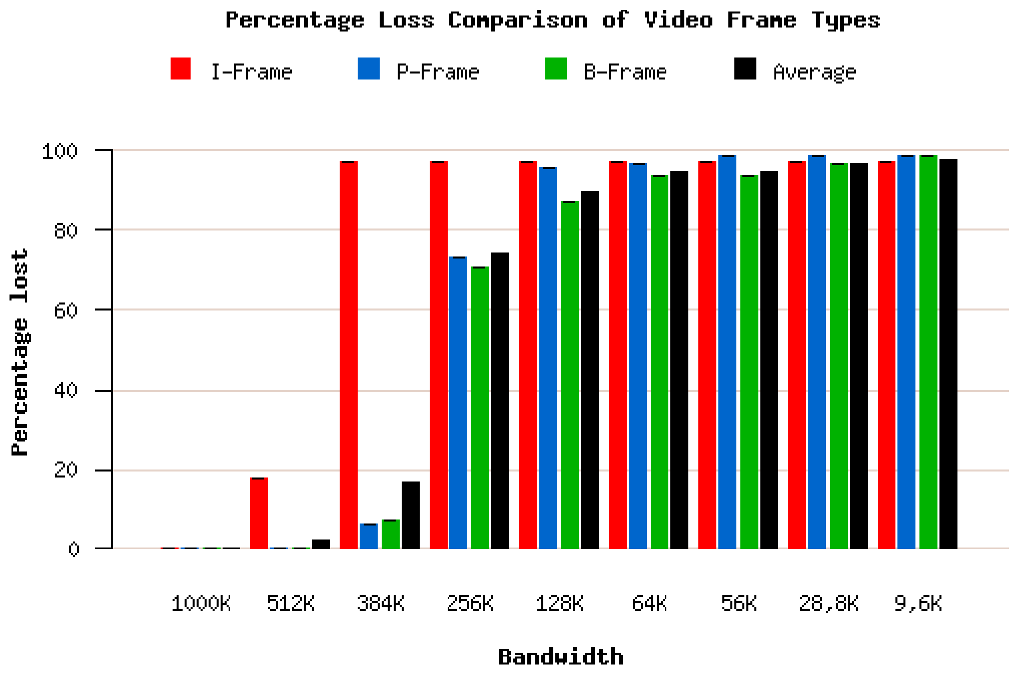


Figure 4.6: Video Frame Percentage Loss for Hall CIF Video

4.3.3 Jitter

Figures 4.7, 4.8 and 4.9 display the jitter (or the variation of the delay) results for the three CIF videos transmitted through the simulated network. The graphs express the jitter in milliseconds experienced as a result of the network transmission through these different bandwidths. The more erratic the curve (and not necessarily the height of the curve), the more likely packets will arrive at different times leading to poorer video quality. However, the jitter graphs are best compared with the equivalent video latency graphs to understand the video stream transmission.

These graphs are illustrated in a similar manner to the latency graphs, consisting of nine different bandwidth tests represented by the different coloured lines. The bandwidth lettering for the lines is to distinguish the order in which these bandwidth tests were performed (ordered from the largest bandwidth test A to the smallest bandwidth test I). The *y-axis* represents the jitter in milliseconds *ms* and provides a range from -2.0 *ms* to 8.0 *ms* for all three videos to simplify the comparison between them. The *x-axis* represents the video frames. The higher the absolute change in jitter measurement, the greater the variation in delay between frames.

The jitter in the Akiyo CIF video as shown in Figure 4.7, shows a jitter curve for the 9,6Kb/s stream. However, the long downward staggers for the dark-yellow 9,6Kb/s stream (the high saw-tooth pattern) represents a collection of lost frames in this instance with the upward jumps highlighting the next found frame. Those streams close to the *x-axis* illustrate minimal variation which corresponds to the high bandwidth streams (1000Kb/s down to 256Kb/s).

Both the Foreman CIF video (Figure 4.8) and the Hall CIF video (Figure 4.9) showed the 9.6Kb/s video stream with a jitter peak of 7.8 *ms* with a final jitter value of -1. *ms* for the Foreman CIF video and the 3.1 *ms* for the Hall CIF video. Another noticeable difference was that the majority of the bandwidth streams revealed curves crossing the *x-axis* (especially the Foreman CIF video).

This illustrates that as the stream is transmitted, the variance in the delay is such that the frames are getting lost or that the frames are being received faster than they are being dispatched.

Only the 1000Kb/s bandwidth stream was able to maintain an average very close to the *x-axis* for both videos, while the other high bandwidth steams (512Kb/s and 384Kb/s) were relatively close to the *x-axis*.

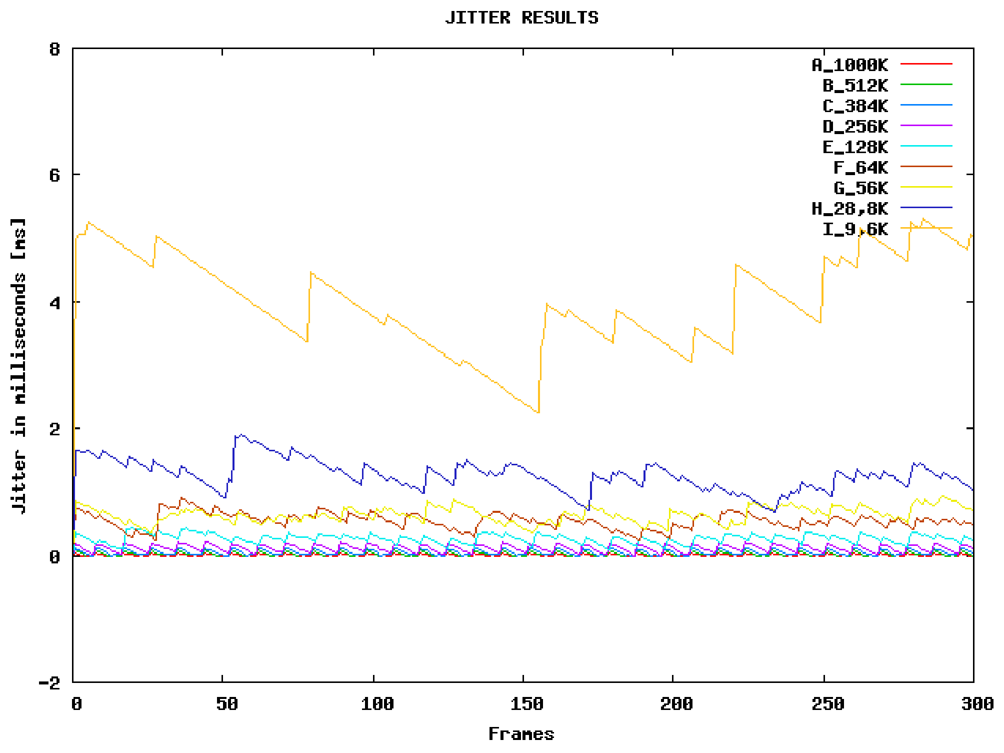


Figure 4.7: Jitter for Akiyo CIF Video

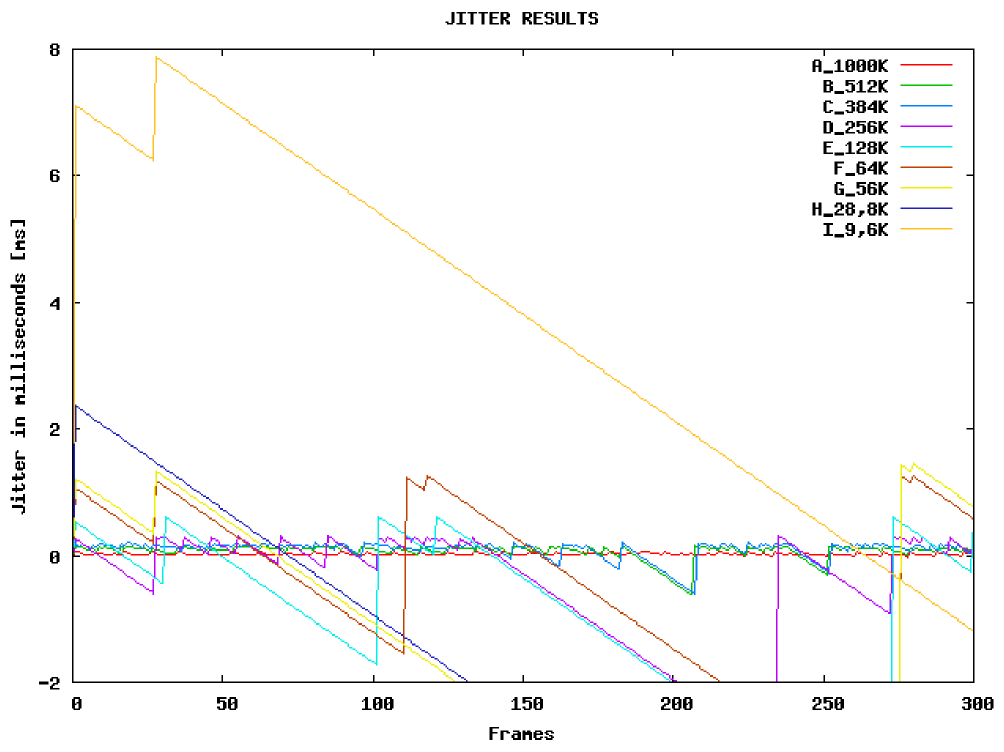


Figure 4.8: Jitter for Foreman CIF Video

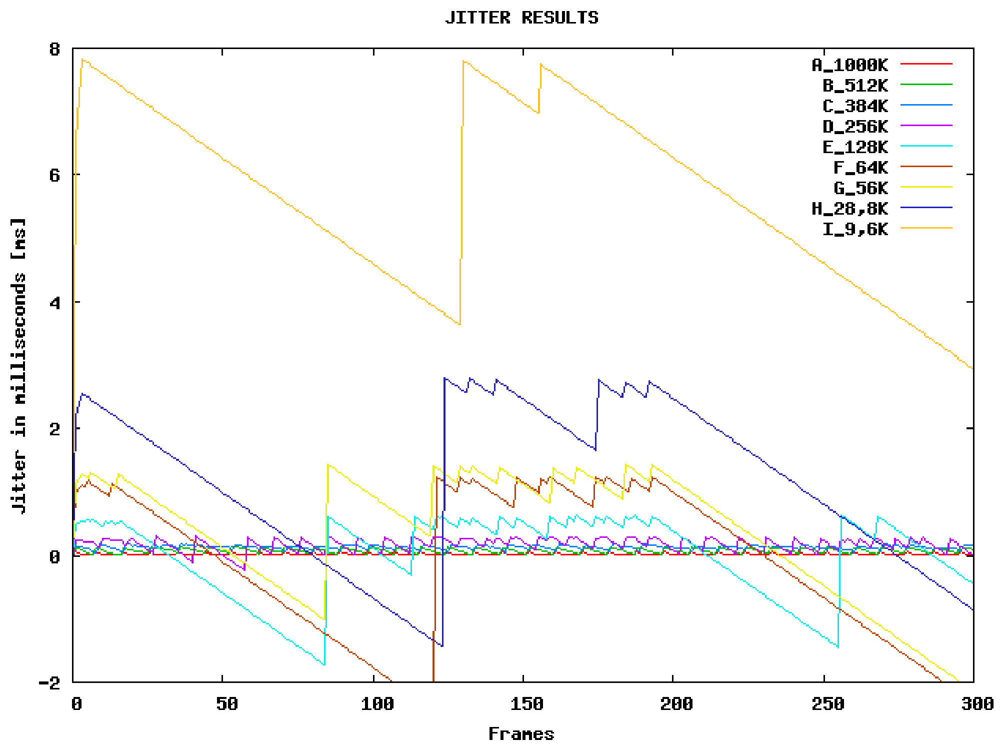


Figure 4.9: Jitter for Hall CIF Video

4.3.4 PSNR vs Reference Video

Figures 4.10, 4.11 and 4.12 illustrate the signal noise experienced within the network transmission. Each graph presents the signal to noise ratio in decibels (dB) for all the 300 frames transmitted. However, these figures also contain a reference line which represents a low noise network transmission (this is represented by the red + line towards the top of the figure (just below the 40 dB margin). This reference video represents the data from a physical network trace over a high bandwidth connection (100 Mb/s).

The lettering colour scheme applies to these graphs in a similar manner to the latency and jitter tests. However, with the introduction of the reference line, the colour scheme shifts accordingly with the highest bandwidth test being assigned the light green line, and the lowest bandwidth test the dark green line. The *y-axis* represents the peak signal to noise ratio value in decibels (dB) and provides a range from 0 dB to 50 dB for all three videos to simplify the comparison between them. The *x-axis* represents the video frames. For these graphs, the higher the value, the less noise is experienced during the network transmission.

In the Akiyo CIF video Figure 4.10, the following bandwidths lie along the reference line: 1000Kb/s down to 256Kb/s. The remainder are situated towards the middle of the graph. Figure 4.11 shows the 1000Kb/s line along the reference line, with the 512Kb/s and 384Kb/s lines averaging below the 20 dB margin. However, what is not visible from Figure 4.11, was that the bandwidths 256Kb/s, down to 28,8Kb/s were displayed along the 9,6Kb/s line (even though only the 9,6Kb/s is displayed). Figure 4.12 illustrated the 1000Kb/s line along the reference line, with the 512Kb/s line averaging close to the 26 dB margin and the 384Kb/s line closer to the 20 dB margin.

The significant drop of the PSNR for the Foreman CIF 1000Kb/s streams (as well as the reference stream) is a result of the 26.5% frame loss for the I-Frame. Even though the average frame loss for this video was only 3.0% (for the 1000Kb/s stream), this yielded a significant increase in the noise of this video.

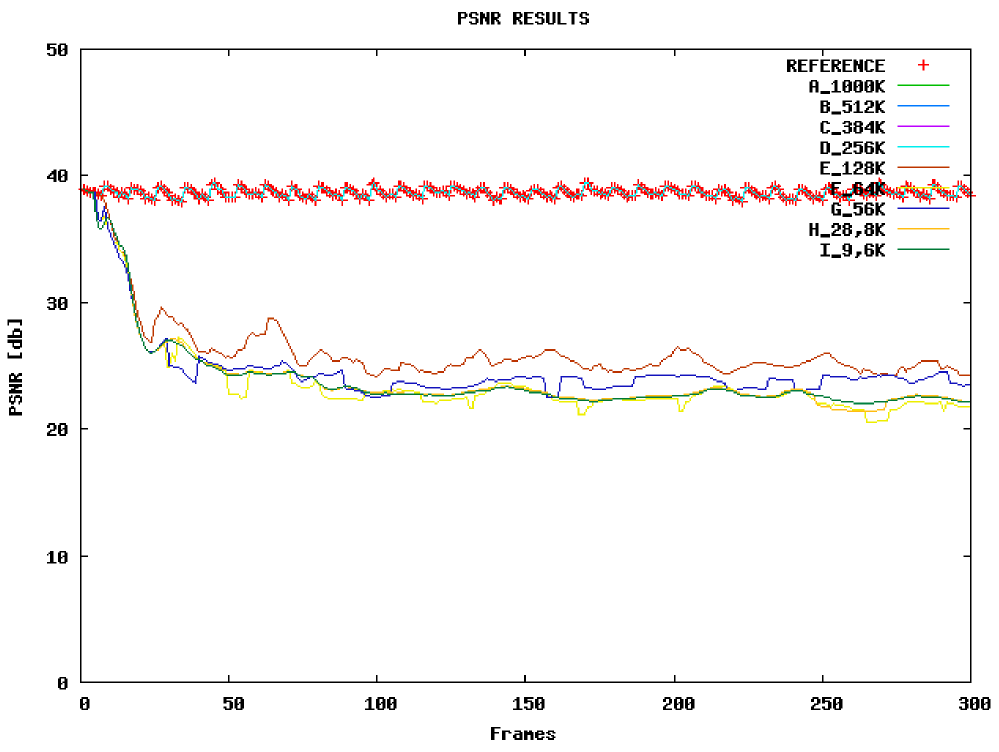


Figure 4.10: PSNR for Akiyo CIF Video

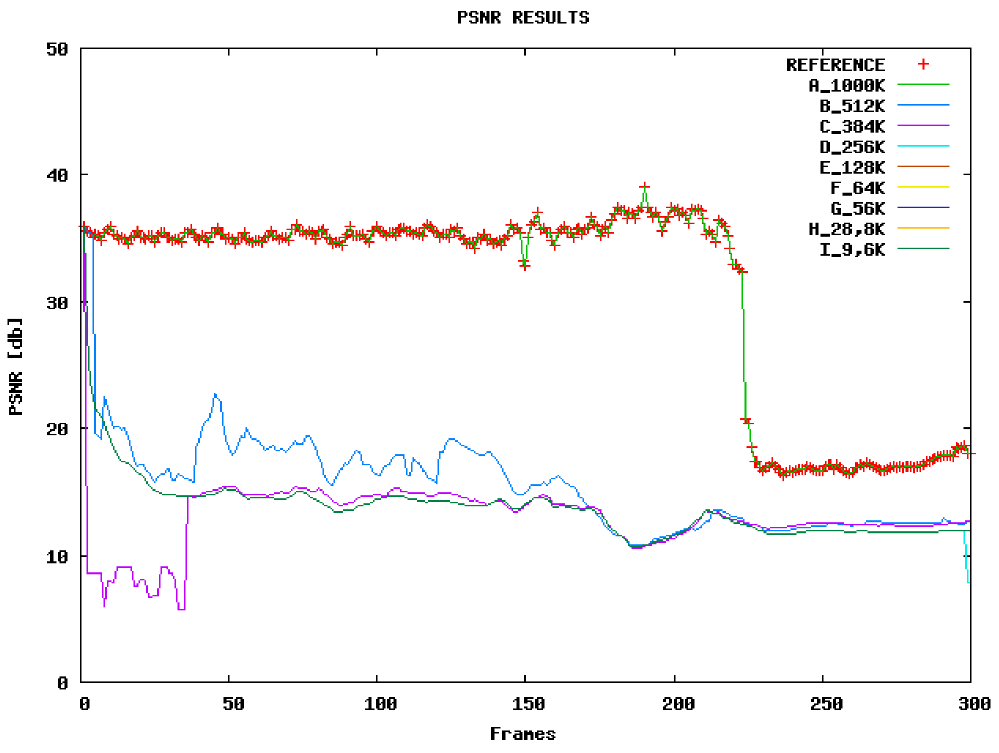


Figure 4.11: PSNR for Foreman CIF Video

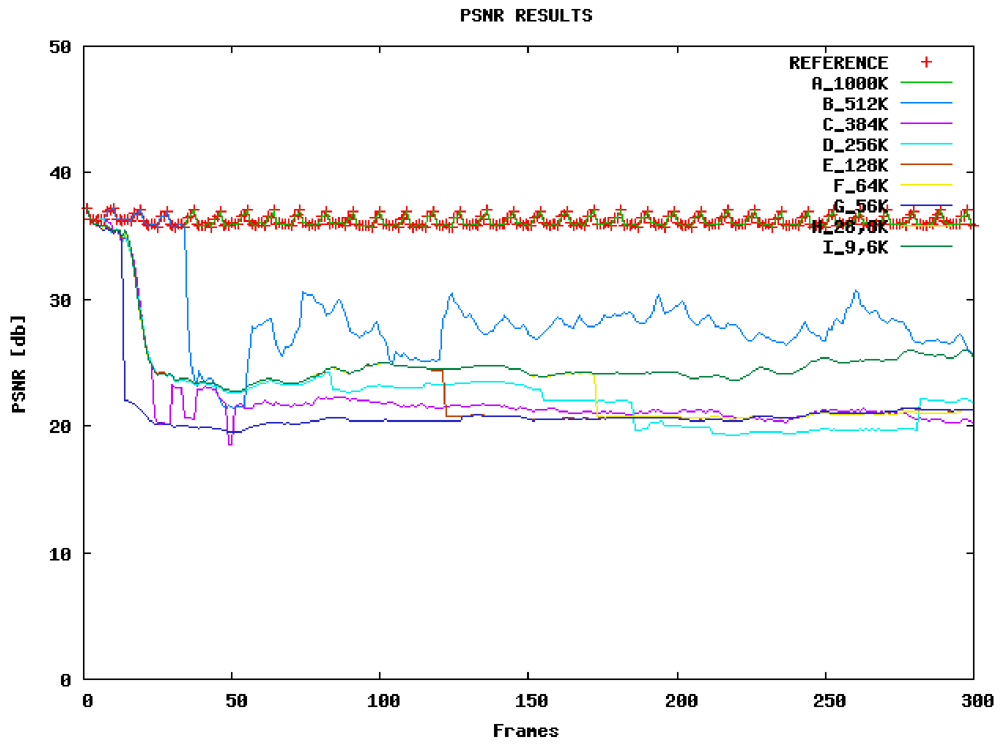


Figure 4.12: PSNR for Hall CIF Video

4.3.5 Average PSNR Values with Standard Deviation

The bar graphs in Figures 4.13, 4.14 and 4.15 illustrate the average PSNR through the different bandwidth tests (shown in red). Each test also provides the standard deviation (the RMS deviation from the average) from the average value attained represented in blue. The higher the PSNR value, the less noise is experienced through the network transmission. The *y-axis* shows the PSNR in decibels (dB) whilst the *x-axis* shows the different bandwidths (Kb/s) ordered from the largest to the smallest.

The average PSNR values show the low bandwidth results being lower than the high bandwidth representations, as expected. The standard deviation is low for the high bandwidth values, but significantly higher for the bandwidths with middle PSNR averages (15 - 33 dB). The results for the averages and standard deviations for the medium bandwidth tests lie midway between the high and low bandwidth tests. The Akiyo CIF video (Figure 4.13) provided higher average values and lower standard deviations than the Foreman CIF video (Figure 4.14). The Hall CIF video (Figure 4.15) results lay closer to the Akiyo CIF video results with respect to both the average signal noise and the standard deviation.

The Foreman CIF video produced poor results across the board except for the 1000Kb/s bandwidth stream reaching 30 *dB.* This illustrates that the Foreman CIF video experienced greater noise across the majority of the bandwidth streams due to the changes in the camera position and movements of the objects within the scene. However, even though the 1000Kb/s bandwidth produced the largest standard deviation amongst all the bandwidth streams, it experienced the least amount of noise.

This does not necessarily indicate that lower standard deviations for streams produce better quality outputs. The lower standard deviations for all the other bandwidth streams for the Foreman CIF video indicate that the majority of the noise experienced was close to the average. Nevertheless, this video still produced poor video transmissions for these bandwidth streams.

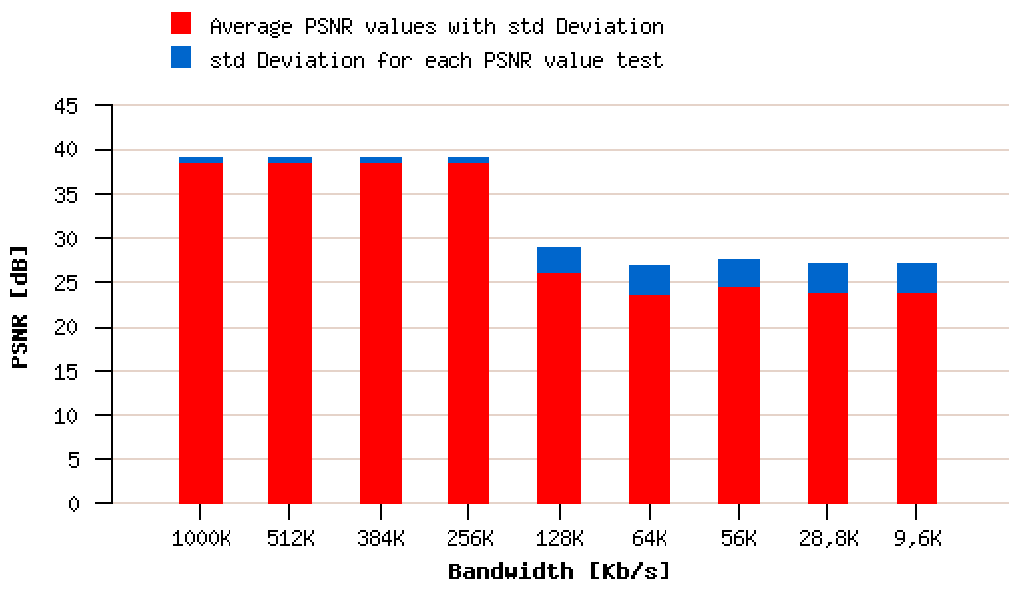


Figure 4.13: PSNR Values & STD Deviation for Akiyo CIF Video

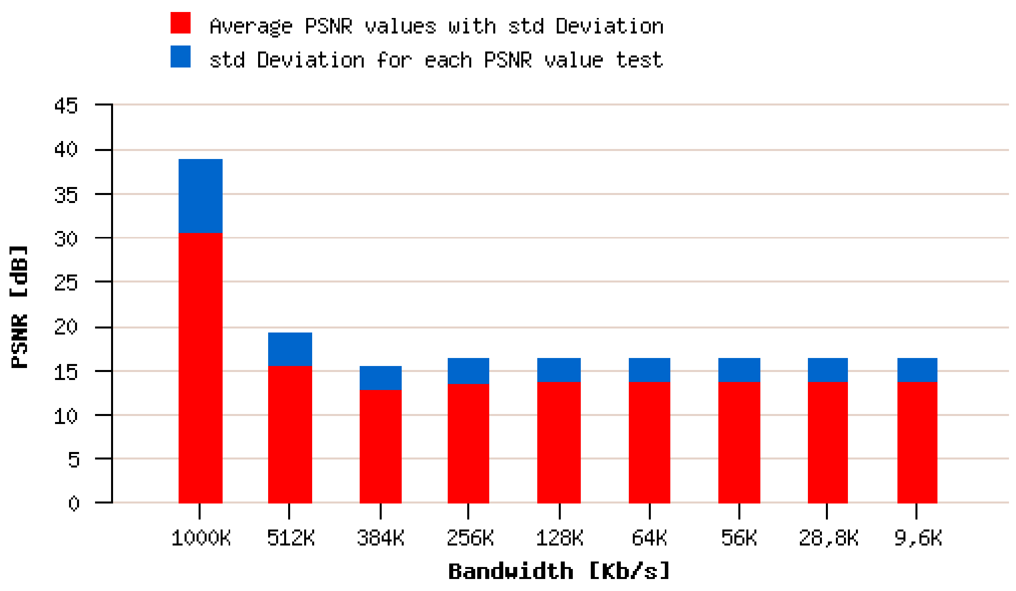


Figure 4.14: PSNR Values & STD Deviation for Foreman CIF Video

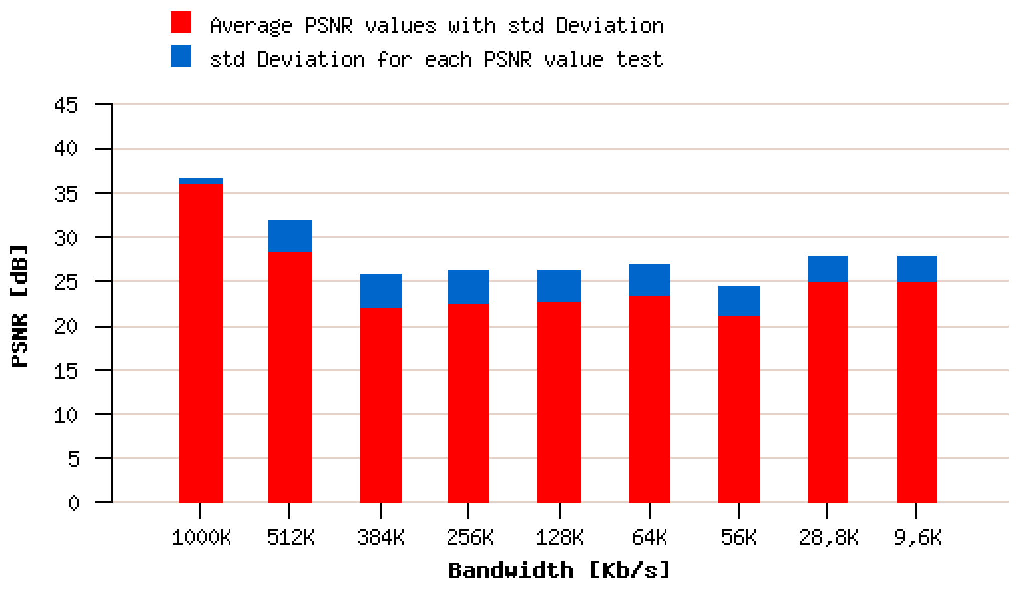


Figure 4.15: PSNR Values & STD Deviation for Hall CIF Video

4.3.6 MOS

Figures 4.16, 4.17 and 4.18 represent the Mean Opinion Scores for the different bandwidth network transmissions. These graphs illustrate the ratings (1-5 with 5 representing the highest quality and 1 representing the lowest quality) against the different bandwidths. The graphs consist of the nine bandwidth tests performed in descending order of size as represented along the *x-axis.* The *y-axis* presents the MOS ratings from 1 to 5.

The MOS results for the Akiyo CIF video as illustrated in Figure 4.16 were high for the high bandwidth tests (1000, 512, 384 Kb/s) and low for the low bandwidth tests (56, 28,8 and 9,6 Kb/s). However, of the the medium bandwidth tests, only the 128Kb/s test achieved an average score of 3 with the 64Kb/s test scoring below average and the 256Kb/s test scoring above average.

The MOS results for the Foreman CIF video as illustrated in Figure 4.17 produced low scores for all the bandwidth tests except the 1000Kb/s test which achieved an average score. The scores for the Hall CIF video illustrated in Figure 4.18, showed the 1000Kb/s test achieving a score of 4, with the following bandwidth tests achieving average scores of 3: 512Kb/s, 28,8Kb/s and 9,6Kbs. However, the remainder of the tests achieved the lowest scores out of all the bandwidth tests (384Kb/s down to 56 Kb/s).

An interesting observation presented for the Hall CIF video was the difference in scores between the middle and low bandwidth streams. However, these MOS results are linked to the PSNR average results and the lower bandwidth averages only differ from the low scoring bandwidths tests (384Kb/s down to 56 Kb/s) by 2.5 *dB.* This difference also falls along the margin between the MOS score of 2 and 3. Another attribute is that the variation of the jitter for this video in Figure 4.9 illustrates both the 9,6Kb/s and 28,8Kb/s streams averaging above the *x-axis.* These bandwidth tests (384Kb/s down to 56 Kb/s) presented a greater portion of jitter below the *x-axis*, which may account for the 2.5 *dB* difference in these outputs.

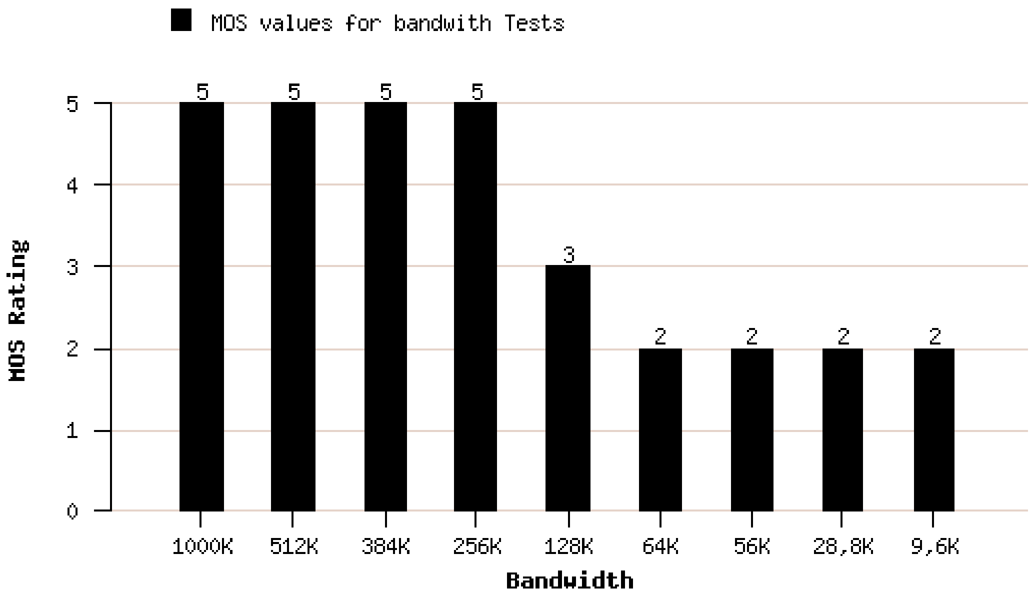


Figure 4.16: MOS for Akiyo CIF Video

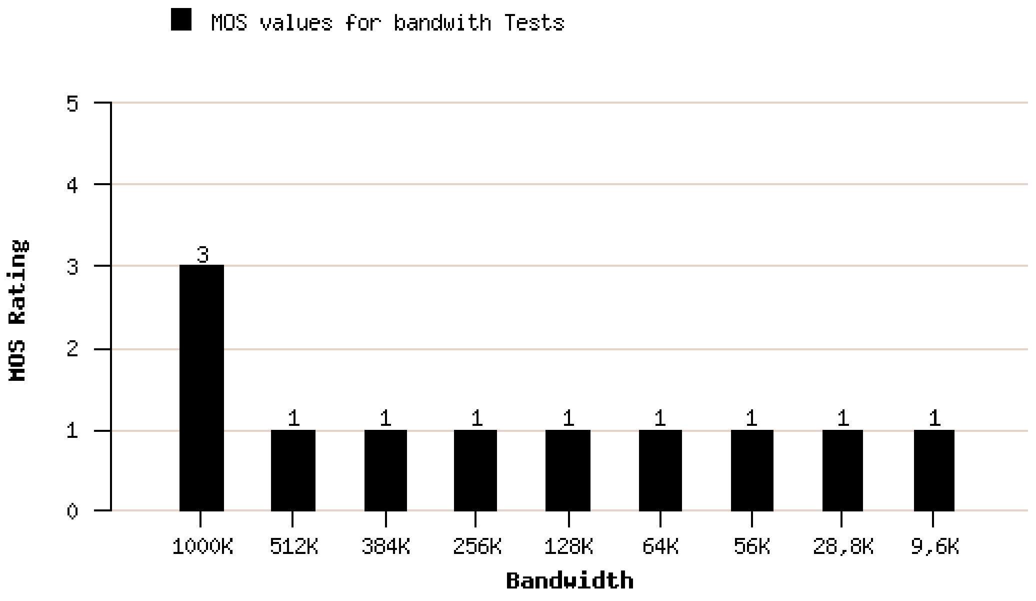


Figure 4.17: MOS for Foreman CIF Video

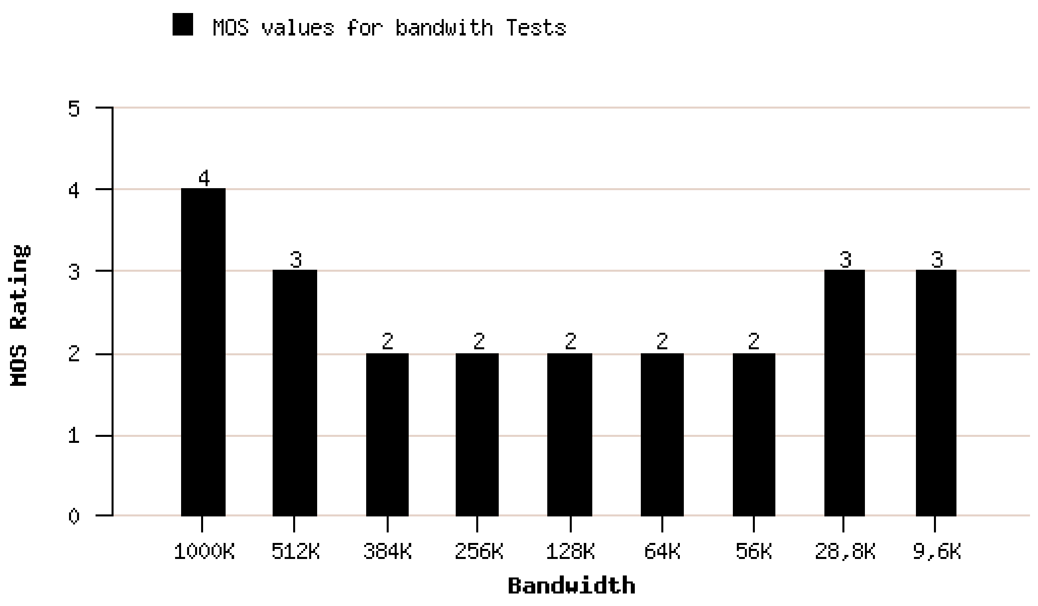


Figure 4.18: MOS for Hall CIF Video

4.3.7 Video Quality

Figures 4.19, 4.21 and 4.20 display the final output of the three videos after they have been transmitted through the network. For the previous graphs presented within this section (the PSNR, PSNR with standard deviations and the MOS graphs), the data was collected on videos after they had been repaired with the FV tool (Fix Video tool is designed to return the videos with lost frames to the original total of 300 frames in order to compare the differences between videos on a frame by frame basis). Figures 4.19, 4.21 and 4.20 represent the video outputs before the frames have been adjusted to the original 300 frames. The arrangement of the videos are shown in Table 4.1.

1000K	512K	384K
256K	128K	64K
56K	28.8K	9.6K

Table 4.1: Video Bandwidth Arrangement

Figure 4.19 illustrates the video outputs of the Akiyo CIF Video for the different bandwidth streams. This illustrates that the following bandwidth streams seem to provide high quality outputs: 1000Kb/s down to 256Kb/s. Some noticeable smudges are present over the *news-reporter's* face within the following bandwidth streams: 128Kb/s down to 28.8Kb/s. Even though it may seem that the 9.6Kb/s video has a better frame output, it is evident by the different *face impression*, that the frames are not in sync with the high bandwidth streams. This demonstrates that the video is experiencing frame loss, delay and jitter.

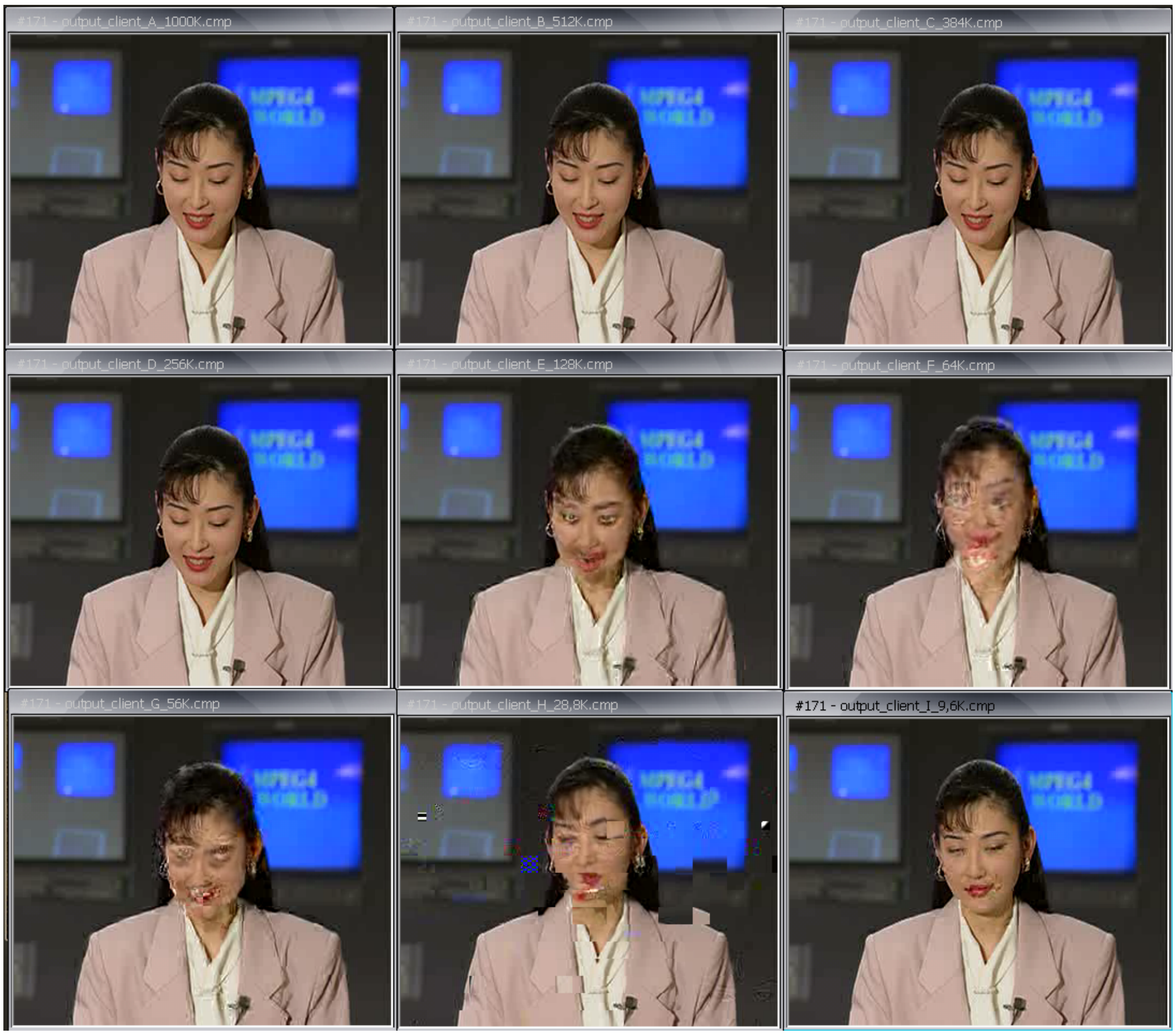

#171 - output_client_A_1000K.cmp
#171 - output_client_B_512K.cmp
#171 - output_client_C_384K.cmp
#171 - output_client_D_256K.cmp
#171 - output_client_E_128K.cmp
#171 - output_client_F_64K.cmp
#171 - output_client_G_56K.cmp
#171 - output_client_H_28,8K.cmp
#171 - output_client_I_9,6K.cmp

Figure 4.19: Bandwidth comparison for Akiyo CIF Video

The Hall CIF video produced slightly different outcomes as illustrated in Figure 4.20, with both the 1000Kb/s and 512Kb/s videos showing high quality outputs. However, the videos 384Kb/s down to 56Kb/s all seem to have the two *office personnel* objects missing. There is also significant noise present in their respective locations within the picture. The remaining 28.8Kb/s and 9.6Kb/s videos provide less noise but no *office personnel* are present. The object movement within this hall scene demonstrates that there is a decrease in the quality of video output for this video in comparison to the Akiyo CIF video.

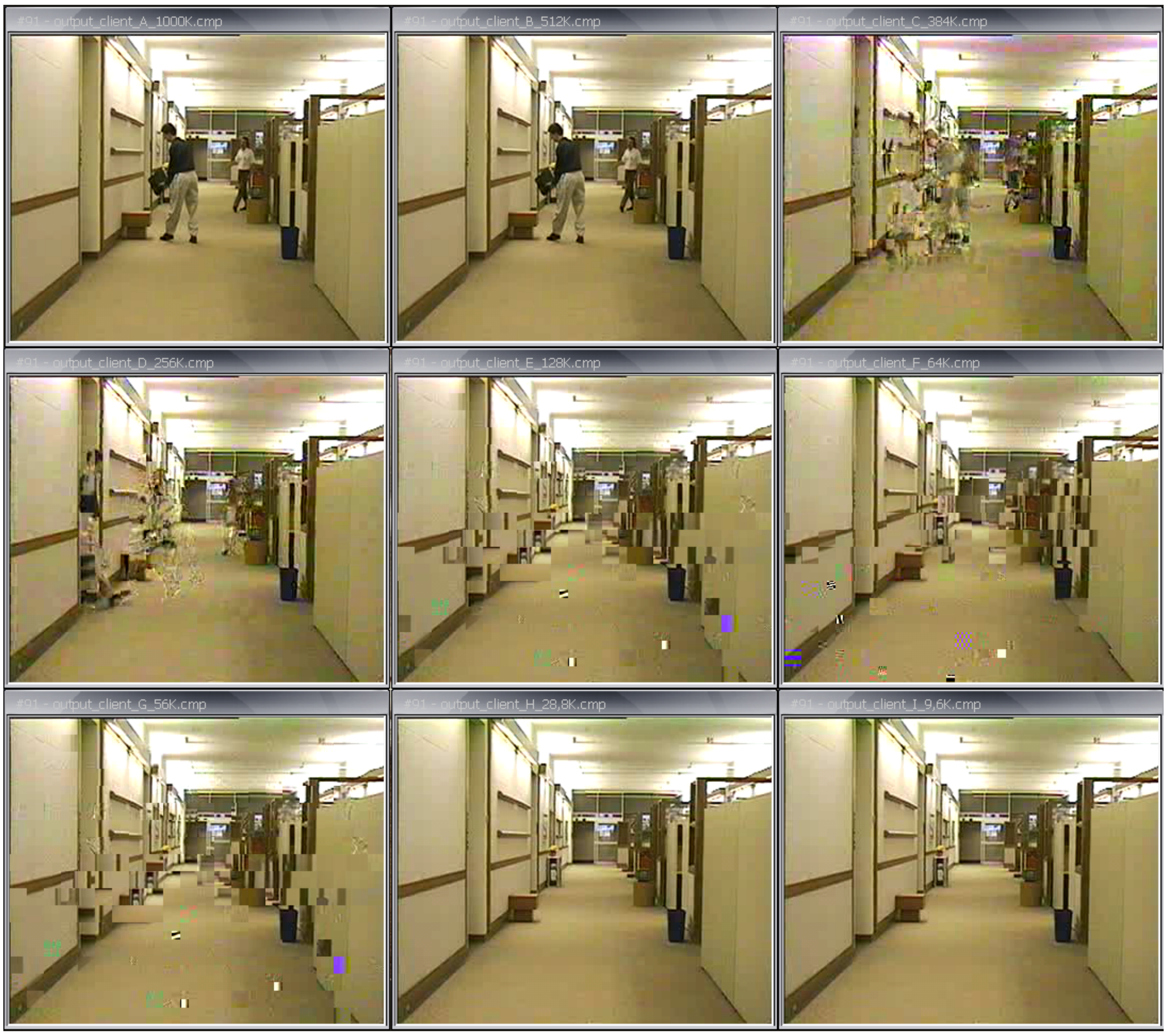

#91 - output_client_A_1000K.cmp
#91 - output_client_B_512K.cmp
#91 - output_client_C_384K.cmp
#91 - output_client_D_256K.cmp
#91 - output_client_E_128K.cmp
#91 - output_client_F_64K.cmp
#91 - output_client_G_56K.cmp
#91 - output_client_H_28,8K.cmp
#91 - output_client_I_9,6K.cmp

Figure 4.20: Bandwidth comparison for Hall CIF Video

The Foreman CIF video illustrated in Figure 4.21, shows the 1000Kb/s stream with correct output (as this video lay along the reference line for the PSNR reference graph in Figure 4.11). Even though the 512Kb/s and 28.8Kb/s streams do provide better picture qualities than the rest (excluding the 1000Kb/s), the pictures they produce are not in sync with the 1000Kb/s video. The 384Kb/s video produces significant noise whilst the remaining bandwidth streams produce excessive noise noted by the green pixelated images. The changes in camera position and object movements within the scene definitely have an effect on the quality of video stream produced.

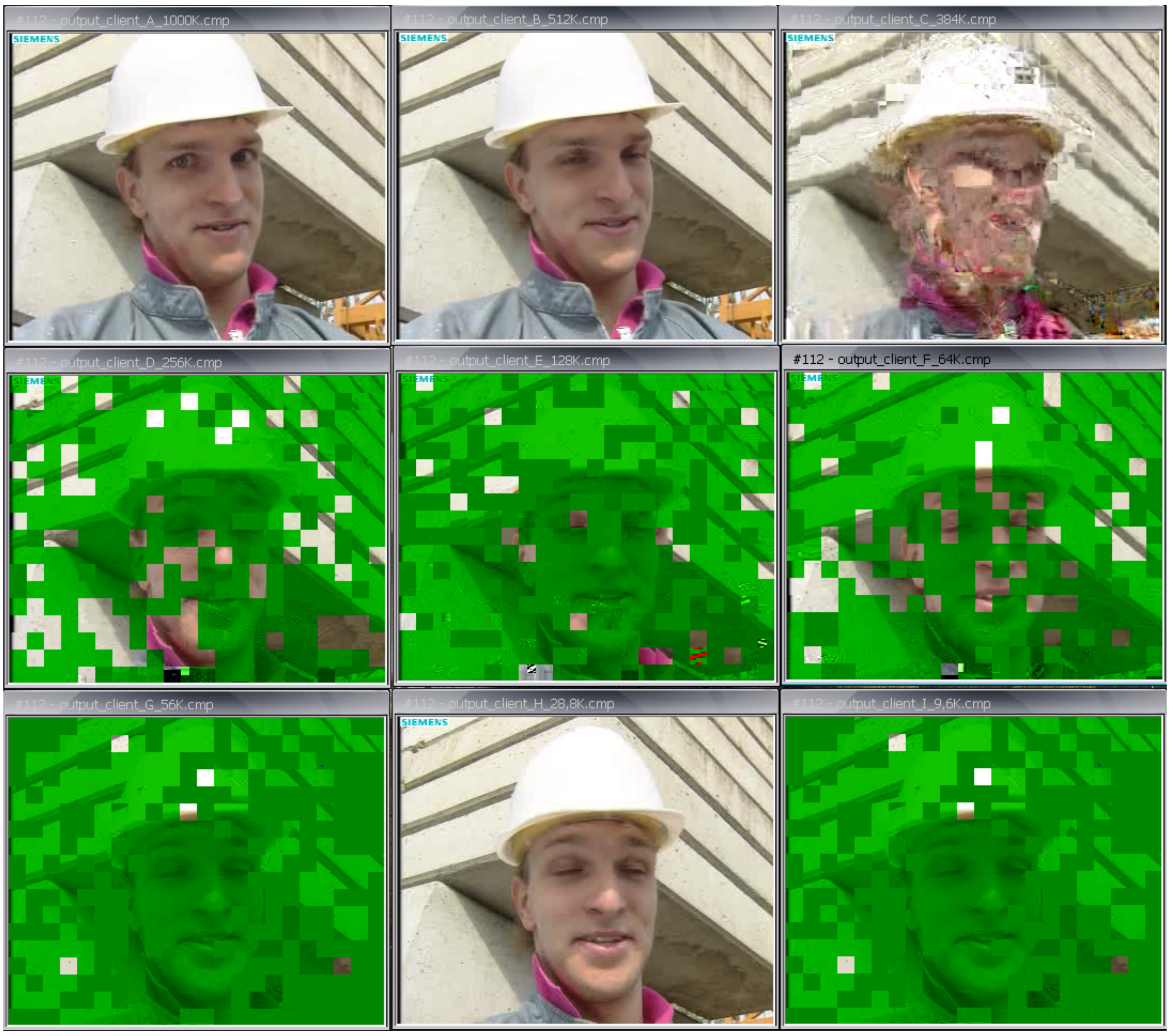

#112 - output_client_A_1000K.cmp
#112 - output_client_B_512K.cmp
#112 - output_client_C_384K.cmp
#112 - output_client_D_256K.cmp
#112 - output_client_E_128K.cmp
#112 - output_client_F_64K.cmp
#112 - output_client_G_56K.cmp
#112 - output_client_H_28,8K.cmp
#112 - output_client_I_9,6K.cmp

Figure 4.21: Bandwidth comparison for Foreman CIF Video

Subjectively, these snapshots suggest that the Akiyo CIF video produced the better quality set of videos over the Foreman CIF videos. The Hall CIF video displayed comparable quality values to those of the Akiyo CIF video but with significant loss of the objects within the scene.

4.3.8 Cache Size

As illustrated in the video quality Figures 4.22, 4.23 and 4.24, the available bandwidth plays a significant part in the quality of the video stream. However, when the bandwidth is insufficient, packets (or frames) get lost or dropped as the network connection can not sustain the network transmission which decreases the quality. In order to correct this quality loss, caching involves providing a buffer to compensate for the lack of available bandwidth and temporarily stores the portion of the video before playing it. Figures 4.22, 4.23 and 4.24 all illustrate the minimum cache size required to produce a video with a MOS rating of 5 (Exceptional quality) for the respective bandwidths.

These values are obtained by comparing the MOS rating of the bandwidth test against the MOS rating of the Reference video (which is always 5) and if the values do not match, the cache size is incremented (by a fixed size of 10 *KB*) and the simulation retested and the MOS ratings re-compared. The *y-axis* represents the minimum cache size in kilobytes (*KB*) with a range of 0 to 1000 *KB* for all three videos to illustrate the comparison between them. The *x-axis* represents the different bandwidths tested.

The 9.6Kb/s bandwidth stream for all videos required a significant cache size of to maintain the same high video quality. However using this cache size with this connection is effectively the same as downloading the full video, which is not streaming. In the case of the Foreman CIF video which required a cache size 986.88 *KB* for this connection, this download would take approximately 800 seconds *((960 * 1000 * 8) / (9.6 * 1000))* or 13 minutes at 100% of the possible bandwidth (which is very difficult to achieve). Also to note that this video clip is only 10 seconds long.

As expected, the required cache size is significantly higher for the low bandwidth tests in relation to the high bandwidth tests throughout all three videos. The Foreman CIF video (Figure 4.23) requires higher cache sizes than the Hall CIF video (Figure 4.24) for all the bandwidth tests. The Hall CIF video requires higher cache sizes than the Akiyo CIF video (Figure 4.22) for all the bandwidth tests, except for the 1000Kb/s bandwidth test where no cache is needed.

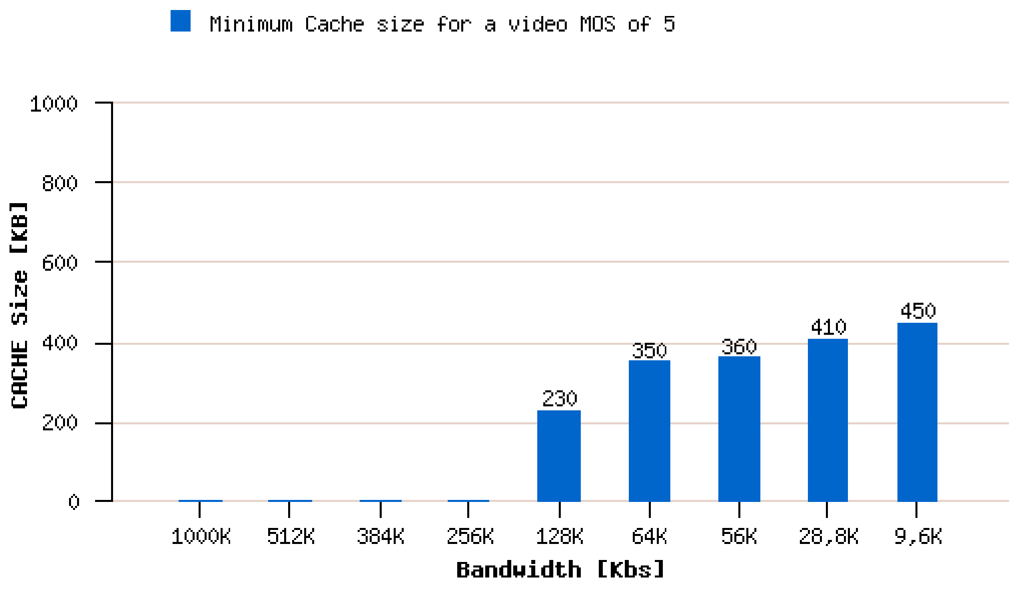


Figure 4.22: Minimum cache sizes for Akiyo CIF Video

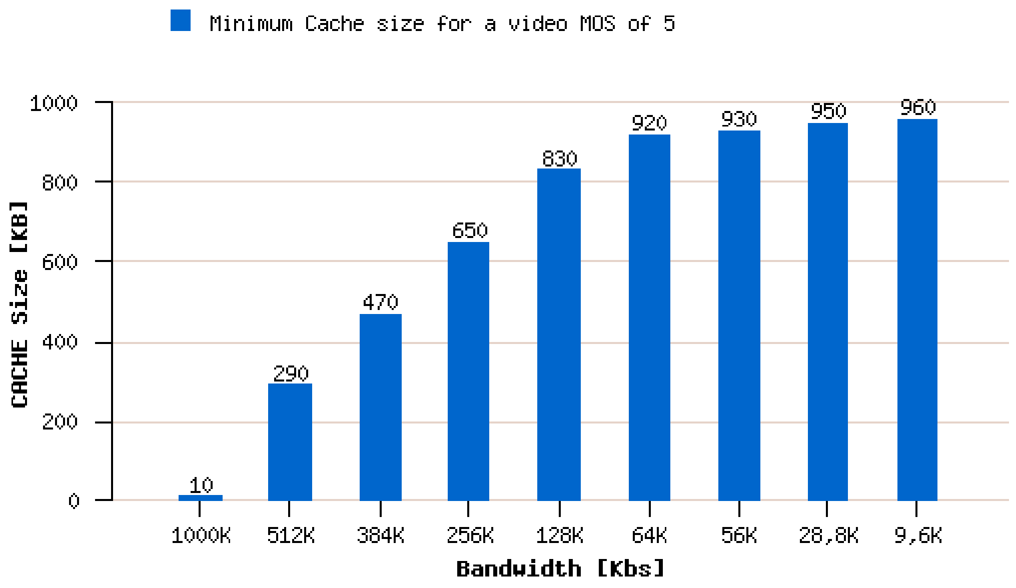


Figure 4.23: Minimum cache sizes for Foreman CIF Video

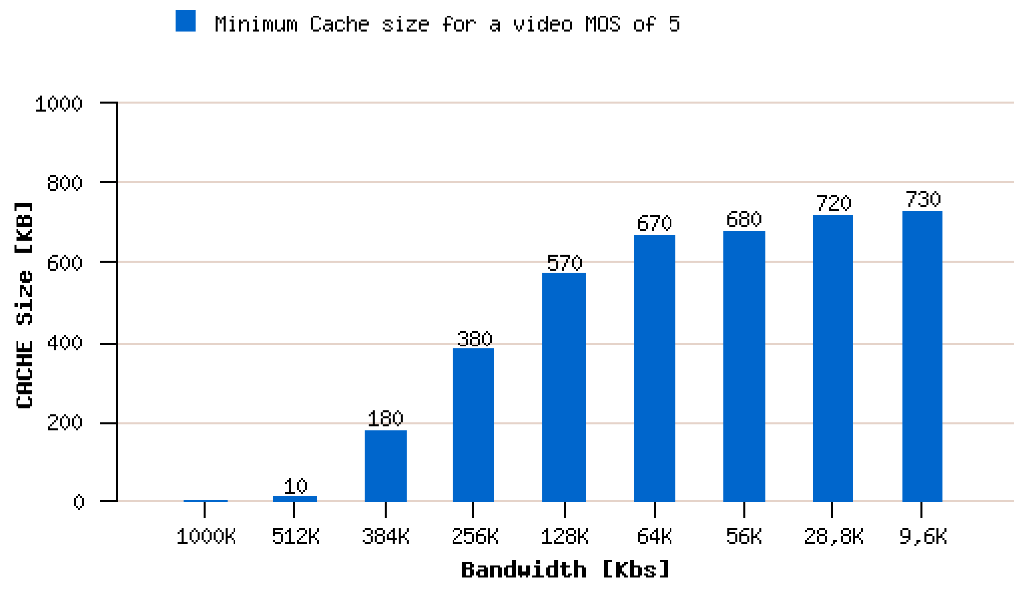


Figure 4.24: Minimum cache sizes for Hall CIF Video

4.4 Small Network Results

In subsections 4.4.1, 4.4.2 and 4.4.3 the aim of the tests is to evaluate CAFSS-Net through a physical network. Each of the following subsections provides results for tests using almost the same configuration and tools (except where stated otherwise) presented in the Peer-to-Peer tests in the previous section. However, instead of transmitting the stream through NS-2, the stream was transmitted through a physical network.

The tracefile data was collected on both the source and receiving devices through the use of a UNIX based network traffic capturing tool called TCPDUMP [112]. The data was collected from the nodes 0 and 1 (server and client_A respectively) with respect to Figure 3.9. The results of the physical tests are presented in the next subsections 4.4.1, 4.4.2 and 4.4.3.

The physical network comprised of two machines running Windows XP, each with a 100Mb/s Ethernet interface card connected to each other through a CAT5 Ethernet cable.

The distance between the two machines was approximately 10 meters which may lead to some latency. To obtain the different bandwidth tests, a software-based bandwidth limiter (or network choking device) was implemented on the client node (TRAFFIC SHAPER XP [113]).

Even though these physical network test comparisons may be referring to the Peer-to-Peer scenario simulation results instead of the Small Network simulation results, the Small Network simulation results presented insignificant differences from the Peer-to-Peer simulation results. This was as a result of the direct connection (in Figure 3.9) between the server and client_A (the two nodes used for the physical network results).

This representation is illustrated in Figure 4.25 (which is an extract from Figure 3.9) with the blue arrow representing the similar Peer-to-Peer simulation representation with node 0 (server) and node 1 (client_A) as the data collection points.

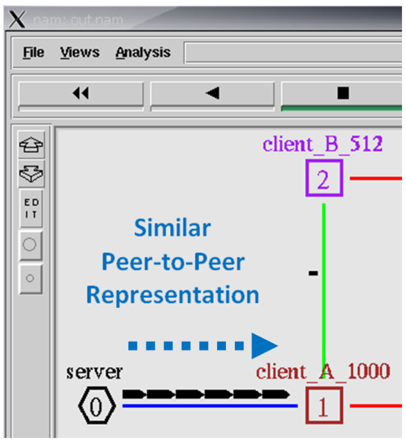


Figure 4.25: Peer-to-Peer Scenario simulation similarity to Small Network Scenario

4.4.1 Frame Loss for Physical Network

Figures 4.26, 4.27 and 4.28 represent the percentage frame loss for the physical network implementation. The *x-axis* represents the six bandwidth tests (as the lower three bandwidth tests were not performed for these three subsections: 4.4.1, 4.4.2 and 4.4.3). The *y-axis* represents the percentage of lost frames with the colour scheme for the percentages remaining the same as the previous simulated frame loss tests in section 4.3.2.

From Figure 4.26, the 56Kb/s bandwidth stream produced the greatest frame loss amongst all the bandwidth streams. However, unlike the frame loss results from the Akyo CIF test performed on the simulated network in the Peer-to-Peer scenario (Figure 4.4), this 56Kb/s test score only experienced an average loss of 12%, whilst the same bandwidth in the Peer-to-Peer test experienced average frame loss of up to 58%. This difference could be attributed to an advantage of non-simulated networks (the accuracy of physical networks as opposed to abstraction for simulated networks).

The Hall CIF video test results shown in Figure 4.28, illustrated results very similar to those of the Foreman CIF video results (Figure 4.27) even though they were both significantly different to Peer-to-Peer simulation results. Though these physical test percentages may be different from the simulated results, the ratio of loss between the three videos highlights that the Hall CIF video performed similarly to the Foreman CIF video which was also evident with the Peer-to-Peer simulation.

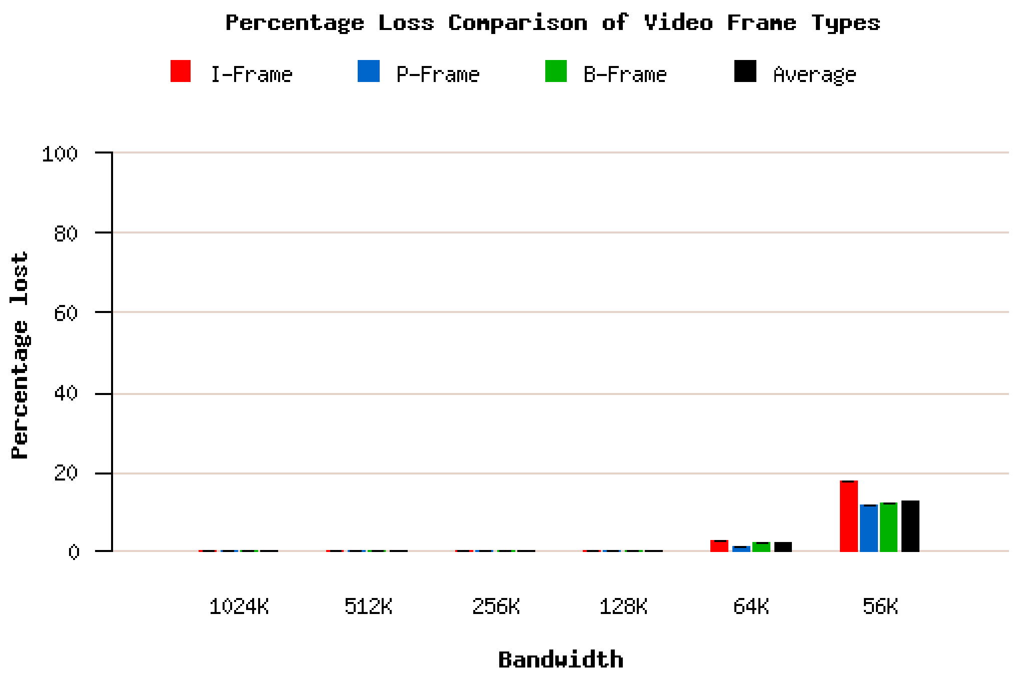


Figure 4.26: Frame Loss for Akiyo CIF Video on Physical Network

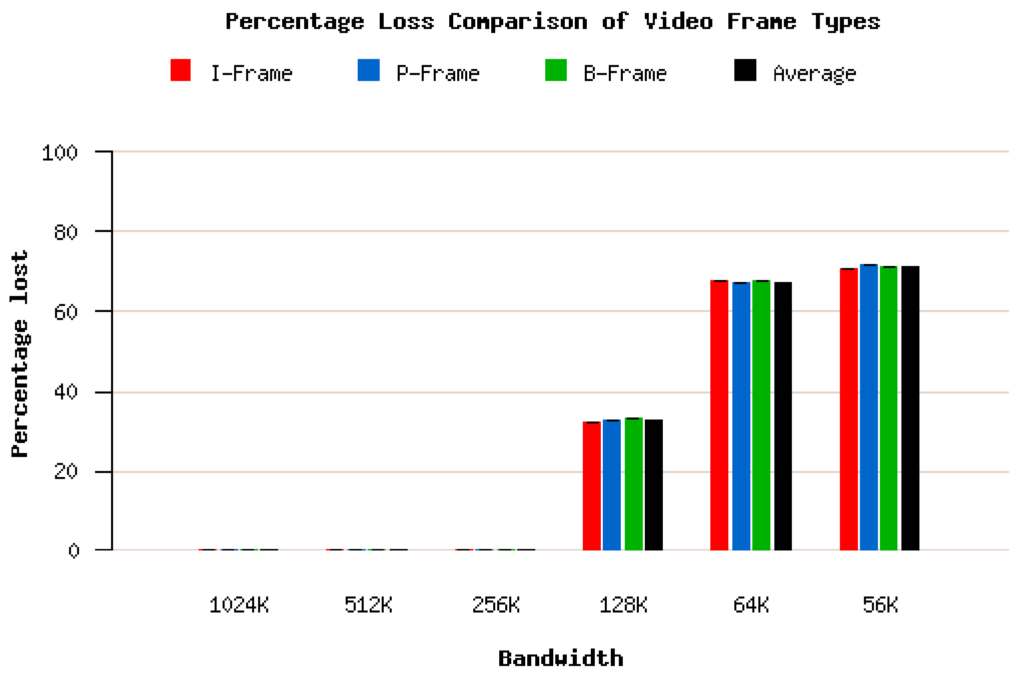


Figure 4.27: Frame Loss for Foreman CIF Video on Physical Network

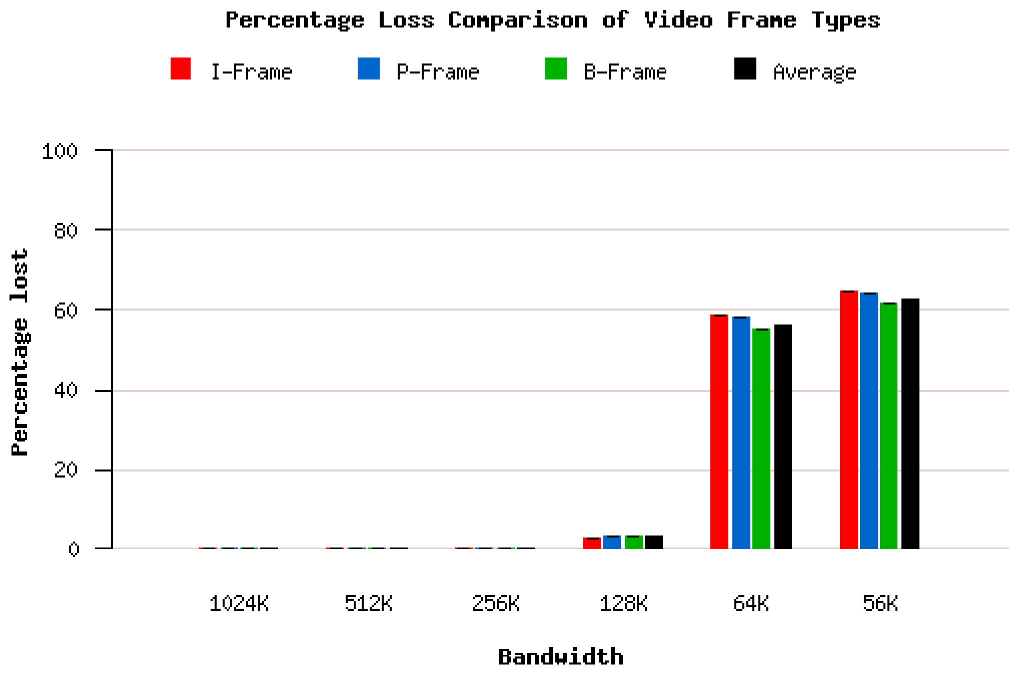


Figure 4.28: Frame Loss for Hall CIF Video on Physical Network

4.4.2 Latency for Physical Network

Figures 4.29, 4.30 and 4.31 represent the end to end delays (latency) of frames for the physical network implementation. The latency results illustrate the latencies for the full 300 frames transmitted though the network and are represented by the *x-axis*. The *y-axis* consists of the latency measured in milliseconds *(ms)* with a range of 0 to 35 *ms* for comparing these graphs. The bandwidth tests are illustrated using the same colour scheme as presented within the previous Peer-to-Peer latency tests in section 4.4.1,4.3. No latency tests were performed for the low bandwidth transmissions.

The Akiyo CIF video produced a set of latency graphs for the physical bandwidth tests as illustrated in Figure 4.29. From this figure, its clear that these latency graphs are indeed different from the latency graphs produced by the Peer-to-Peer simulated tests which are illustrated in Figure 4.1. This difference is a result of physical distance between the the server and client_A for the physical tests, which can account for the increased latency. Even though the peaks are higher, the continuous positive changes in the curves suggests that the delay remains close to constant, hence there is a small change in the latency from frame to frame.

This significant height difference relates to the end-to-end delay between frames. The server and Client A where significantly further apart from each other as opposed to the simulated results. For this video, only the 56Kb/s and 64Kb/s videos experienced frame loss, and this is illustrated with the brown and light blue streams “bouncing” up and down within Figure 4.29. This is similar to the peaks presented within the Peer-to-Peer simulations for the 9,6Kbs stream in Figure 4.1.

The Foreman CIF video (Figure 4.30) illustrated the lost frames by the sharp 90 degree drop for the 56, 64 and 128 Kb/s streams. The secondary spikes are the remaining frames that were not lost being transmitted. The Hall CIF video (Figure 4.31) presents a similar graph representation, except that more frames are transmitted after the first major loss for the 56 and 64 Kb/s streams (indicated by the continuous peaks for these streams).

The high bandwidth streams did remain close to the *x-axis* as opposed to the medium bandwidth streams, which indicates that the high bandwidth streams experienced less latency than their counterparts. This finding matches the simulated results and even though the end-to-end delays were significantly different, the similar peak representation (between lost frames) was present within both the simulated and physical network tests.

The Akiyo CIF video presented a lower latency representation than the Hall CIF video which is illustrated by comparing the highest latency times achieved within each graph. In turn, the Hall CIF video yielded a slightly lower latency representation than the Foreman CIF video.

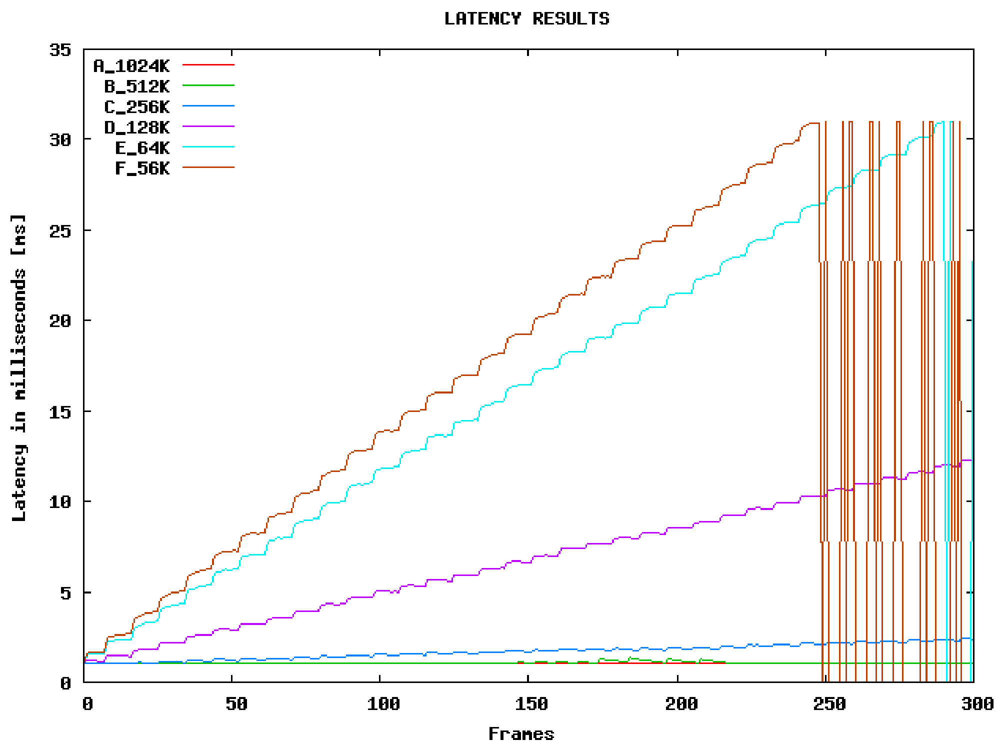


Figure 4.29: Latency for Akiyo CIF Video on Physical Network

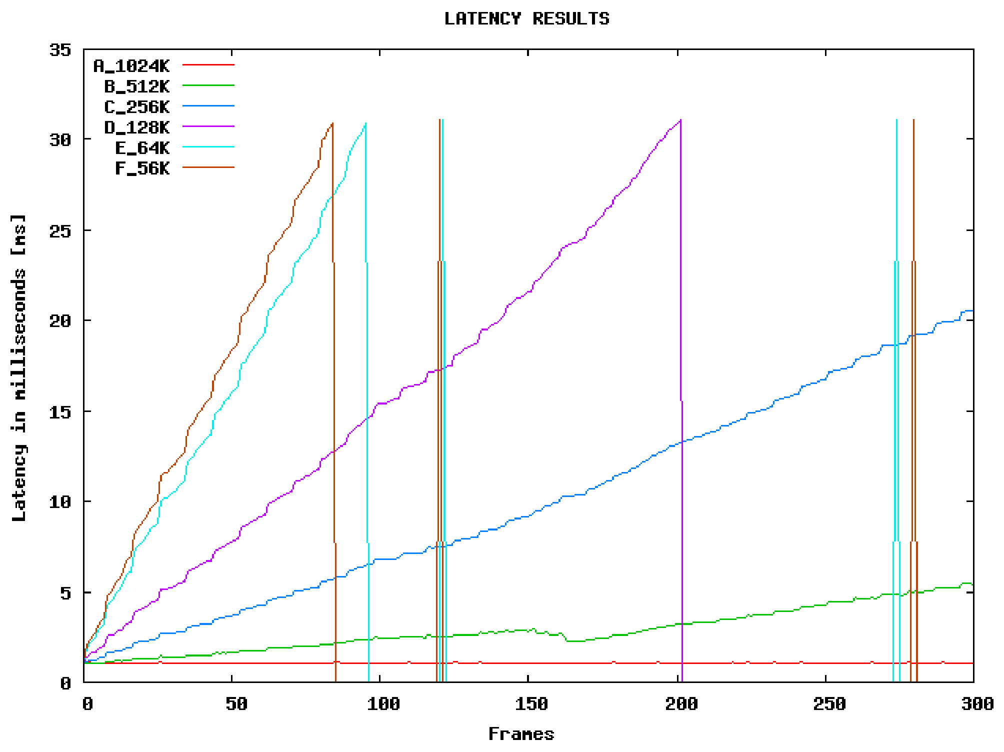


Figure 4.30: Latency for Foreman CIF Video on Physical Network

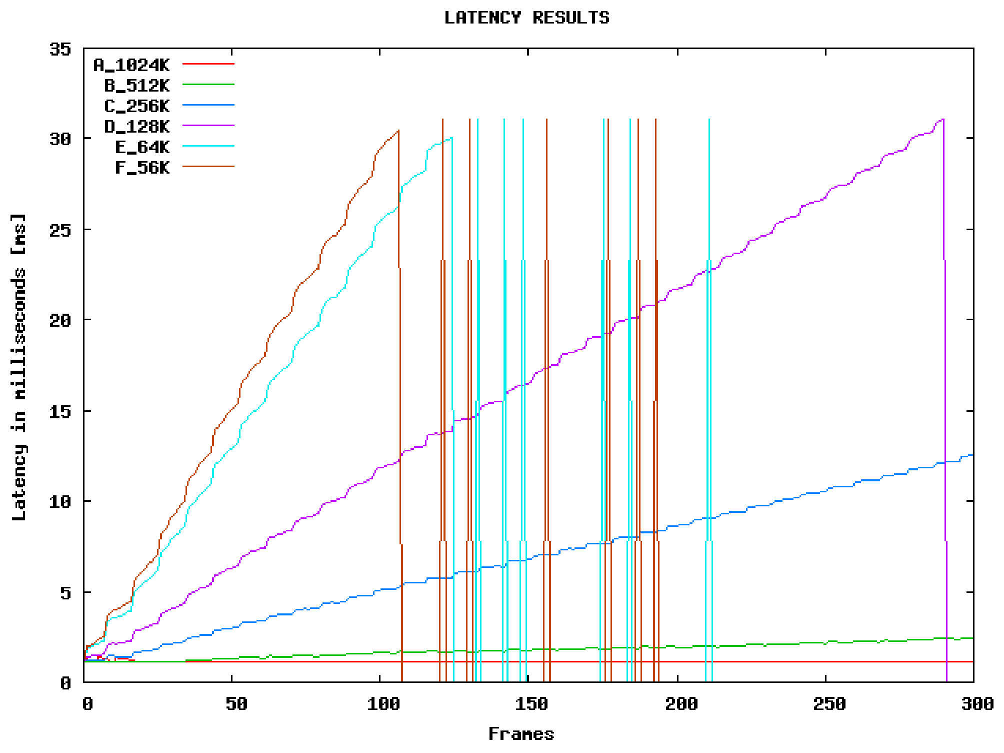


Figure 4.31: Latency for Hall CIF Video on Physical Network

4.4.3 Jitter for Physical Network

Figures 4.32, 4.33 and 4.34 represent the variation in the end to end delays (jitter) of frames for the physical network implementation. These graphs illustrate the full 300 video frames as the *x-axis* and the jitter measured in milliseconds *(ms)* for the *y-axis* with a range of 0 to 30 *ms*. The higher the absolute change in jitter, the greater the variation in the delay between frames.

The Akiyo CIF video jitter curves closely resemble the latency curves for the physical network test in Figure 4.29. This is a result of the significant increase in the latency for the physical network. The 56Kb/s and 64Kb/s streams for this video increase at a constant rate until the point where frames are lost. These lost frames are illustrated by the horizontal "flow" (close to the 30 *ms* mark). Even though lost frames are experienced, these two bandwidth streams do not provide negative jitter. However, the high bandwidth streams (particularly the 1024Kb/s and 512Kb/s) produced the same result as the Peer-to-Peer simulated tests in Figure 4.7.

The Foreman CIF video (Figure 4.33) results provide an interesting representation for the low bandwidth streams. Aside from the high latency graphs (as a result of the significant latency experienced from the physical network), a similar downward stagger (saw-tooth pattern) emerges as a result of the jitter producing negative results. Though the changes may not be as significant as the Peer-to-Peer simulations, the overall shape of the graph highlights the same negative jitter found in the Peer-to-Peer jitter simulations. Once again, the 1024Kb/s and 512Kb/s streams presented the similar results as the Peer-to-Peer simulations.

The Hall CIF video (Figure 4.34) produced the same saw-tooth shape as the Foreman CIF video, except that this saw-tooth pattern is significantly narrower than the Peer-to-Peer simulation. This similarity between the Foreman CIF video and the Hall CIF video was also present within the Peer-to-Peer simulation tests. However, both the Foreman CIF and Hall CIF video did not experience as much frame loss for this physical network tests as the Peer-to-Peer simulated tests did.

The results demonstrated that the jitter variance increased as the bandwidth decreased for all videos. The Foreman CIF video produced a wider saw-tooth pattern than the the Hall CIF video, while even though the Akiyo CIF video jitter peak representation did experience frame loss, it did not produce this saw-tooth pattern (which indicates negative jitter in this instance). These tests proved that the Foreman CIF video produced the most jitter as compared to both the Hall CIF video and the Akiyo CIF video. The Akiyo CIF video produced the least amount of jitter.

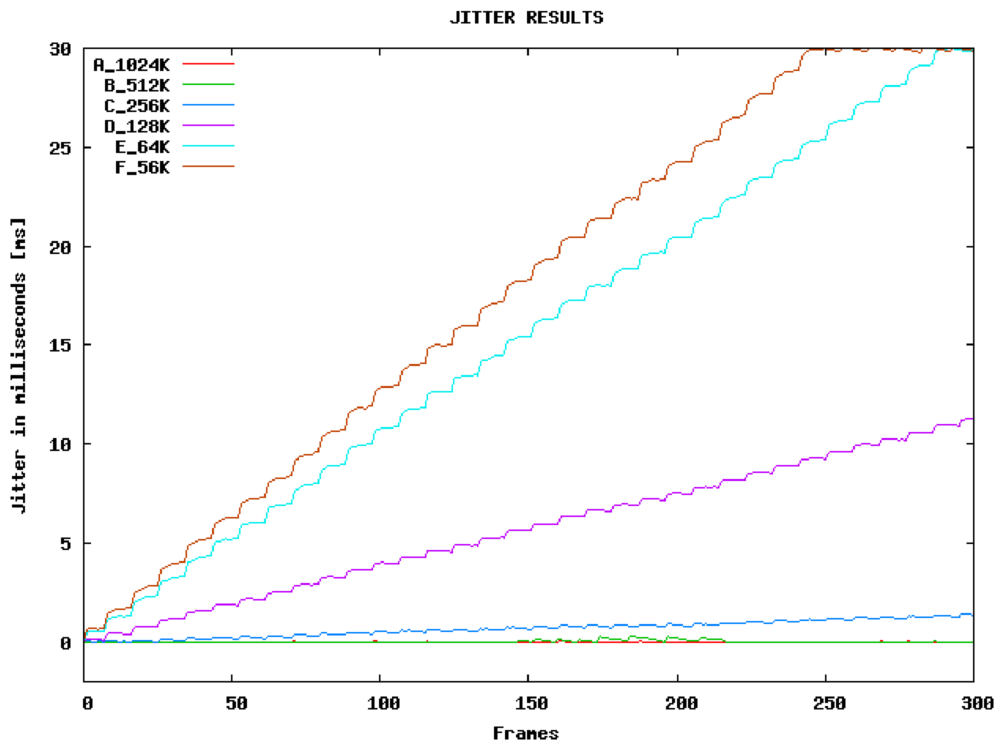


Figure 4.32: Jitter for Akiyo CIF Video on Physical Network

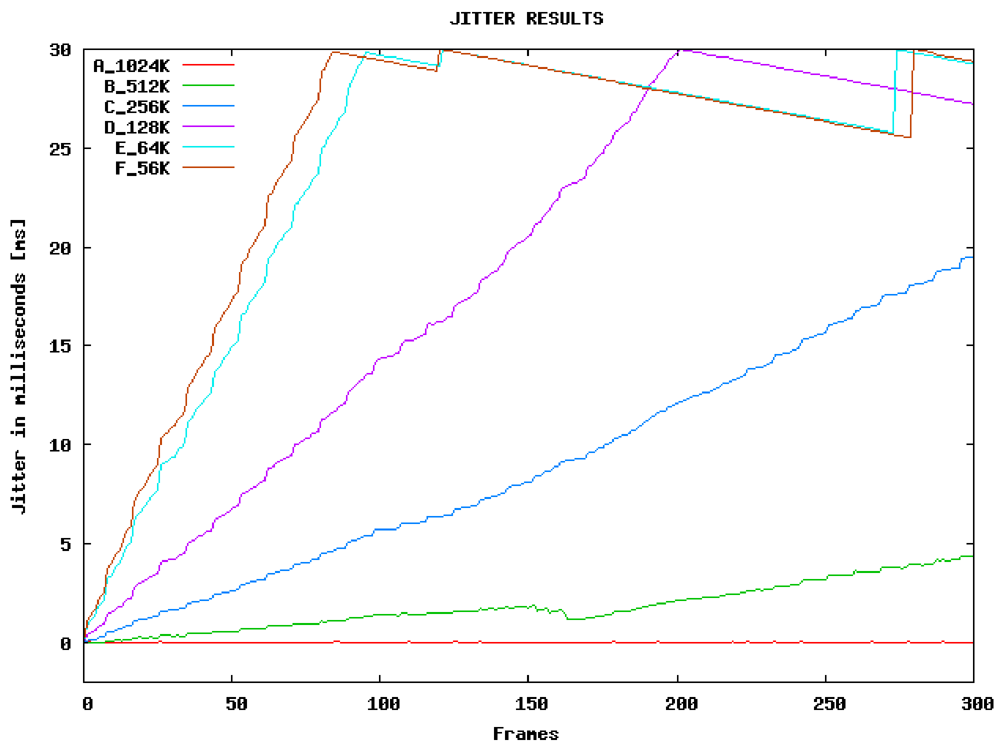


Figure 4.33: Jitter for Foreman CIF Video on Physical Network

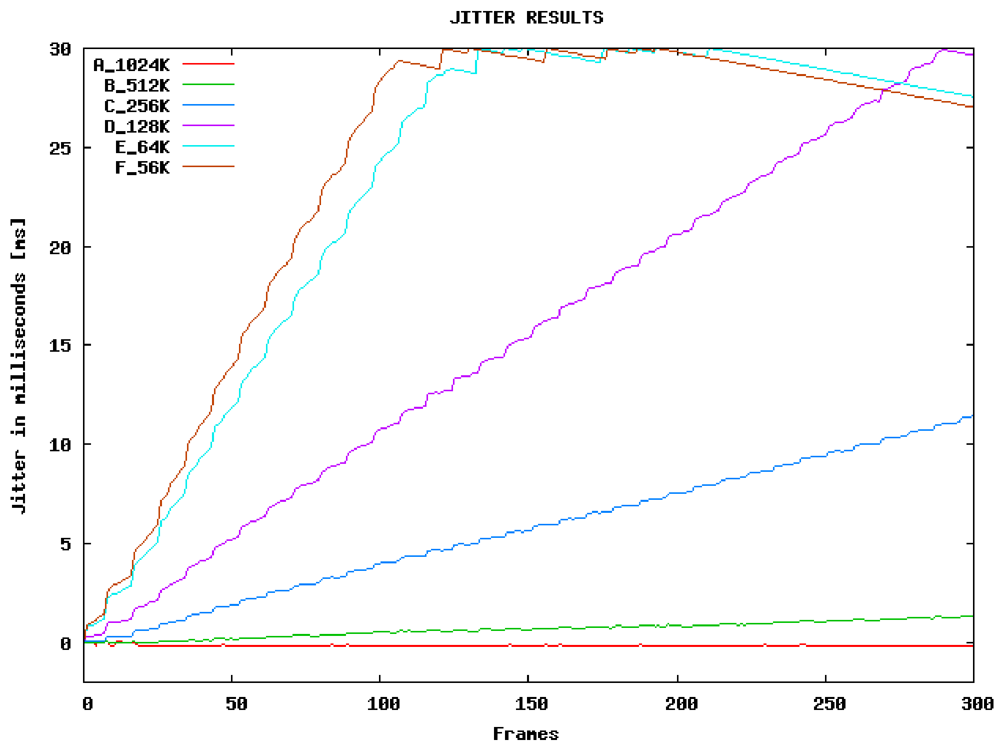


Figure 4.34: Jitter for Hall CIF Video on Physical Network

4.4.4 Cache Comparison

The series of cache comparison tests were performed within the simulated environment (NS-2). These tests aim to determine if there is a difference in the minimum cache sizes (the cache size needed to produce a video MOS of 5) between two consecutive nodes when the bandwidths of these two consecutive nodes change (which may lead to new minimum cache sizes). With reference to Figure 3.9, the cache size in this instance is the minimum cache size needed to compensate for node 2's (client_B) frame loss incurred by the frame loss of node 1 (client_A). For these tests, no other nodes were considered.

To help illustrate the representation of these minimum cache comparisons, Figure 4.35 provides an example of the testing performed. The left bar graph in blue, illustrates the minimum cache sizes for Client A and Client B which each have a 128Kb/s connection. These tests investigate the difference for the minimum cache size of Client B when the bandwidth connection for Client A changes (which may change its minimum cache size).

The result of the new minimum cache sizes is represented by the the red bar graph on the right in Figure 4.35. Figure 4.35 illustrates that the bandwidth change of the network link for Client A from 128Kb/s to 56Kb/s yielded an increase in its minimum cache size from 230 *KB* to 340 *KB*. The minimum cache size for Client B remained at 230 *KB* in this instance.

However, the focus of these tests is whether Client B's minimum cache size changes as a result of the the network link bandwidth change for Client A. These tests are grouped into three Client A and Client B pairs, where the bandwidth link for Client B remains fixed in each pair, whilst the bandwidth link for Client A decreases. The test pairs and the bandwidths of the links tested for the Clients are represented in Table 4.2 which shows the three test pairs (Client B network bandwidth specifications) with the respective network bandwidth changes for Client A (T1, T2, T3 and T4).

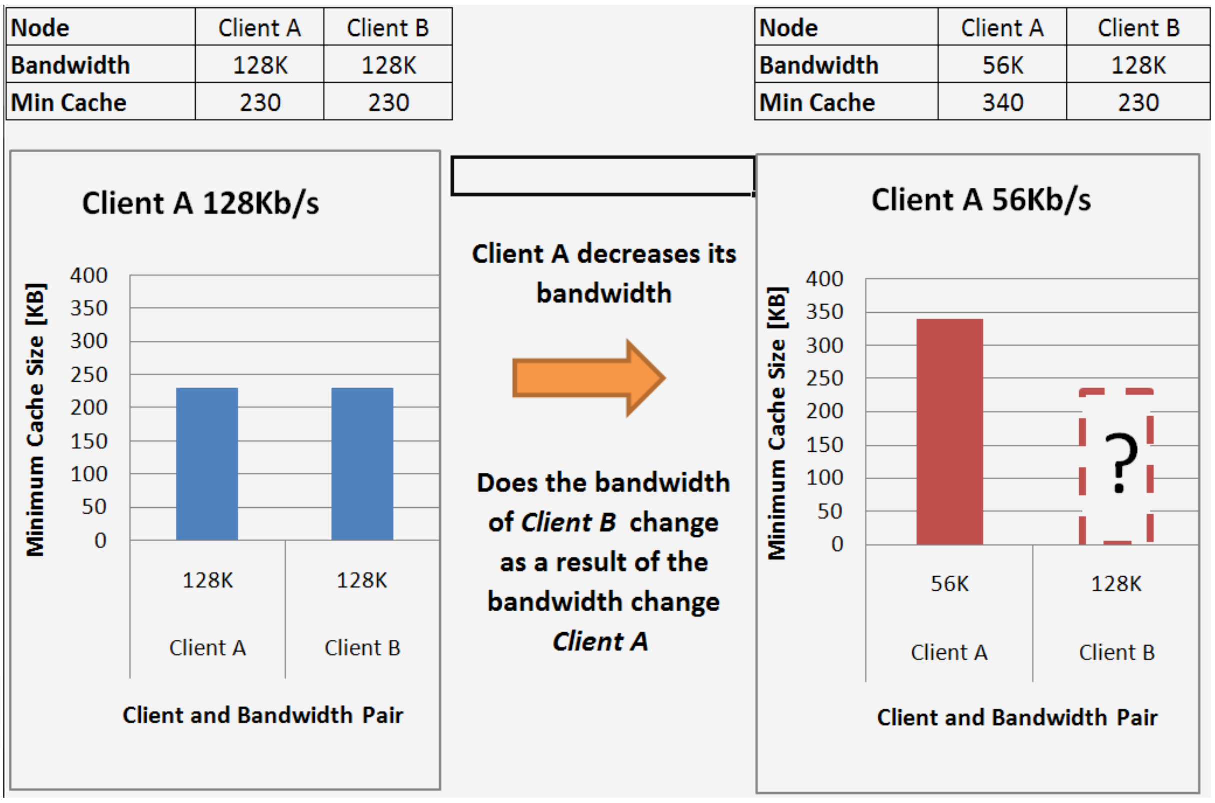


Figure 4.35: Minimum cache size comparison summary of tests

Pair Test (Client B)	Client A T1	Client A T2	Client A T3	Client A T4
1. 256Kb/s	1000Kb/s	384Kb/s	128Kb/s	56Kb/s
2. 128Kb/s	1000Kb/s	384Kb/s	128Kb/s	56Kb/s
3. 64Kb/s	1000Kb/s	384Kb/s	128Kb/s	56Kb/s

Table 4.2: Test pairs and Client network link specifications

The high bandwidth results are placed closer to the front since high bandwidths are expected to yield lower minimum cache sizes. Table 4.2 shows that each pair test consists of four Client A network bandwidth tests (T1, T2, T3 and T4). An example of a complete pair test from Table 4.2 (Pair Test 2. 128Kb/s) is illustrated in Figure 4.36. The yellow arrow displays the focus of the change in results, with the bandwidth link for Client A decreasing away from the *x-axis* along the *z-axis*. This pair illustrates that the bandwidth to Client B remains fixed along this *axis* whilst the bandwidth to Client A decreases as each test is performed.

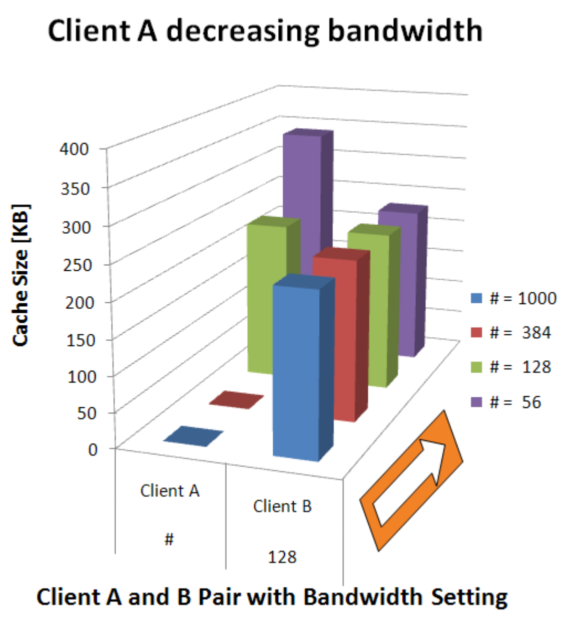


Figure 4.36: Complete Pair test with Client A bandwidth link decreasing

Figure 4.37 represents the three pair tests from the perspective of Client A, where Client A remains fixed and Client B is tested with the following network bandwidth links: 256Kb/s, 128Kb/s and 64Kb/s. The legend on the right illustrates the bandwidth link for Client A which remains constant for all these tests (which is 128Kb/s in this instance). The bandwidth decreases to the right along the *x-axis* as illustrated by the orange arrow.

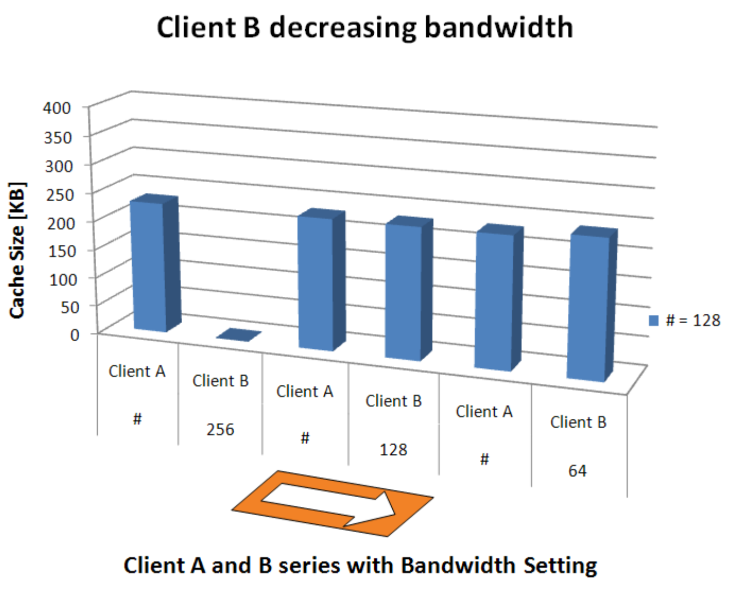


Figure 4.37: Client B results for a fixed Client A test

Figure 4.38 illustrates the cache comparison for the Akiyo CIF video. No cache was required for Client B's 256Kb/s connection. When Client B's connection dropped down to 128Kb/s, for all the Client A tests (T1, T2, T3 and T4), the cache remained at 230 *KB.* However, when Client B's connection was 64Kb/s and Client A's connection was 128Kb/s, the bandwidth required for Client B's 64Kbs reduced from 340 *KB* to the same as that of Client A (which was 230 *KB*). However, this was the only reduction for all these Akiyo CIF video tests.

Figure 4.39 illustrates the cache comparison for the Foreman CIF video and immediately shows larger minimum cache sizes needed for the majority of tests presented. However, unlike the Akiyo CIF video, there are four instances where a similar reduction is expressed. They are shown in Table 4.3 which illustrates Client A's bandwidth connection change, the bandwidth connection for Client B (BW connection) and the cache size reduction for Client B as a result of Client A's change.

Client A connection reduction	Client B BW connection	Client B cache difference
Client A = 1000Kbs -> 384Kb/s	256Kb/s	180 *KB*
Client A = 1000Kbs -> 384Kb/s	128Kb/s	170 *KB*
Client A = 1000Kbs -> 384Kb/s	64Kb/s	70 *KB*
Client A = 384Kbs -> 128Kb/s	64Kb/s	70 *KB*

Table 4.3: Cache comparison changes for Client A and B for the Foreman CIF video

Figure 4.40 shows the cache comparison results for the Hall CIF video, where four noticeable cache size reductions are presented for Client B. Client B with a bandwidth connection of 256Kb/s, experiences a minimum cache size drop by 130 *KB* when Client A reduces its bandwidth connection from a 1000Kb/s connection to a 384Kb/s. Client A still with a 256Kb/s connection experiences a minimum cache size drop of 320 *KB* and when Client A reduces from a 128Kb/s connection to a 56Kb/s connection. Client B (with a 128Kb/s connection) experiences a cache size drop of 60 *KB* when Client A reduces its BW from a 1000Kb/s connection down to a 384Kb/s connection. The last noticeable change is when Client B has a 64Kb/s connection, it experiences a cache size drop of 60 *KB* when Client A reduces from a 384Kb/s to a 128Kb/s connection.

These results indicate that certain low bandwidth combinations can yield decreased minimum cache sizes. Since the minimum cache size would be reduced, this would effectively decrease the wait time for the video stream to be received. Though these values may seem small, when longer videos are streamed across wide network links consisting of numerous connections of different low bandwidths, these cache combinations can decrease the wait time substantially if correctly applied.

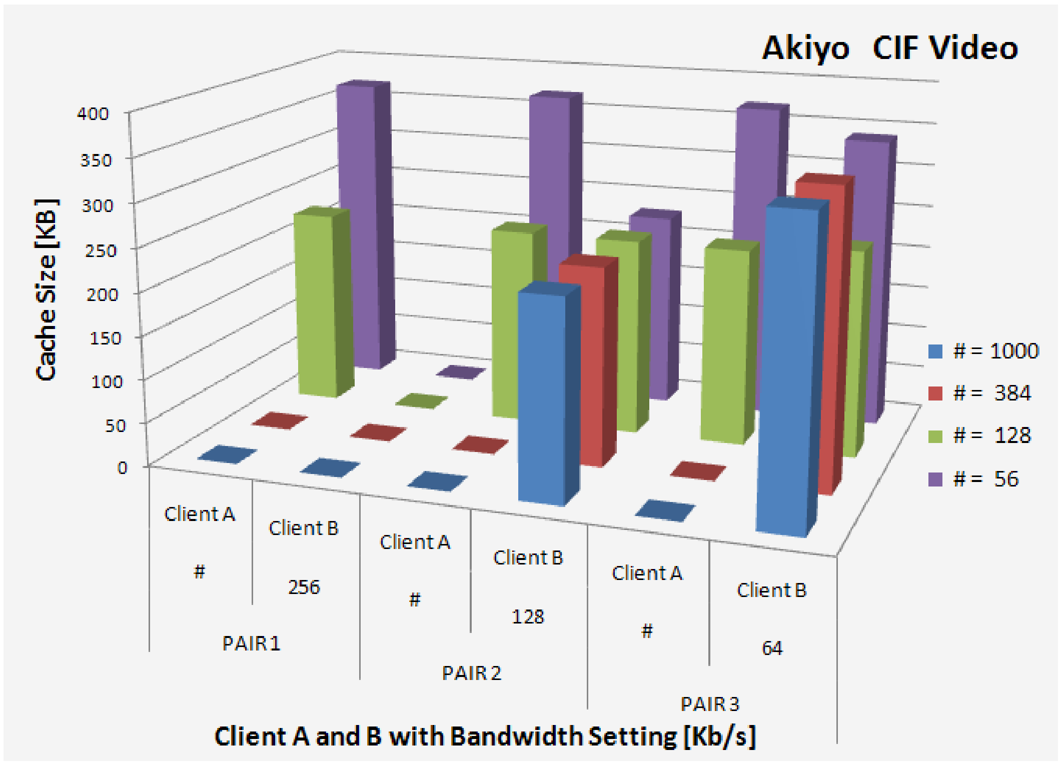


Figure 4.38: Cache Comparison for Akiyo CIF Video

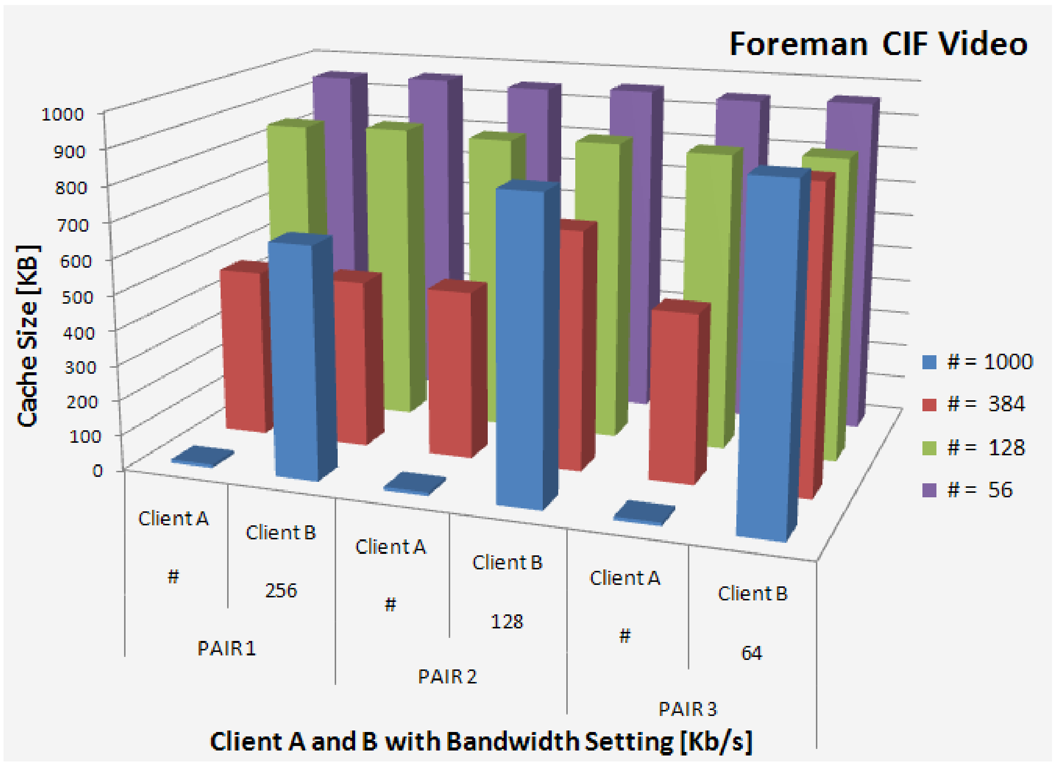


Figure 4.39: Cache Comparison for Foreman CIF Video

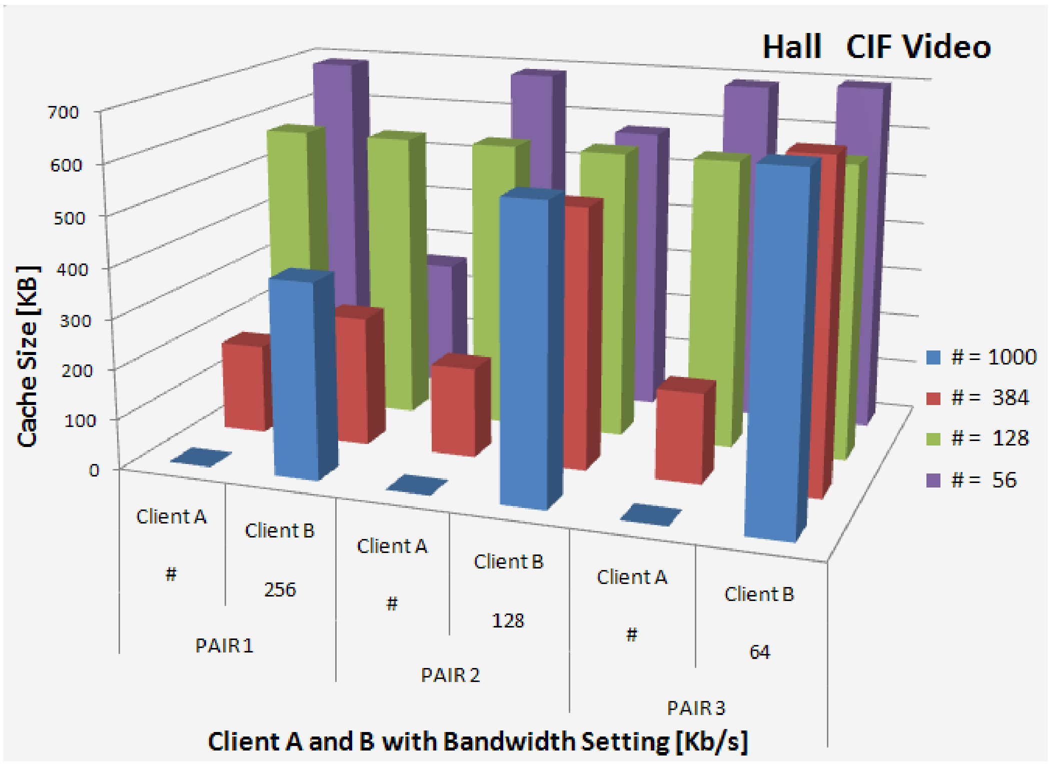


Figure 4.40: Cache Comparison for Hall CIF Video

4.4.5 Placement Difference

As mentioned in section 3.4.2, the illustration 3.8 represents the test setup for the network animation expressed in Figure 3.9. This test aimed to determine if the placement of the node in the network affected the minimum cache size if the links between the nodes were uniform. Figure 4.41 illustrates a simplified representation for this test and the following bandwidths were applied to the red network connection: 56, 64, 128, 256, 384, 512, and 1000 Kb/s.

The test compared the minimum cache sizes for Client B, D and G to determine whether the placement distance away from the server (Client D being further away from server than B and G being further away than both B and D) yielded any difference in the minimum cache size between the Clients if the links between the nodes (the green connections) are uniform. The differences in placement focused on the clients B, D, and G (nodes 2, 4, and 7) as illustrated in Figure 3.8.

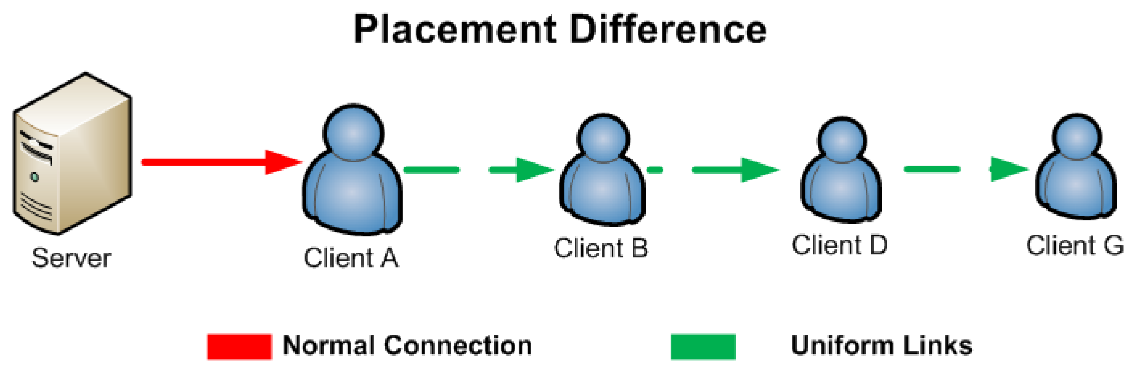


Figure 4.41: Simplified placement difference setup

The results for these tests showed that the minimum cache size differences between the Clients were all zero for all the bandwidths of Client A as expected, since all the links between Clients B, D and G were uniform. Once the appropriate minimum cache size was determined for Client B, the same algorithm was applied for Clients D and G yielding the same minimum cache size regardless of the Client's distance from the server.

4.5 Large Network Prediction Results

The network animation diagram as presented in Figure 3.11 displays a collection of three networks which are presented as follows: The Office network where the external traffic is minimal, the Administrative network where the external network traffic is moderate, and the Internet where the external network is significantly larger than the other two.

Client A (node 1 in Figure 3.11) resides within an office network and is tested with three different bandwidth streams: 1Mb/s, 10Mb/s and 100Mb/s. Client B (node 2 in Figure 3.11) resides within Administrative (Admin) network while Client C (node 3 in Figure 3.11) resides on the Internet. Both Client B and Client C are tested with the same bandwidth streams as Client A. The extra network traffic generated within these networks increases: with the Office network receiving the least amount of extra network traffic, the Admin network getting a moderate increase in extra network traffic and the Internet getting the greatest amount of extra network traffic.

The tests in the following subsections consist of the standardised video streaming distribution, but with the focus on the clients A, B and C. All videos are 10 seconds in duration. However, the traffic generated by all the external nodes will only be transmitted for five seconds because this will help illustrate the impact of the external network traffic within the graphs (for the first half of the test, the external traffic will be applied, and then the stopped for the remainder of the video transmission). The aim is to examine the extent that the levels of noise (or external traffic) decreases the quality of the video stream in each of the tests described. Clients A, B and C will be allocated the following bandwidth connections as shown in Table 4.4.

Bandwidth Connection	**Alternative Name**
1000Kb/s	1.0 Mb/s T1 Line (or T1)
10 000Kb/s	10Mb/s Cable Line (or Cable)
100 000Kb/s	Std 100Mb/s FastEthernet line (or Ethernet)

Table 4.4: Clients A, B and C bandwidth connections

For the purposes of these tests, remodelling a network administration building into a collection of seven nodes and the whole Internet into a collection of 12 nodes serves to control the conditions of the simulations in order to provide a manageable set of results. The results of the tests are presented in the next subsections.

4.5.1 Latency

Due to the cluttered output of having all the Client's latency results for each video illustrated within a single figure, the end to end delays of the network connections for Clients A, B and C where separated within each video. All the figures for these latency tests illustrate the end to end delays of network connections for Clients A, B and C with their respective levels of external network traffic. Each simulation consisted of Clients A, B and C operating with the same network bandwidth connection specification. The simulation was repeated three times with each turn having the bandwidth connection specification adjusted in accordance with Table 4.4.

For all the Figures, it is quite evident to note that the erratic nature of the curves only runs for the first 150 frames (or five seconds). Another observation is that the majority of the high bandwidth curves with low extra network traffic (Client A *Ethernet* streams) are situated by the 0.12 *ms y-axis* latency mark as opposed to the 0 *ms y-axis* mark for all the previous 1Mb/s latency tests from the Peer-to-Peer and Small Network scenarios. This is likely the result of the network distance between the streaming server and the different networks as illustrated in Figure 3.10.

Figures 4.42, 4.43 and 4.44 illustrate the latency tests performed on the Akiyo CIF video for the different Clients A, B and C. It is immediately noticeable that the extra network traffic imposed on the clients has produced an erratic set of latency curves in comparison to the latency test results from the Peer-to-Peer simulated networks tests in Figure 4.1. Client A in the Office network (Figure 4.42) produced low latency ranges with minimum interference from the extra network traffic for the *Cable* and *Ethernet* line tests (illustrated in blue and green lines along the 1.2 *ms* mark) whilst Client C's *T1* connection (from the Internet in Figure 4.44) produced the widest range of the three Akiyo CIF video tests.

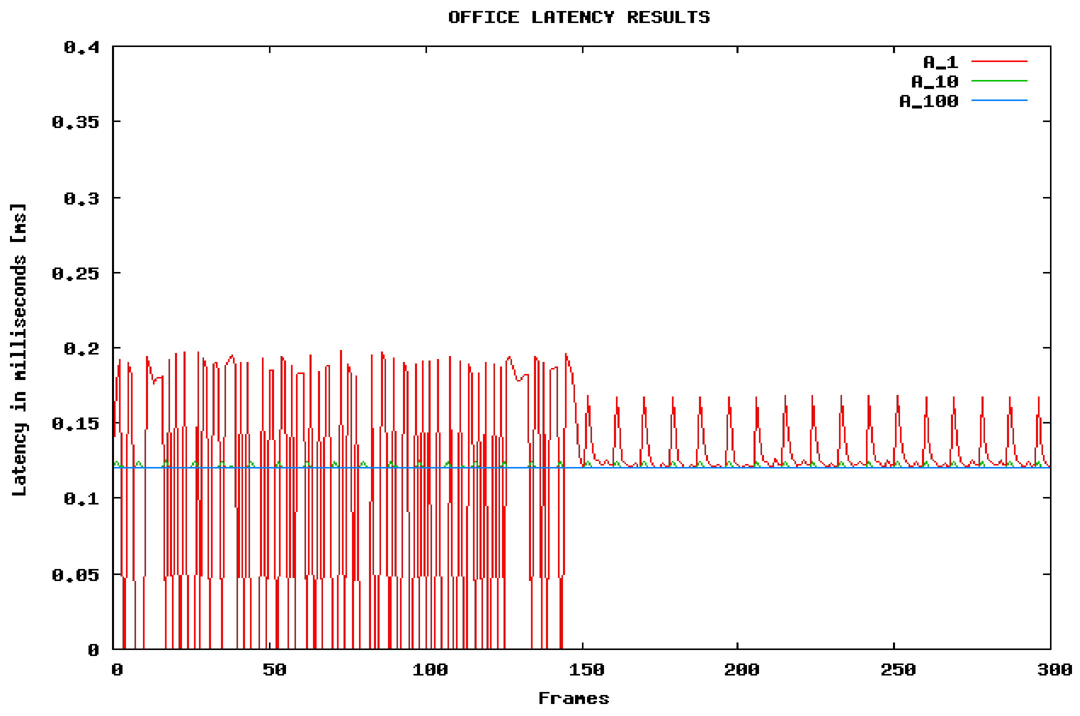


Figure 4.42: Client A Latency for Akiyo CIF Video with External Traffic

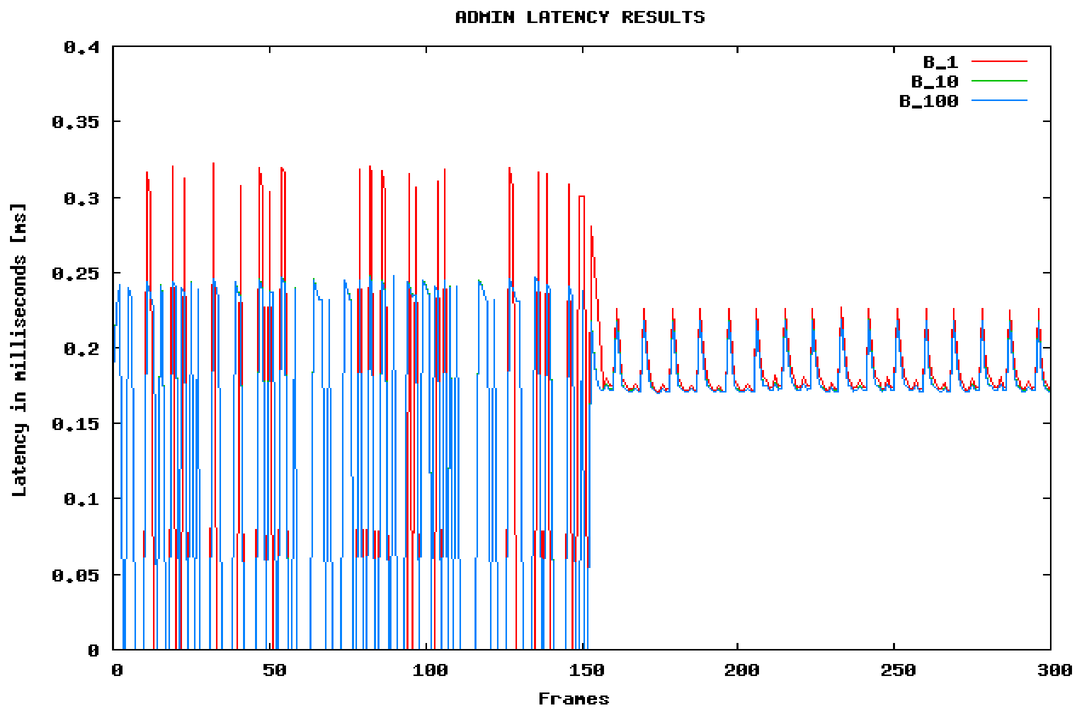


Figure 4.43: Client B Latency for Akiyo CIF Video with External Traffic

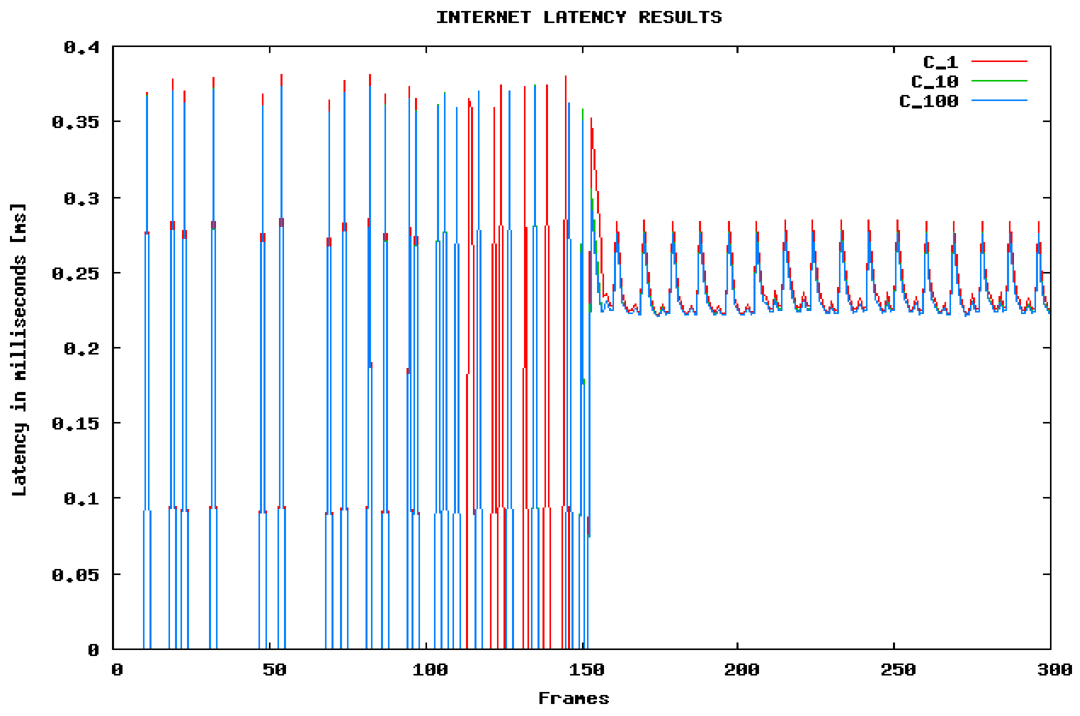


Figure 4.44: Client C Latency for Akiyo CIF Video with External Traffic

Figures 4.45, 4.46 and 4.47 show the results to the Foreman CIF video tests for the different networks and also show an opposite representation to the Akiyo CIF video test results. This is a result of the packet loss encountered during the first five seconds, as opposed to the erratic variation from the Akiyo CIF video. These noise free zones (areas with lost frames) present only the *Cable* and *Ethernet* connections from Client A (Figure 4.45) hovering around the 0.12 *ms* latency mark and occasionally falling to zero towards the end of the stream.

After the 150 frames, the videos latency curves decrease in noise activity, however, the Foreman CIF video shows noisier curves in comparison to the Akiyo CIF representation particularly after the 150 frames mark. This Foreman CIF simulation produces different results in comparison to the Peer-to-Peer scenario latency results in Figure 4.2 as a result of the extra network traffic.

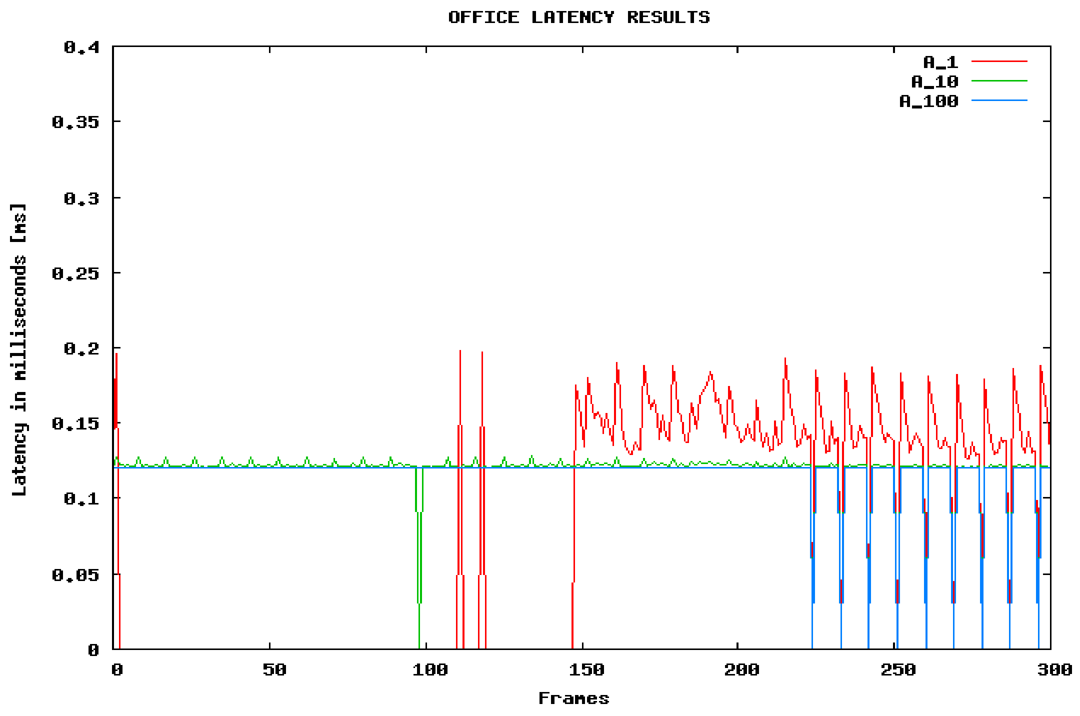


Figure 4.45: Client A Latency for Foreman CIF Video with External Traffic

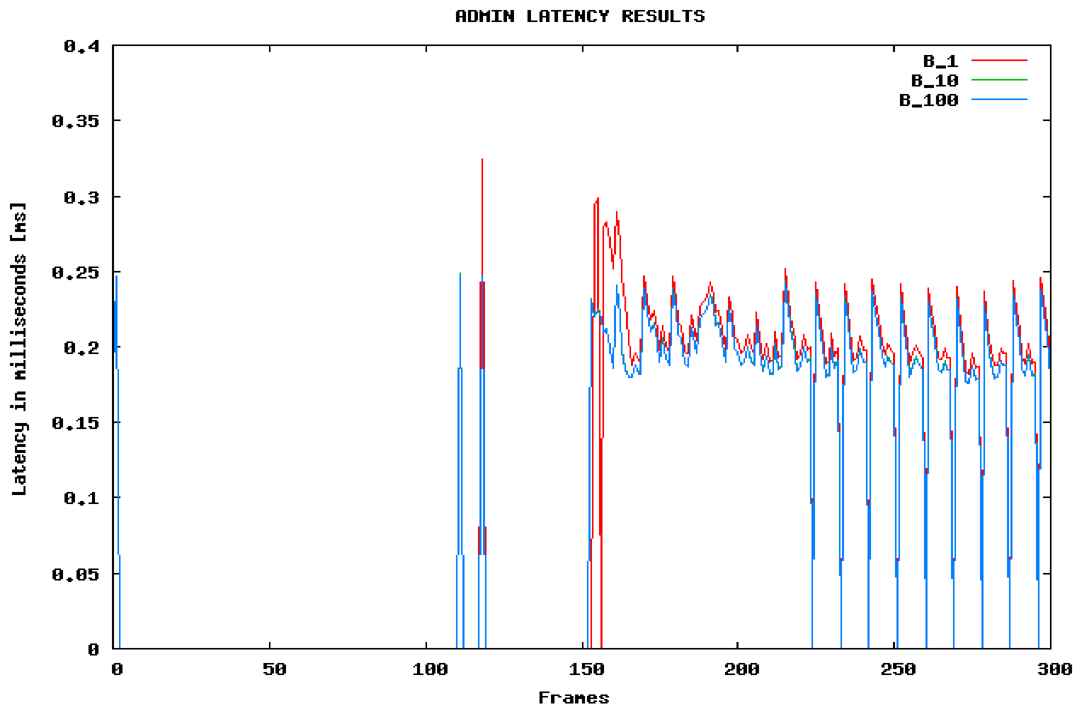


Figure 4.46: Client B Latency for Foreman CIF Video with External Traffic

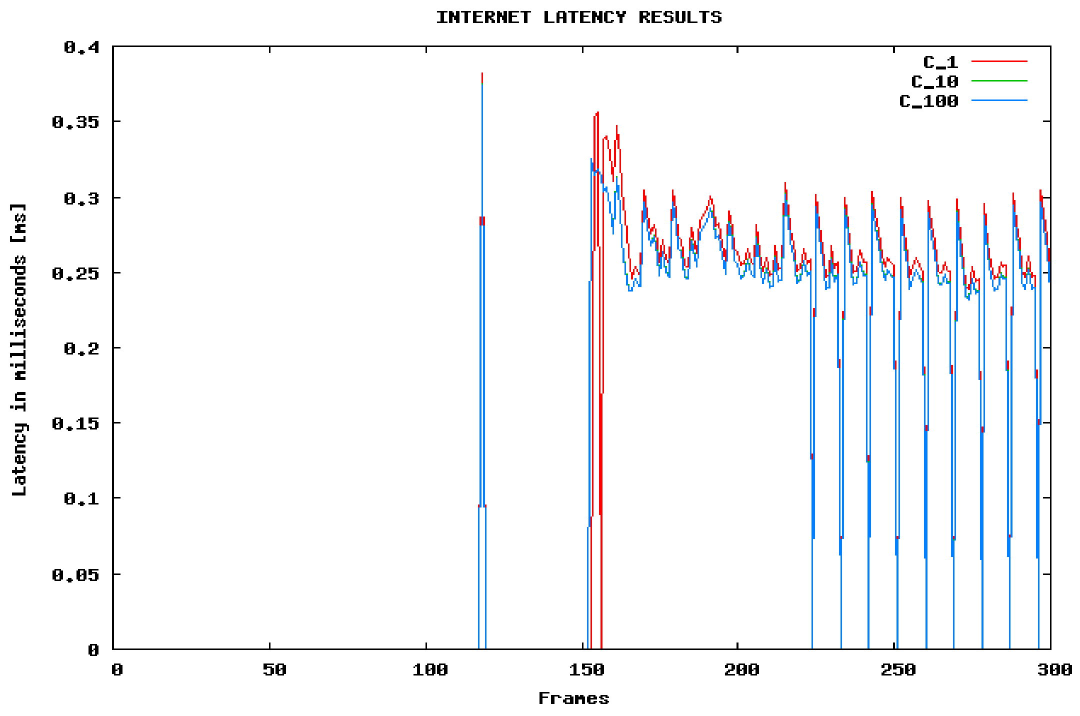


Figure 4.47: Client C Latency for Foreman CIF Video with External Traffic

Figures 4.48, 4.49 and 4.44 illustrate the latency curves for the Hall CIF video which present a similar representation to the Foreman CIF video, except this video experiences less latency. This is evident by the shorter noise free representation where the transmission resumes 30 frames before the half-way frame marker (150 frames along the *x-axis* and not including the middle spike). After the five seconds of noise (external network traffic), the graphs almost resemble the transmission of the Akiyo CIF video, even though the first half resembled that of the Foreman CIF video. However, the similar latency "gaps" experienced in this video and the Foreman CIF video are most likely linked to the object in scene and camera panning movements within these videos. The Akiyo CIF received the least amount of latency compared to the other two videos, with the Foreman CIF video producing the most latency.

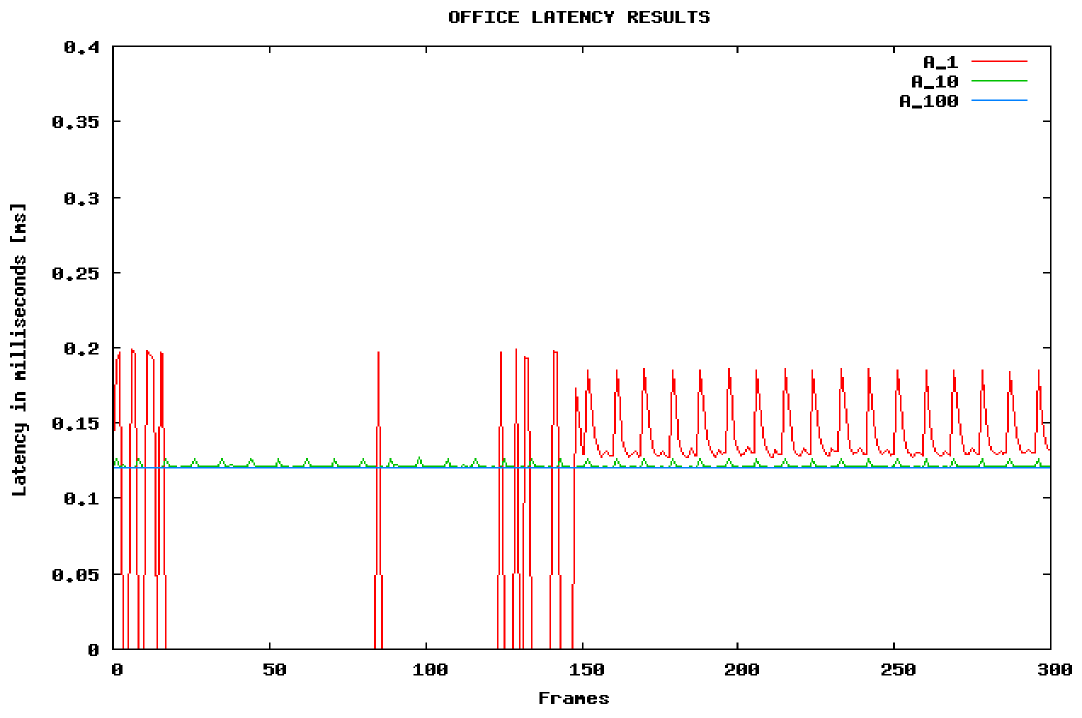


Figure 4.48: Client A Latency for Hall CIF Video with External Traffic

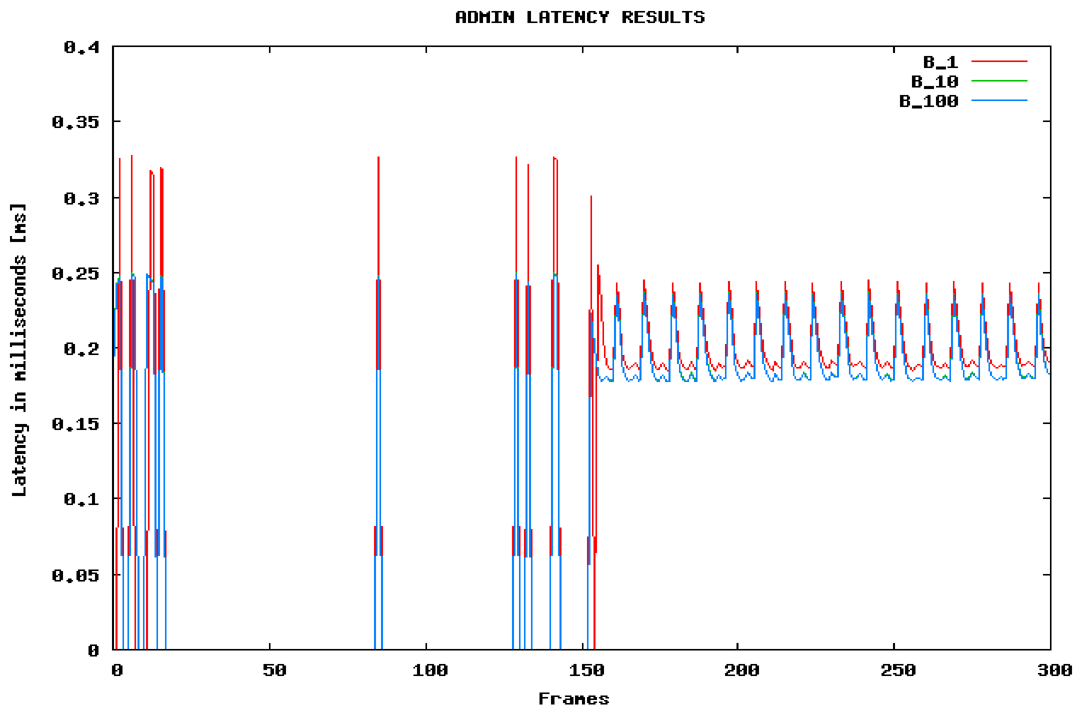


Figure 4.49: Client B Latency for Hall CIF Video with External Traffic

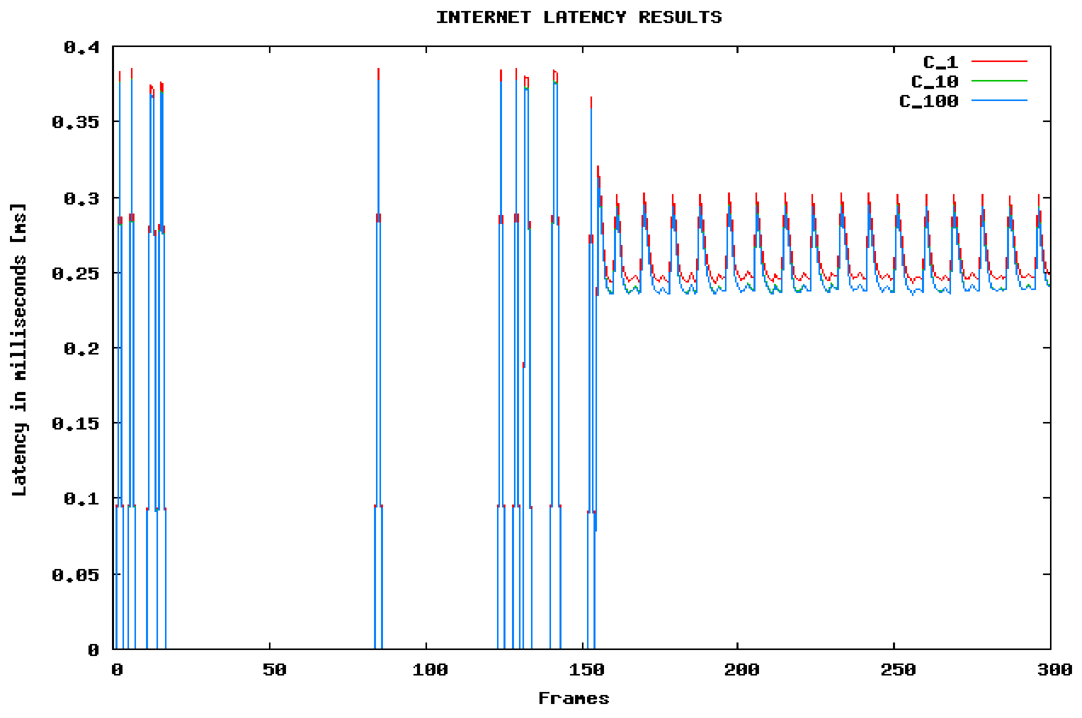


Figure 4.50: Client C Latency for Hall CIF Video with External Traffic

4.5.2 Frame Loss

Figures 4.51, 4.52 and 4.53 illustrate the percentage frame loss for Clients A, B and C within their respective networking conditions. The percentages of lost frames presented within these graphs are due to external traffic imposed on them. Though it may not be clear where the external network transmission stops and starts within these graphs, the bandwidth connections for all these tests are capable of providing video streams with a MOS of 3-5 (depending on the movement within the video) without the presence of external network traffic (as discovered by results of section 4.3.2).

For the Akiyo CIF video, it is evident that the Client B streams experienced less frame loss than the Client C streams, whereas with the Hall CIF and Foreman CIF videos, this difference was not present. For these two videos (Hall CIF and Foreman CIF), the frame loss remained constant across the board (except for the Client A *Cable* and *Ethernet* streams) which indicates that the average frame loss caused by the extra network traffic from the Admin network and the Internet, results in the loss of about 50% of the video.

Since the external network traffic was only present for 50% of the video stream, the average percentage frame loss is close to 50% (for all the tests) for the Foreman CIF video and just below 50% for the Hall CIF video which coincides with the latency graphs as well. The loss for these two videos also explains the significant latency gaps experienced with the results from the Foreman CIF and Hall CIF latency Figures (4.47 and 4.50 respectively). However, even though there were no latency gaps present for the Akiyo CIF videos during the first 150 frames, there was still evidence of frame loss as illustrated in Figure 4.51.

For each of the videos, it is clear that Client A with an *Ethernet* connection experienced the least amount of frame loss, followed closely by Client A with the *Cable* connection. However the dissimilarities between the Akiyo CIF video and both the Foreman CIF and Hall CIF videos begin to show with respect to Client A's *T1* representations. Another observation is that even though the Figures for the Foreman CIF and Hall CIF videos are very similar, the I-Frame loss is about 20% higher for the Foreman CIF video than the Hall CIF video.

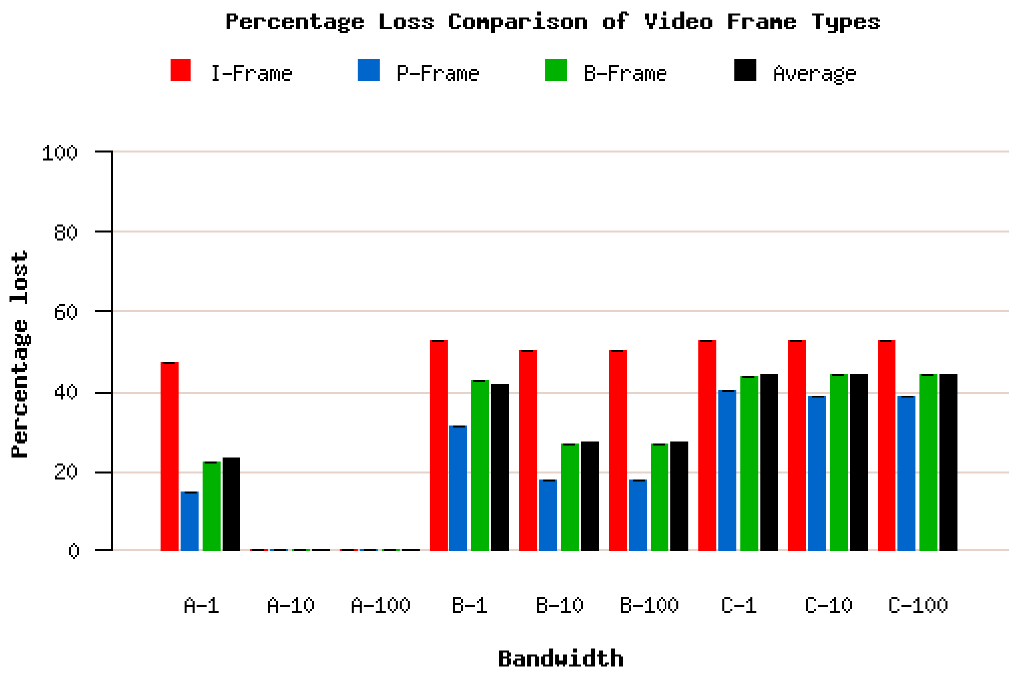


Figure 4.51: Frame Loss for Akiyo CIF Video with External Traffic

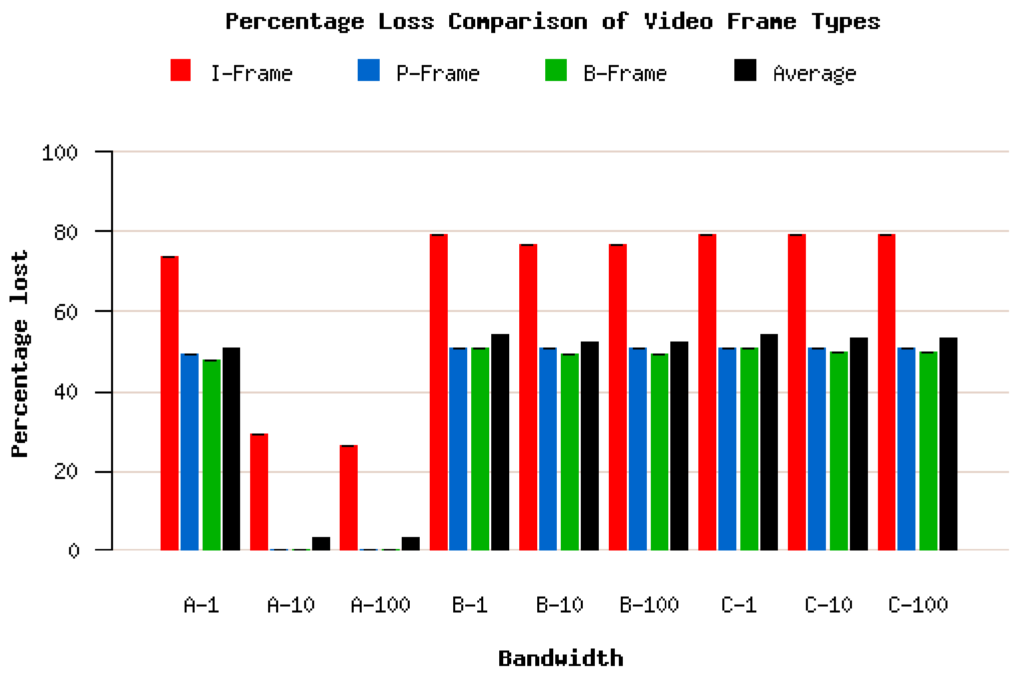


Figure 4.52: Frame Loss for Foreman CIF Video with External Traffic

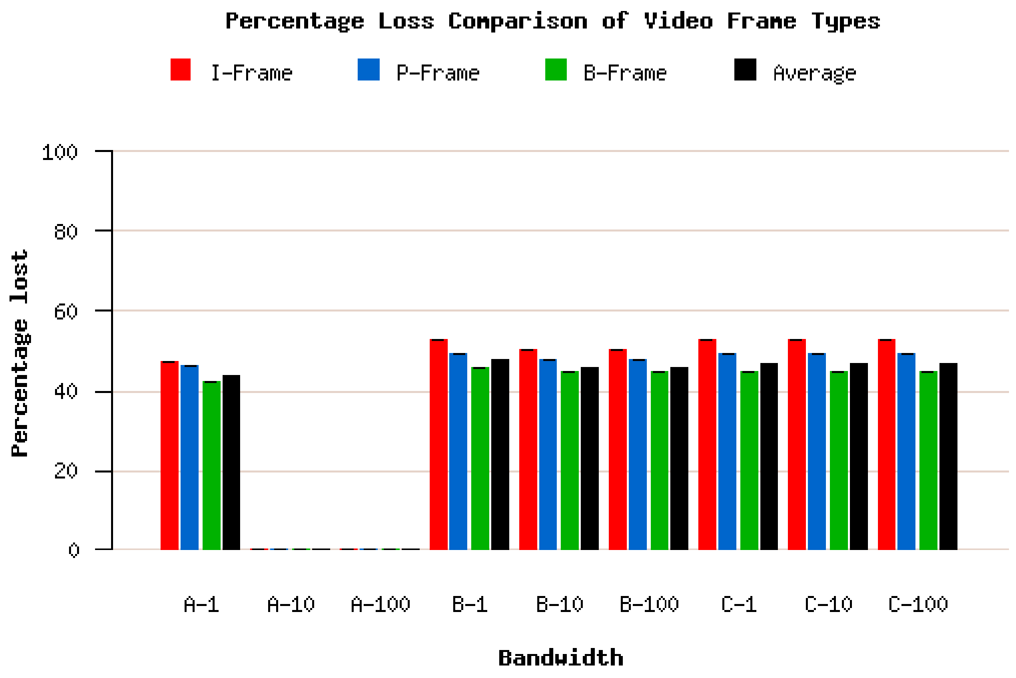


Figure 4.53: Frame Loss for Hall CIF Video with External Traffic

4.5.3 Jitter

Figures 4.54, 4.55 and 4.56 display the jitter for the Akiyo CIF, Foreman CIF and Hall CIF videos by Clients A, B and C as a result of their respective external network traffic conditions. These latency curves follow the same format as the previous set of latency curves from both the Small Network and the Peer-to-Peer scenarios except with a slight alteration to the *y-axis*. These graphs illustrate the full 300 video frames as the *x-axis* and the jitter measured in milliseconds *(ms)* for the *y-axis* with a range of -3.5 to 0.5 *ms*.

The Akiyo CIF video (Figure 4.54) shows the extra network traffic presence by the negative jitter curves for both Client B's *T1* connection stream and all of Client C's connection streams (*T1*, *Cable* and *Ethernet*). However, even after the external network traffic was removed from the video transmission (five seconds into the test), the jitter still remained at the same horizontal level for these bandwidths, but the variation for these curves reduced significantly at this point.

Though not clearly visible, both the *Cable* and the *Ethernet* streams for Client A were situated along the 0 *ms* jitter level while the *T1* connection for Client A was situated relatively close to the 0 *ms* jitter level (in comparison to the other curves).

The Hall CIF video (Figure 4.56) shows a significant increase in the start for the negative jitter with a jitter progressive drop down to -2.3 *ms* followed by an immediate return to 0 *ms* level. However another progressive drop occurs to a second jitter level of -1.5 *ms*. Another immediate vertical return happens, this time maintaining a level close to the -0.2 *ms*.

A noticeable relationship exists between these progressive jitter drops for the Clients (Client A's *T1* stream and all of Client B's and C's streams) and the gaps within the equivalent latency results (Figures 4.48, 4.49 and 4.44). When the latency gap occurs, the jitter results begin to decrease steadily, but when the latency results resume, this is indicated by the sharp jump back towards the 0 *ms* level for the jitter graphs.

The Forman CIF video (Figure 4.55) represents a similar output to the Hall CIF video results except with a larger progressive drop down to -5.0 *ms* for all the Client C streams.

This drop displays the complete frame loss for the following Client transmissions: Client B's *T1* stream and all the Client C's streams. Client A's *T1* stream experienced frame loss for the first 110 frames, but recovered for a short period, as illustrated by the red line in Figure 4.55 (which lies behind the brown line until about half way through the graph). Again this recovery is visible by the spike within the Foreman CIF latency graph 110 frames into the graph (as shown in Figure 4.47).

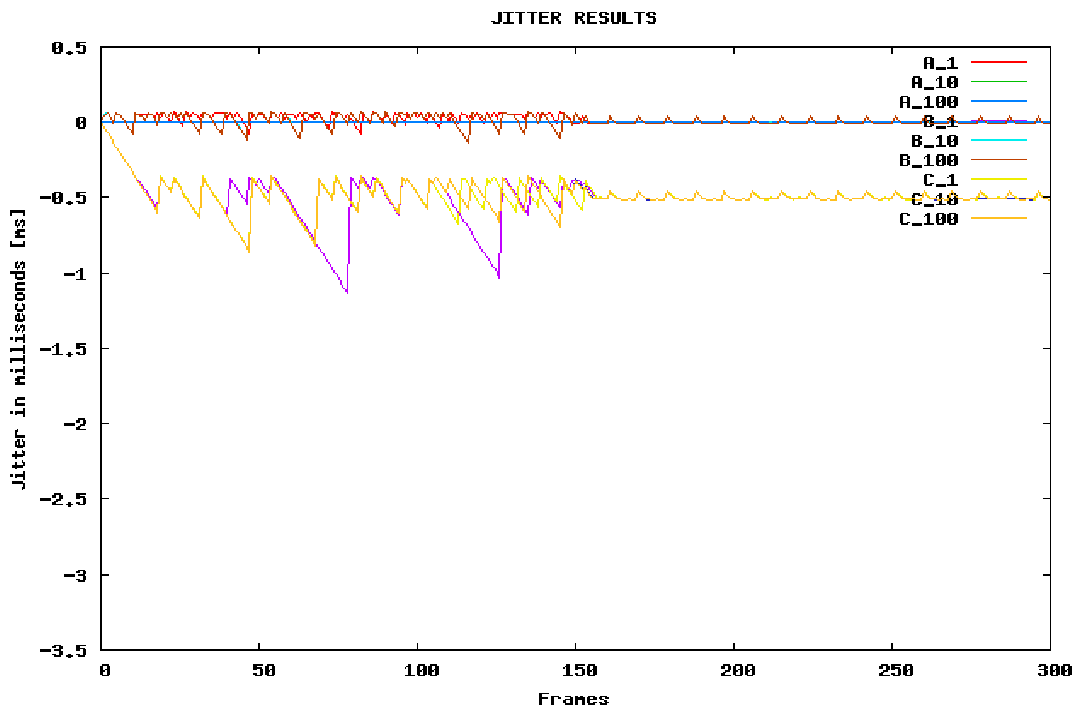


Figure 4.54: Jitter for Akiyo CIF Video with External Traffic

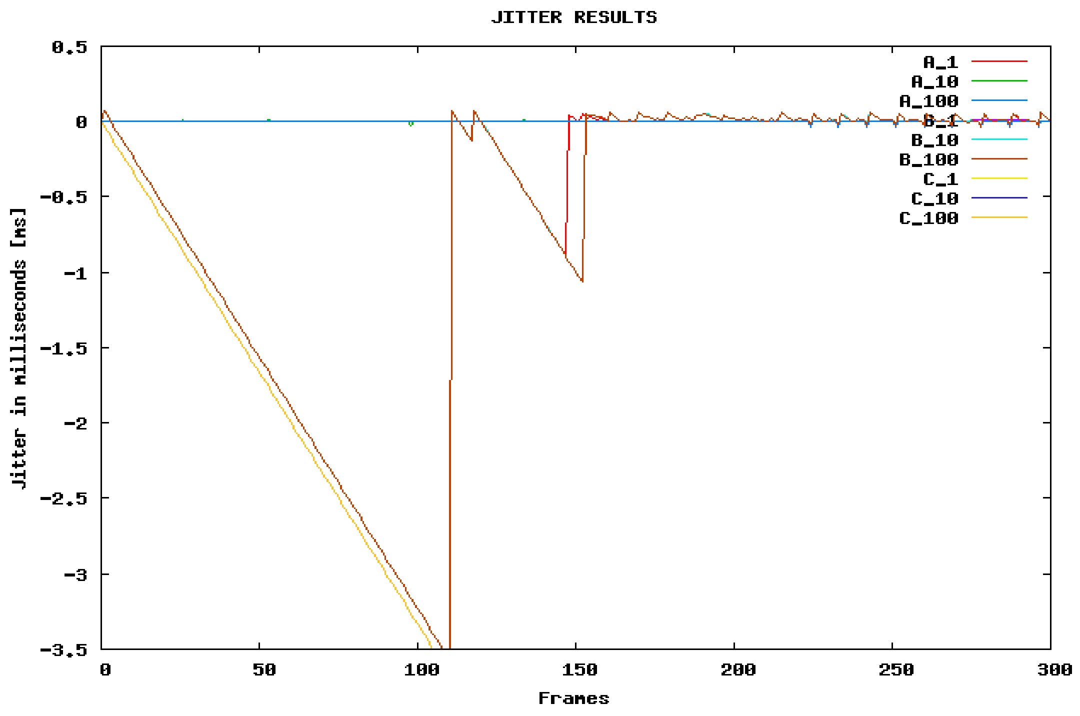


Figure 4.55: Jitter for Foreman CIF Video with External Traffic

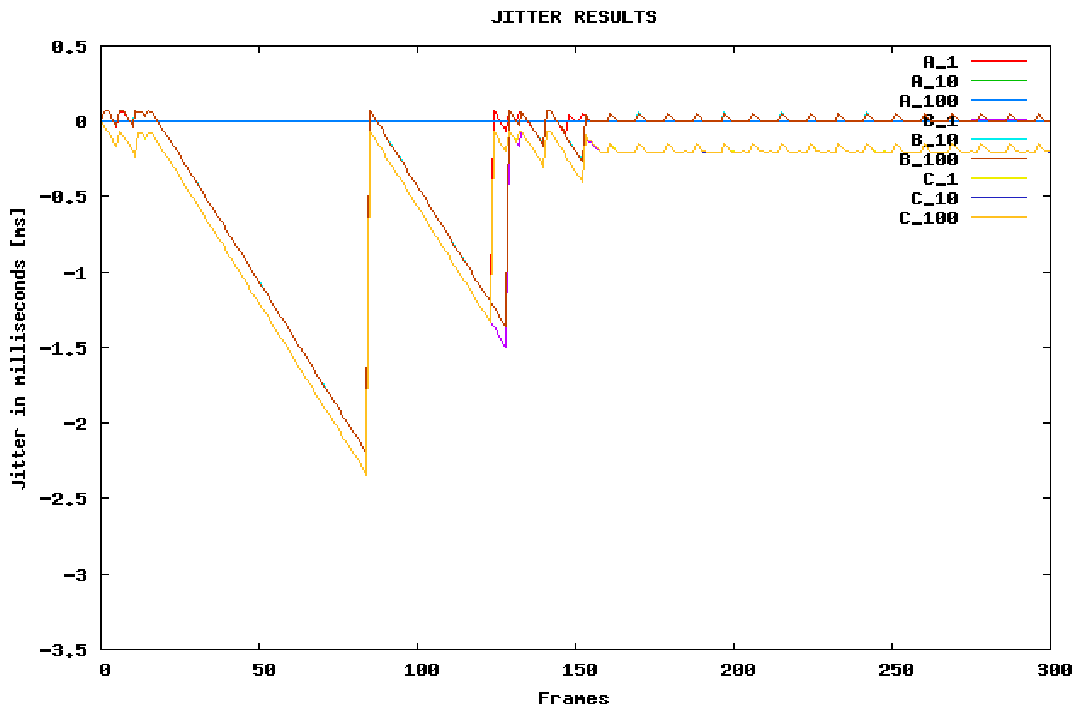


Figure 4.56: Jitter for Hall CIF Video with External Traffic

4.5.4 Average PSNR Values with Standard Deviation

Figures 4.57, 4.58 and 4.59 present the peak signal to noise ratio for Clients A, B and C attributed to their different external network traffic scales. These graph representations are similar to the previous two scenarios in their construction except with modifications to the *axes*. The *y-axis* still represents the PSNR value achieved for each test except with a range of 0 to 45 *dB* (down from 50 *dB*). The *x-axis* represents each Client with its three different bandwidth connection speeds in *Mb/s*.

The results presented in Figure 4.57 display the PSNR averages for the streams of the Akiyo CIF video. This image shows that the Client A *Cable* and *Ethernet* streams produced high PSNR averages with low standard deviations. The *T1* stream for Client A and both Client B's *Cable* and *Ethernet* streams all achieved above 25 *dB* with slightly larger standard deviation values.

An interesting result is noted comparing these average PSNR results and the results of the Peer-to-Peer scenario tests as illustrated in Figure 4.13. This observation shows that the lower results from this Akiyo CIF video test (Figure 4.57) have higher standard deviations than the same Peer-to-Peer simulated tests. This is largely due to the extra network traffic experienced by these Clients. The Internet also provided an interesting observation showing that regardless of the size of the bandwidth connection for Client C, both the average PSNR and the standard deviation remained unaffected. This shows that the external network traffic is capable of decreasing the quality of high bandwidth connections.

Figure 4.59 illustrates the averages for the Hall CIF video. There is an overall decrease for all the tests in relation to the Akiyo CIF video, with the top performers just shy of the 37 *dB* PSNR value. In comparison to the Akiyo CIF video, the decrease in the scores for the Internet is larger than the decrease for the Office network. However, the standard deviation remained the same between the two videos. This shows that the object movements within the scene for this video yielded lower scores than the Akiyo CIF video.

Figure 4.58 (the Foreman CIF video) yielded significantly lower scores for all the tests. However, there are noticeable changes to the results of the standard deviations experienced by these tests. The highest scoring bandwidths tests (Client A's *Cable* and *Ethernet* streams) scored just above 30 and 25 *dB* respectively, but produced larger standard deviations than both the Hall CIF and Akiyo CIF videos.

A secondary observation for the Foreman CIF video was the low scores achieved by Client B's *T1* connection and all of Client C's streams. The scoring below 5 *dB* suggests extremely poor quality. However, even with these extremely low PSNR scores, the average frame loss was still only 50% for the majority of these tests (excluding the Client A *Cable* and *Ethernet* tests) as illustrated in Figure 4.52. This was primarily the result of the external traffic only being distributed for half of the stream transmission.

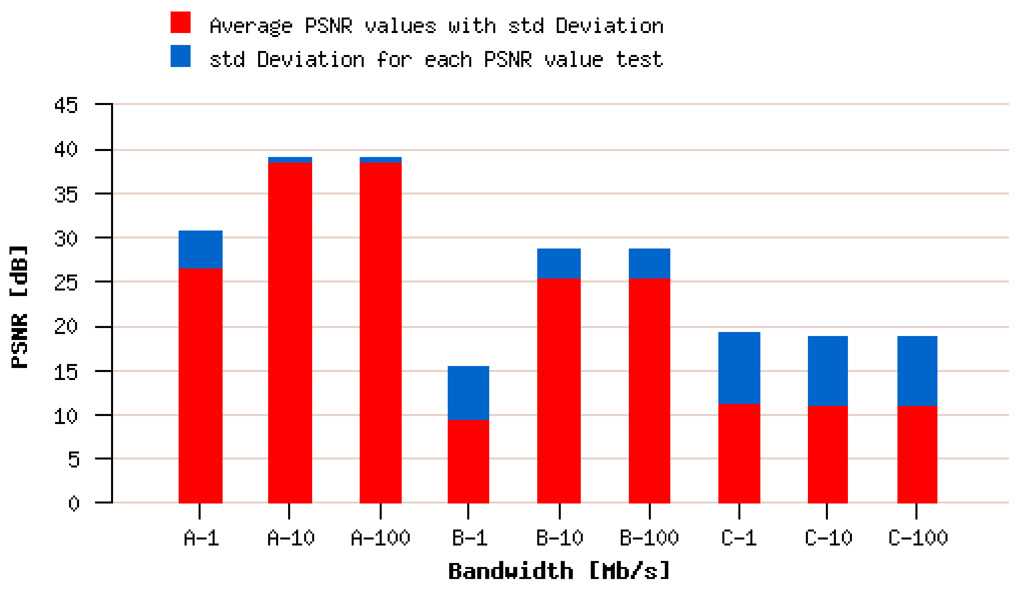


Figure 4.57: PSNR Values & STD Deviation for Akiyo CIF Video with External Traffic

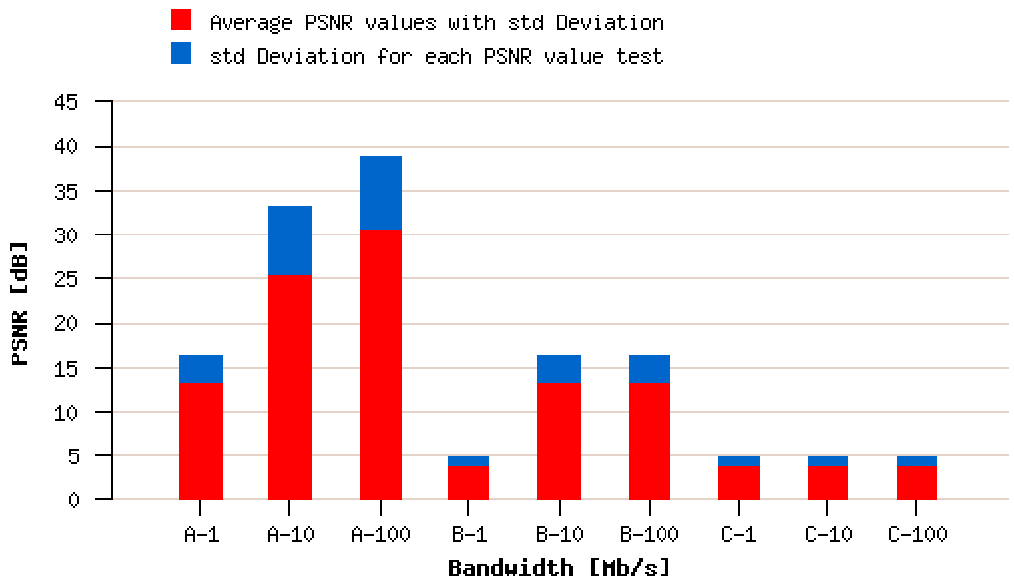


Figure 4.58: PSNR Values & STD Deviation for Foreman CIF Video with External Traffic

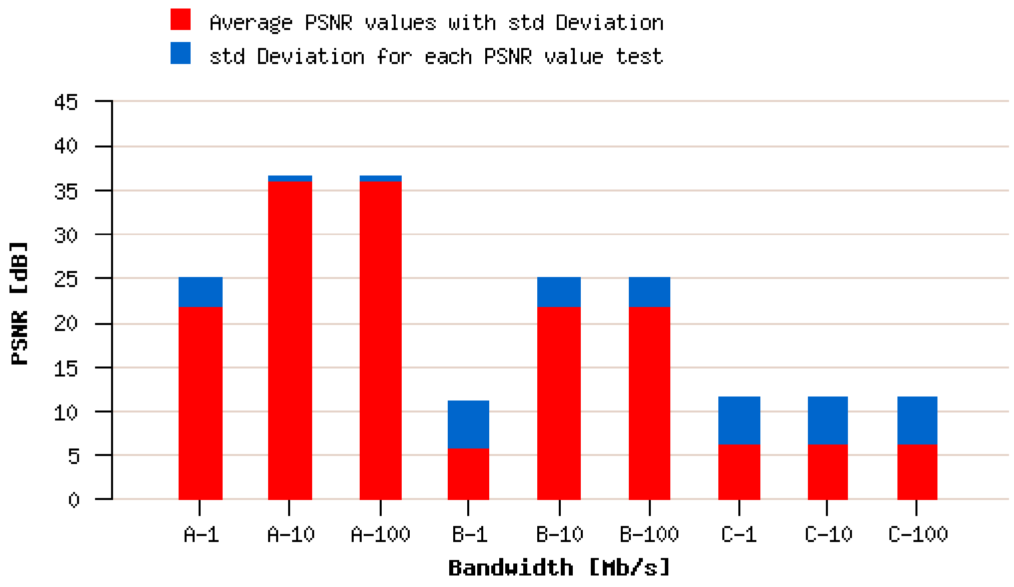


Figure 4.59: PSNR Values & STD Deviation for Hall CIF Video with External Traffic

4.5.5 MOS

The graphs in Figures 4.60, 4.61 and 4.62 represent the Mean Opinion Scores for the video stream transmissions to Clients A, B and C. The *x-axis* scale is the same presentation as the previous section 4.5.4 with the *x-axis* consisting of each Client with its successive bandwidth connection tests (*T1*, *Cable* and *Ethernet*). The scores are based on the previous sections results (PSNR averages).

From Figures 4.60 for the Akiyo CIF video, 4.61 for the Foreman CIF video and 4.62 for the Hall CIF video, it is clear that the Akiyo CIF video scored the highest set of results, with Client A's *Cable* and *Ethernet* streams obtaining exceptional video qualities as shown in the Figure 4.60. At the same time, the Akiyo CIF video was still able to produce average qualities for the Client A's *T1* stream and the two upper bandwidth Client B streams. The remaining streams produced unacceptable qualities, which is interesting, as even with an *Ethernet* connection, Client C still only produced an unacceptable quality rating.

Client A's top results for the Hall CIF video only managed to produce decent scores, with the next top three scores providing below average results as illustrated in Figure 4.62. This is largely attributed to the increased object movement within scene. Once again the Internet produced low scores with unacceptable ratings achieved for all the Client C tests. It is also interesting to note that Client A's *T1* connection from the Office network achieved a higher score than any of the Client C tests. If these Hall CIF scores are compared to the Peer-to-Peer scores attained in Figure 4.18, it is evident that a 512Kb/s connection from a Peer-to-Peer setup can achieve a higher quality rating than an *Ethernet* connection for both the Client B and C networks (assuming that the 512Kb/s connection is not affected by external network traffic).

These results show that the Akiyo CIF video yielded higher score ratings than the Hall CIF video and the Hall CIF video scored higher than the Foreman CIF video (Figure 4.61). The bandwidth played a significant role in the overall performance of the video streams, except in special cases (like the Foreman CIF video streams under extensive external network traffic) where the increase in bandwidth did not necessarily yield the expected increases in performance. These results demonstrate an increase in performance in relation to bandwidth increase.

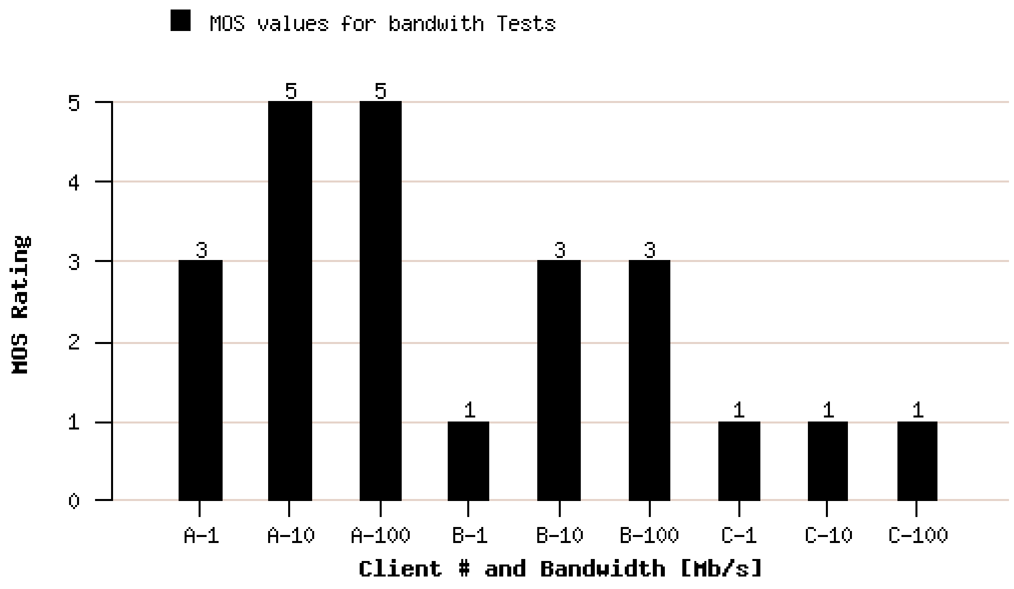


Figure 4.60: MOS for Akiyo CIF Video with External Traffic

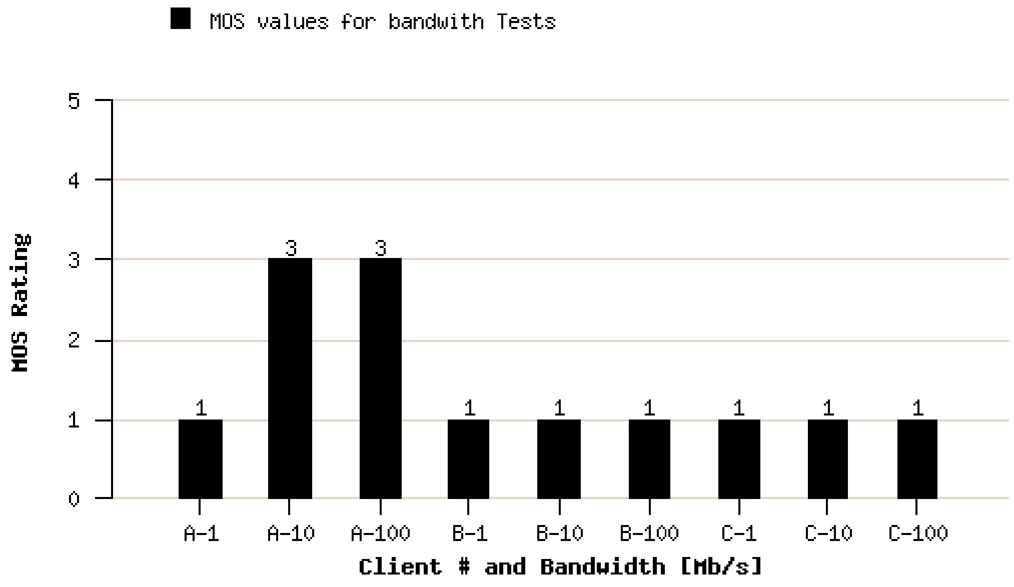


Figure 4.61: MOS for Foreman CIF Video with External Traffic

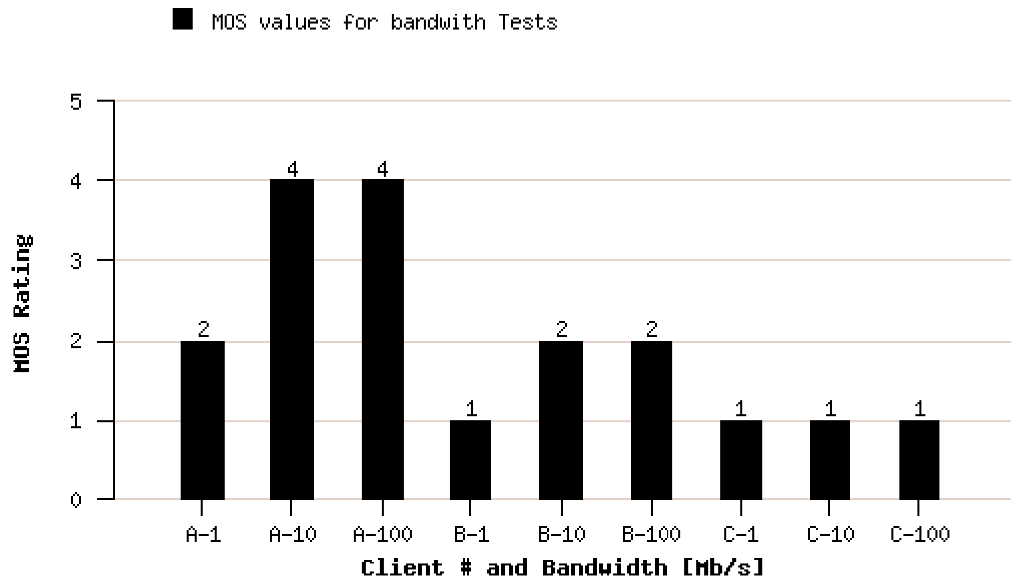


Figure 4.62: MOS for Hall CIF Video with External Traffic

4.6 Low Resolution Examples

The results from the tests presented within this chapter were collected from CIF (352x288) resolution videos. Though a majority of these tests were also performed on QCIF (176x144) resolution videos, the results of these tests will only be presented in Appendix D (DVD-ROM). However, the section 4.7 provides a summary for the majority of the tests performed which includes the results for both the high and the low resolution videos. The following are samples of the low resolution results which highlight some of the interesting differences between the high resolution and low resolution test results.

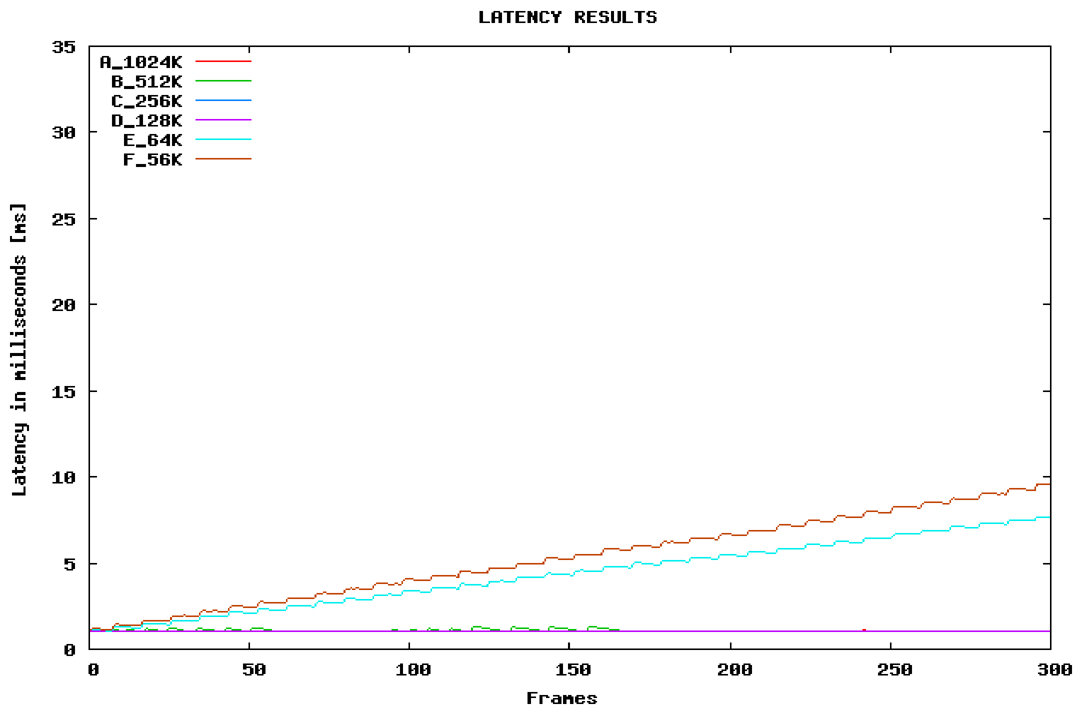


Figure 4.63: Latency for Akiyo QCIF Video from Small Network scenario

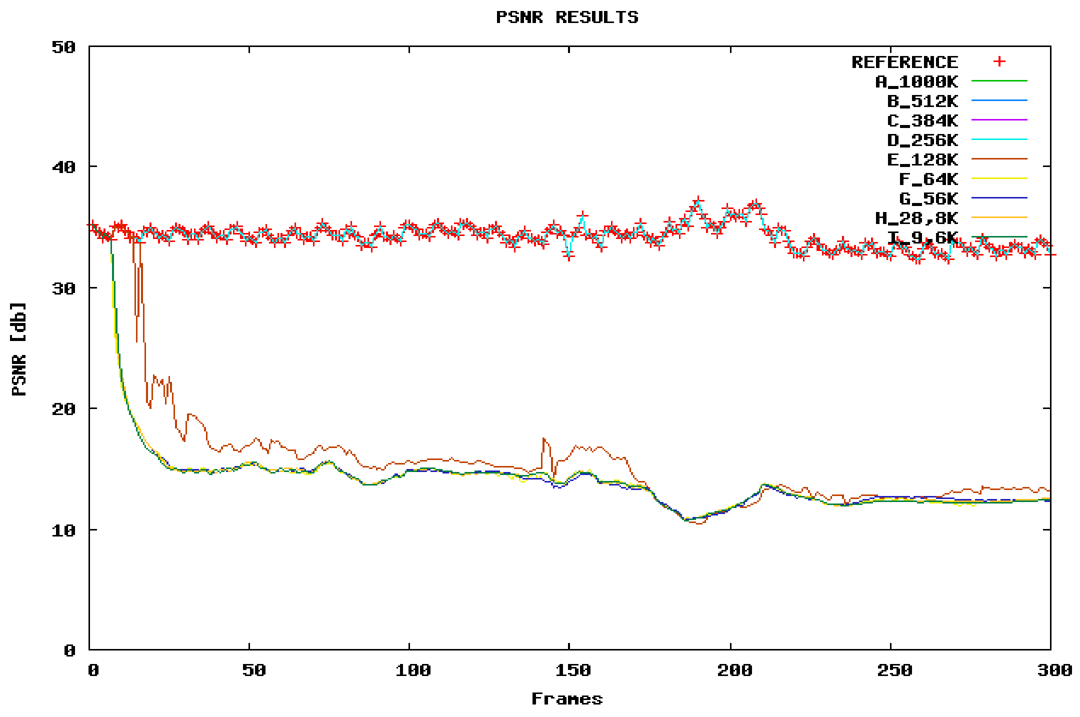


Figure 4.64: PSNR for Foreman QCIF Video from Peer-to-Peer scenario

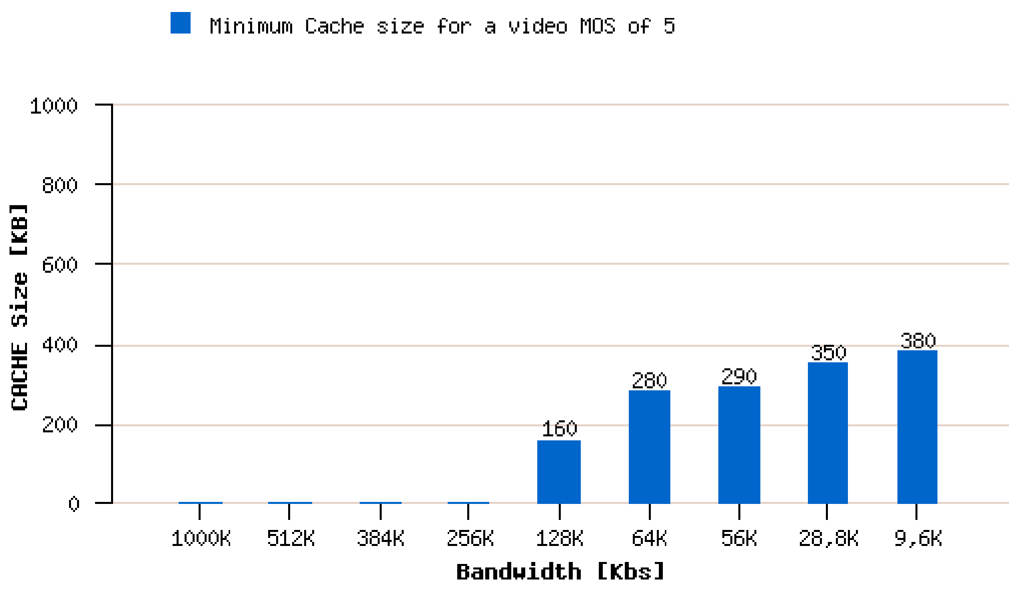


Figure 4.65: Minimum Cache sizes for Foreman QCIF Video from Peer-to-Peer Scenario

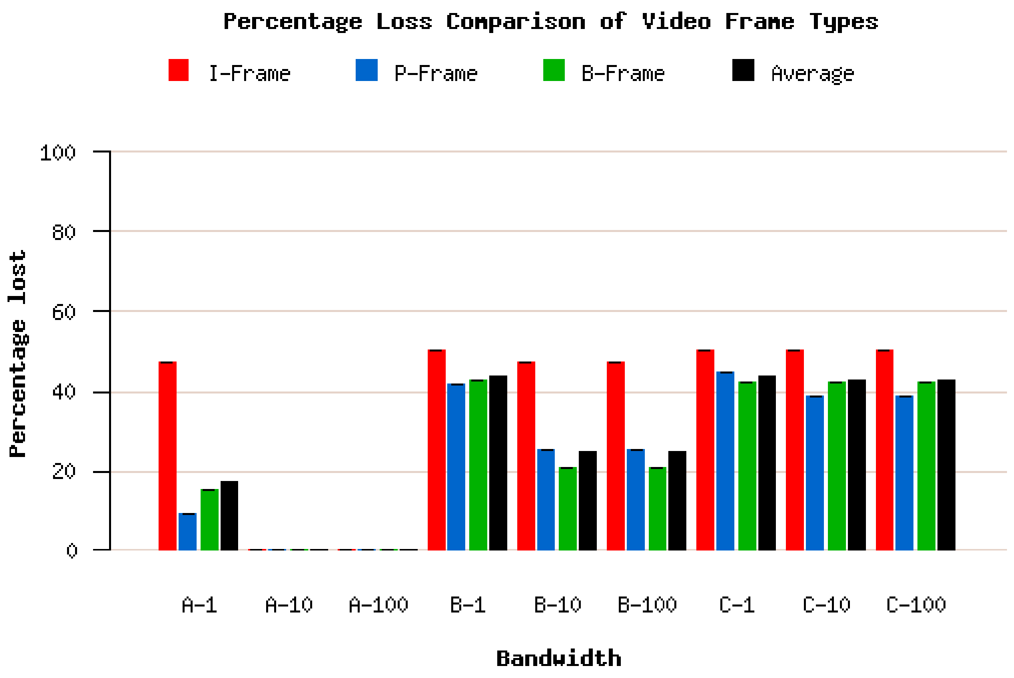


Figure 4.66: Frame Loss for Hall QCIF Video from Large Network scenario

4.7 Overview

This is a summary of the majority of the overall testing done so far. Figure 4.67 depicts a comparative matrix which shows how all these various aspects of the system are evaluated. The discussion with respect to the various components will be presented in the next chapter, nevertheless this figure presents an overview of the results. The key colour coordination provides a measurement scale which allows the different aspects of the matrix to be distinguishable with respect to excellent values down to bad values. Table 4.5 presents key ratings for the different components assessed within the Figure 4.67 (these ratings are subjective for the majority of tests except for the PSNR values which are linked to the MOS ratings from [99]).

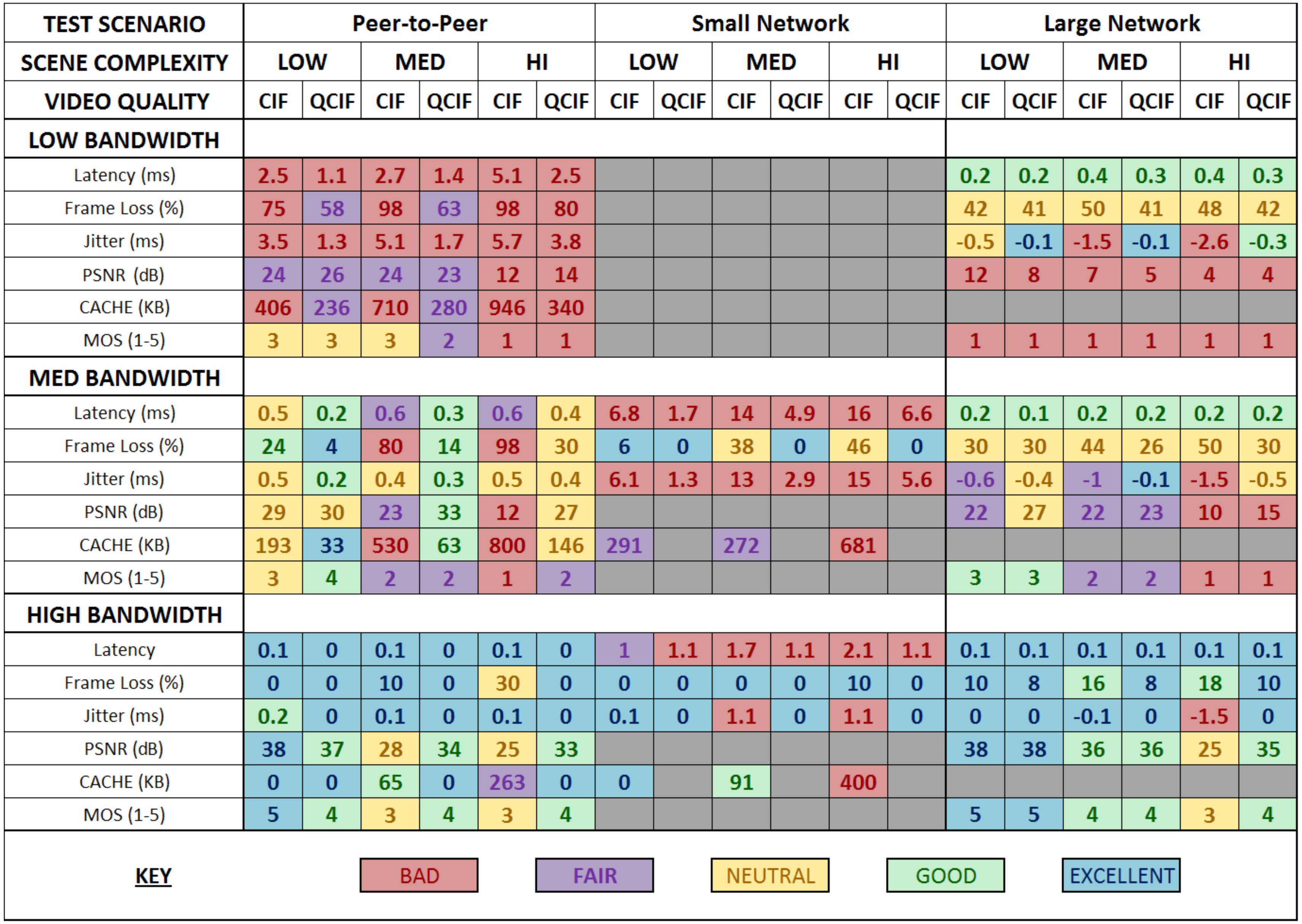
KEY RATING	PSNR	MOS	CACHE	LATENCY/JITTER	% Loss
Excellent	> 37	5	< 50	$- 0.1 > x < 0.1$	< 10
Good	31 - 37	4	50 - 100	$- 0.3 > x < 0.3$	10 - 20
Neutral	25 - 31	3	100 - 200	$- 0.5 > x < 0.5$	20 - 50
Fair	20 - 25	2	200 - 300	$- 1.0 > x < 1.0$	50 - 70
Bad	< 20	1	> 300	x <- 1.0 or x>1.0	> 70

Table 4.5: Quality Ratings for CAFSS-Net matrix components

The following additional details apply to the construction and presentation of the tests done in Figure 4.67 in relation to the bandwidth speeds. Please note that the both the Peer-to-Peer (P2P) and Small Network (Small) scenarios present *averages* of the bandwidths, whilst the Large Network (Large) scenario focuses on the external network *traffic* present along the 10Mb/s connection (average of 1Mb/s, 10Mb/s and 100Mb/s) :

Specification	P2P	Small	Large
Low Bandwidth	9.6K, 28.8K, 56K	Not Tested	High Traffic
Med Bandwidth	64K, 128K, 256K	56K, 64K, 128K	Med Traffic
Large Bandwidth	384K, 512K, 1000K	256K, 512K, 1024K	Low Traffic

Table 4.6: Result Specification Summary for Figure 4.67



TEST SCENARIO	Peer-to-Peer						Small Network						Large Network					
SCENE COMPLEXITY	LOW		MED		HI		LOW		MED		HI		LOW		MED		HI	
VIDEO QUALITY	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF	CIF	QCIF
LOW BANDWIDTH																		
Latency (ms)	2.5	1.1	2.7	1.4	5.1	2.5							0.2	0.2	0.4	0.3	0.4	0.3
Frame Loss (%)	75	58	98	63	98	80							42	41	50	41	48	42
Jitter (ms)	3.5	1.3	5.1	1.7	5.7	3.8							-0.5	-0.1	-1.5	-0.1	-2.6	-0.3
PSNR (dB)	24	26	24	23	12	14							12	8	7	5	4	4
CACHE (KB)	406	236	710	280	946	340												
MOS (1-5)	3	3	3	2	1	1							1	1	1	1	1	1
MED BANDWIDTH																		
Latency (ms)	0.5	0.2	0.6	0.3	0.6	0.4	6.8	1.7	14	4.9	16	6.6	0.2	0.1	0.2	0.2	0.2	0.2
Frame Loss (%)	24	4	80	14	98	30	6	0	38	0	46	0	30	30	44	26	50	30
Jitter (ms)	0.5	0.2	0.4	0.3	0.5	0.4	6.1	1.3	13	2.9	15	5.6	-0.6	-0.4	-1	-0.1	-1.5	-0.5
PSNR (dB)	29	30	23	33	12	27							22	27	22	23	10	15
CACHE (KB)	193	33	530	63	800	146	291		272		681							
MOS (1-5)	3	4	2	2	1	2							3	3	2	2	1	1
HIGH BANDWIDTH																		
Latency	0.1	0	0.1	0	0.1	0	1	1.1	1.7	1.1	2.1	1.1	0.1	0.1	0.1	0.1	0.1	0.1
Frame Loss (%)	0	0	10	0	30	0	0	0	0	0	10	0	10	8	16	8	18	10
Jitter (ms)	0.2	0	0.1	0	0.1	0	0.1	0	1.1	0	1.1	0	0	0	-0.1	0	-1.5	0
PSNR (dB)	38	37	28	34	25	33							38	38	36	36	25	35
CACHE (KB)	0	0	65	0	263	0	0		91		400							
MOS (1-5)	5	4	3	4	3	4							5	5	4	4	3	4

KEY					
	BAD	FAIR	NEUTRAL	GOOD	EXCELLENT

Figure 4.67: CAFSS-Net Research Data Overview

4.8 Summary

This chapter set out to provide data and results from the various tests and investigations presented through the analysis of CAFSS-NET. This chapter first explained the data collection process in section 4.2, which consisted of presenting a deeper investigation into the TRACE_FILE's and their importance to the tabulation of results (with reference to the packet loss, latency and jitter calculations). This section also presented the various details for the standard network/video statistics and their relation to CAFSS-Net.

The next section 4.3 presented the results of the Peer-to-Peer scenario. It consisted of a simple client-server architecture with the following tests: Latency, Frame loss, Jitter, PSNR vs Reference video, Average PSNR with standard deviation, MOS, Video quality and Cache size. The aim of these tests was to simulate different bandwidths for each of the prescribed videos: Akiyo, Foreman and Hall.

Section 4.4 consisted of the results for the Small Network scenario. This scenario provided three sets of tests which were as follows: The physical network tests (similar to the Peer-to-Peer scenario except through a physical network for the frame loss, latency and jitter tests only), the cache comparison tests (which examined the changes in the minimum cache sizes of two consecutive nodes caused by the changes of their bandwidths) and the placement difference tests (which examined if the location of a node affected its minimum cache size if the links between the nodes were uniform).

The last section 4.5 yielded the results of the Large Network scenario. This scenario consisted of three networks with different degrees of external network traffic. The first one consisted of a small Office network with minimal external traffic. The next network consisted of a Administrative network with medium degree of external traffic. The last network represented the Internet. This scenario was characterised as a prediction since it was not confirmed with the testing of a physical network.

This chapter concluded with an overview of the results obtained, presented in section 4.7. This overview displays a matrix detailing the results of the tests performed. This matrix is accompanied by a colour key table that represented the ratings (subjective) of the results found and another table summarising the bandwidth specifications for the tests (the definitions for low, medium and high bandwidth for each of the scenarios tested).

Chapter 5

Discussion

5.1 Introduction

CAFSS-Net has been formulated and developed into a Common Analysis Framework for Simulated Streaming-video Networks. Chapter 3 presented three distinct test scenarios to evaluate the integration of the various tools and components of CAFSS-Net. The three major components of CAFSS-Net are: the videos, the network simulator (with the stream server and the client(s)) and the evaluation tools (with the video publishing tools). Chapter 4 presented the results applicable to the evaluation tools presented in CAFSS-Net (some of which were used with the NS-2 simulation engine or an existing physical network).

In this chapter, we discuss the results from the tests performed in the previous chapter with reference to CAFSS-Net and its three major components. However, the limitations of CAFSS-Net will be discussed first in section 5.2. These limitations dictate the specifications for both the video and the networking components.

Section 5.3 discusses the results of the tests produced in the previous chapter with reference to the three scenarios. Figure 4.67 from section 4.7, summarises the results of the tests performed during the investigation and will be used as the primary focus for the comparison of these different scenarios. This discussion will critically analyse the results and discuss the interesting observations.

Section 5.4 will evaluate CAFSS-Net as a complete framework with its various components (including the minor components) in relation to the findings drawn from the previous section. This section focuses on the implications for CAFSS-Net as a result of the analysis performed within section 5.3, which does suggest further testing, possible amendments and proposals for future work.

5.2 Limitations

5.2.1 Videos

CAFSS-Net accepts a raw video source file as input and produces a raw video destination file as an output. The following limitations and stipulations for CAFSS-Net have been imposed on the system design to control the scope of the investigation:

5.2.1.1 Encoding Format

All the streaming tests were based on a limited collection of raw YUV videos obtained from an online digital library of research video [114].

The video publishing tool(s) (the Video Sender component of the Evalvid tool-set) sends a compressed form of the YUV stream to alleviate the network load. The Evalvid tool-set has a limited set of compression algorithms which consists of the following compression formats [115]: ffmpeg (MPEG-4), XviD (MPEG-4), x264 (H.264), JM 10.2 (H.264), ffmpeg (H,264).

The Video Publishing tool(s) have been tested with only one of the available alternative compression schemes, namely XviD (MPEG-4) for a single video stream transmission. This encoding format was able to successfully transmit an XviD encoded video through the simulated component of CAFSS-Net. The testing and possible implementation of the other alternative compression schemes into CAFSS-Net can be possible extensions for future work [115].

5.2.1.2 Resolutions

All high quality videos consist of the standardised CIF (Common Interface Format) with a resolution fixed at 352x288. All low quality videos consist of the standardised QCIF (Quarter Common Interface Format) with a resolution fixed at 176x144. CAFSS-Net was coded to utilise these two resolutions only.

However, the addition of other resolutions into CAFSS-Net is not difficult. This involves inserting parameters into the evaluation scripts, and ensuring that the system is aware of the dimensions of the video being transmitted (for reconstruction and evaluation purposes). This would allow CAFSS-Net to utilise any video resolution.

5.2.1.3 Real-time vs Pre-Defined

Video streaming comprises of live or pre-defined videos transmitted to clients through a network. CAFSS-Net has been tested with six pre-defined videos stipulated in subsection 3.3.6. CAFSS-Net will not utilise real-time streaming.

However, real-time streaming can be represented as a live stream transcoded to a file and then the file transmitted as a pre-defined stream. All videos are transmitted using a controlled bit-rate scheme.

5.2.1.4 General Video Specifications

All videos are uniform in frame length with 300 frames. These videos consist of 10 seconds of video footage and the videos operate at 30 frames per second. These limits are for test comparison purposes only. CAFSS-Net can stream videos of any length with the appropriate amendments to the evaluation scripts (adjustments of the graph axes for the new frame measurements).

5.2.2 Network

CAFSS-Net consists of evaluation tools that can interpret both simulated and physical [1] network transmissions. However, there are specifications imposed on the structure of CAFSS-Net that limit the interaction with a network. These details are stipulated below:

5.2.2.1 Bandwidth

CAFSS-Net can evaluate any common network bandwidth standard (simulated and physical). The bandwidths utilised during the investigation of the different test scenarios are limited to the following speeds (in b/s): 9.6K, 28.8K, 56K, 64K, 128K, 256K, 384K, 512K, 1000K (1024K in physical tests), 10M, 100M.

5.2.2.2 Network Protocols

CAFSS-Net utilises CBR packet based video transmissions because the tools can provide greater accuracy with a controlled bit-rate. CAFSS-Net does not implement video streaming over TCP connections. The limits of the protocol are defined by the network simulation engine and the evaluation tool-set

5.2.2.3 Wireless

NS-2 does provide support for wireless network simulation which can simulate signal range, node movements (mobile node(s) communicating with different wireless antennae) and wireless positioning. This representation however presents a different TRACE_FILE to the traditional wired network. Hence the following specifications have been implemented for wireless connections:

- CAFSS-Net assumes that wireless network nodes exhibit the same characteristics as wired network nodes except they may introduce more packet loss, delay and possible jitter.

[1] with the aid of the third party tool TCPDUMP [112]

- CAFSS-Net is not configured to interpret the NS-2 wireless TRACE_FILE format, but this can be a possible extension for future work.

5.3 Analysis of the Results

A primary quality assessment for video stream outputs are the actual videos themselves. However, the videos do not always provide an accurate measurement for the the video stream transmission. For this reason, CAFSS-Net encompasses various evaluation tools to provide different quality assessments for a given video stream transmission.

The previous chapter presented three test scenarios along with the results of the various quality assessment tests which were the Peer-to-Peer scenario, the Small Network scenario and the Large Network scenario. This section aims to discuss the results of these three test scenarios from an analytical point of view, highlighting the expected outcomes and investigating the interesting observations. This analysis will be based on Figure 4.67 which presents a summary of the results for all the tests presented in the previous chapter.

5.3.1 Bandwidth Discussion

The subsections 5.3.1.1, 5.3.1.2, 5.3.1.3 discuss the overall results from Figure 4.67 in relation to the three categories of bandwidths: low, medium and high. Each of the these subsections will highlight the interesting observations encountered throughout the three test scenarios.

5.3.1.1 Low Bandwidth Discussion

The Peer-to-Peer scenario presented video quality outputs according to different bandwidths for the Akiyo CIF, Foreman CIF and Hall CIF videos in section 4.3.7. A subjective assessment concluded that the Akiyo CIF video produced better overall video quality outputs than the other videos, with the Foreman CIF video producing the lowest quality results. These results are caused by the difference in scene complexity between the three videos, with the Akiyo CIF video containing the lowest scene complexity and the Foreman CIF containing the largest scene complexity. Within this scenario, this deduction has been expressed throughout the different tests performed. They have been interesting observations that further highlight the differences between these three videos.

However, an interesting observation is made within the context of the Large network scenario. The high extra network traffic decreased the MOS for all the videos, showing that even though the latency results may be *good* and even with the presence of *excellent* jitter results, the video may still not produce acceptable quality. This scenario is very interesting as it shows the connected architecture of the various elements of streaming video, and how evaluation tests can not always be interpreted independently.

5.3.1.2 Medium Bandwidth Discussion

The medium bandwidth test section within Figure 4.67 illustrates a significant increase in the ratings for the majority of the test results compared to the low bandwidth test results. These medium bandwidth tests for the Peer-to-Peer scenario present a greater contrast between the three videos than the low bandwidth tests. We can immediately see the Akiyo CIF video producing *neutral* and *good* results for the tests. When the resolution is decreased to a lower size (QCIF), the quality increases substantially, with the average frame loss dropping from 24% down to 4% (an *excellent* rating).

However, the increase in bandwidth was not sufficient for the Hall CIF and Foreman CIF videos, which still maintained *bad* and *fair* results from the Peer-to-Peer scenario. A better example of these poor scores (for the Hall CIF and Foreman CIF videos) are both the frame loss percentages (from section 4.3.2) and the minimum cache sizes (from section 4.3.8). They both demonstrate a significant weakness with the medium bandwidth scores in comparison to the Akiyo CIF videos. The Foreman CIF video produces even lower scores (higher percentages and cache sizes) than the Hall CIF video. This same ratio of quality between the videos was present throughout the low bandwidth tests discussed from the previous subsection 5.3.1.1.

The results in Figure 4.67 summarise the physical network tests from the Small Network scenario and show an interesting observation that provides different results to the Peer-to-Peer scenario. Even though no PSNR and MOS tests were performed within this scenario, the latency, jitter and frame loss tests show an interesting difference amongst themselves. The simulated latency and jitter results (from the Peer-to-Peer scenario) ranged from *fair* to *good* whilst the physical network results (from the Small network scenario) produced *bad* results across the board. On the other hand, the physical frame loss results ranged from *neutral* to *excellent* whilst the simulated results ranged from *bad* to *excellent*. These anomalies pose a significant difference, but these anomalies will be explained in section 5.4.

5.3.1.3 High Bandwidth Discussion

This section of results produced a significant difference between the previous bandwidth sections as illustrated in Figure 4.67. The majority of the results presented *excellent* ratings with some instances of *good* results and few lesser ratings. This is a clear observation showing that high bandwidth connections produce better quality video streams than the lower to medium bandwidth streams. High quality video streams are possible even in the presence of external network traffic (as shown in the Large Network scenario results illustrated in Figure 4.67).

The physical network tests from the Small Network scenario, present *fair* and *bad* latency results in Figure 4.67, with two instances of *bad* jitter results (the Hall CIF and Foreman CIF videos from test section 4.4.3). However, even with the presence of these *bad* latency and jitter (for the limited instances) results, there was negligible frame loss experienced. This was a result of the high bandwidth and the physical network test setup which consisted of the server and the client being further away from each other. This relationship between these tests will be further examined in section 5.4.

These high bandwidth results show an overall similarity in the ratings between all the three test scenarios. The slight differences between these results are definitely indicative of the different video scene complexities. Nevertheless, the overall similarity suggests that the simulated network test results are within the same range as the physical network results for the majority of these tests. This is illustrated by the excellent *ratings* achieved for the majority of the frame loss and jitter tests, with the PSNR and MOS tests all scoring *neutral* to *excellent* ratings. Aside from the *bad* latencies obtained from the Small Network scenario, it is clear that CAFSS-Net is able to produce simulated test scores close to those of a physical network. However, further testing is suggested in section 5.4 to validate the accuracy of this relationship.

5.3.2 Measurement Discussion

It is clear that from the previous bandwidth discussions, there is an overall relationship between the bandwidth connection and the result of the measurement test. However, further analysis suggests that they are certain relationships between the measurement tests themselves. These relationships between the tools can be catagorised into three sections: independent tools, dependent tools and a combination of tools.

5.3.2.1 Independent Evaluation Tools

The independent evaluation tools are tools that can provide an accurate measurement about the end result of a video stream independently from other evaluation tools. From Figure 4.67, examples of independent tools are the MOS and the PSNR. The PSNR tool calculates the Peak Signal to Noise Ratio experienced by the video transmission. The MOS assigns a value 1-5 as a quantitative result for the quality output of the video. The frame loss is another independent measure which provides the percentage of lost frames and can also provide an effective indication to the final quality output of a video stream. These tools can assess whether the result of the video stream quality was *good* or *bad* independently from other tools (if the MOS was high, the video stream quality was *good/excellent* or if the MOS was low, the video stream quality was *fair/bad*).

5.3.2.2 Dependent Evaluation Tools

The dependent evaluation tools are tools that can only provide an accurate measurement to the final outcome of a video with assistance of other evaluation tools (preferably independent tools). The latency and jitter tools are examples of dependent evaluation tools. As illustrated in Figure 4.67, the medium bandwidth tests from the Small Network scenario show *bad* results for both the jitter and latency tests. However, the frame loss (an independent evaluation tool) scores show *excellent* results for the majority of the tests (with *neutral* results for the rest). This demonstrates that though the latency and jitter tests are excellent measurement tools for analysing the transmission of a video stream, their results have to be interpreted with other evaluation tools (preferably independent tools) and also within the context of the whole video stream transmission (the video resolution, scene complexity, the bandwidth).

The minimum cache size tool is an interesting exception. This tool determines its value by comparing the MOS of the source and destination videos and applying a cache size to compensate if there is a difference (a full description has already been presented in section 4.3.8). Though it can provide information about the final quality of the video transmission (the bigger the cache size, the more likely the final video quality will be poor without it), it also has to be viewed within the context of the whole transmission.

An example is taken from section 4.3.8 (with reference to the Foreman CIF video Figure 4.23) where the aggregated high bandwidths require a cache size of 263 KB which is a *fair* rating (as illustrated in Figure 4.67) whilst the majority of the other quality tests (latency, jitter, frame loss, MOS and PSNR) score *neutral* to *excellent* ratings for the same video as illustrated in Figure 4.67. It is for this reason that we classify this tool as dependent.

5.3.2.3 Combination of Tools

The following are evaluation tools that require the results of other evaluation tools in order to function:

MOS	A 1-5 rating that is a measure of the average PSNR value
Jitter	The variance in the latency (end to end delay)
Cache	A calculation based on the difference between MOS values

The introduction of extra network traffic in the Large network scenario and the physical network tests from the Small Network scenario, yielded some unique combinations of results. These two scenarios presented results that can show how different external factors (distance between the server and client, foreign traffic, video resolution) affect the video stream. They also reveal interesting test combinations that can provide significant information about the video stream transmission.

Table 5.1 shows examples of how the *positive* and *negative* combination of test results from different tools can be used to evaluate the video stream. It consists of the *positive* test results (*neutral, good* and *excellent* ratings) for tools and the *negative* test results (*fair* and *bad* ratings) for tools followed by what the evaluation of the video stream would be. An example is provided by the second row showing that when the frame loss tests yielded *positive* results, combined with the *negative* results for the latency and jitter tests, the *evaluation of the video stream* was that the distance between the source and destination was large and a medium bandwidth connection was utilised (as illustrated in Figure 4.67 for the medium bandwidth test for the Small Network scenario).

Positive test Results	Negative test Results	Evaluation of the video stream
Latency & **Jitter**	**PSNR** & **MOS**	Extra Network traffic
Frame Loss	**Latency** & **Jitter**	Large distance between source and destination over a med bandwidth connection
Frame Loss & **Jitter**	**Latency**	Large distance between source and destination over a high bandwidth connection
-	**MOS** & (**Jitter** $<$ -1.0 *ms)*	Extra Network Traffic

Table 5.1: Combinations of test results with evaluation of the video stream

5.4 Implications for CAFSS-Net

CAFSS-Net thus far has produced a series of video stream transmissions which have been evaluated using its evaluation tool-set to determine the validity of the framework as a whole. In light of the analysis presented from the previous section 5.3, both the bandwidth and measurement discussions yielded some interesting observations (and in some cases anomalies) which were discussed, but needed further investigation beyond the tool-set itself. This section evaluates CAFSS-Net based on the findings of the previous section. The focus will be on the interesting observations identified, and the anomalies they present with reference to the components of CAFSS-Net as illustrated in Figure 3.6.

5.4.1 Video Component Discussion

The results from both the previous chapter and the discussion presented in section 5.3, show that CAFSS-Net is able to successfully accept raw YUV videos, transmit them through its network simulation tool and reproduce the reconstructed video. The Akiyo, Hall and Foreman videos with their different scene complexities were tested by all the evaluation tools within CAFSS-Net. CAFSS-Net was able to successfully show the differences between the scene complexities through the test results as these differences were reflected within results in the previous chapter. CAFSS-Net was also able to transmit these videos through different bandwidth tests and reproduce the transmitted streams as was shown by the results from section 4.3.7.

The various limitations imposed on the design of CAFSS-Net (from section 5.2) show that CAFSS-Net still requires some additional modifications to incorporate other video resolutions. The existing resolutions QCIF and CIF were the only two resolutions tested within this study. However, through amendments to the various evaluation scripts, CAFSS-Net is capable of accepting other resolutions. These amendments are possible extensions for future work.

5.4.2 Video Publishing Discussion

CAFSS-Net utilises video compression and decompression tools to minimise the bandwidth utilisation during the video stream transmission as is done in practise when streaming video. Part of the limitations previously mentioned in section 5.2 state that CAFSS-Net utilised the MPEG-4 encoding and decoding tool from the Evalvid tool-set. CAFSS-Net was able to successfully transmit a video stream with an alternative XviD encoding tool, but no evaluation was done on this video stream transmission as the existing evaluation tools would not accept this video encoding format. The Evalvid tool-set provides other alternative compression schemes which can be incorporated into CAFSS-Net as future work. However, an interesting extension to CAFSS-Net would be an evaluation between these alternatives. Examples of evaluation tests between these alternatives could be to investigate which encoding format is better suited for low bandwidths or is more susceptible to noise.

5.4.3 Streaming Server Discussion

The streaming server resides within the network simulation component of CAFSS-Net. However, the Small Network scenario presented physical network tests that involved capturing the network trace data from physical machines. In this instance, the streaming server was external to the network simulation component of CAFSS-Net. From the results obtained in Chapter 4, it is evident that the streaming server is able to distribute the video stream through a simulated network and through a physical network.

5.4.4 Client(s) Discussion

The clients also reside within the network simulation component for CAFSS-Net. However, like the streaming server, the clients resided outside of the network simulation component of CAFSS-Net for the Small Network scenario (for the physical network tests). In both the physical and simulated networking environments, the majority of the network test data was collected from these clients. It is also clear from the results of the previous chapter, that the clients were able to receive the video stream transmissions. More importantly, the clients could successfully collect the trace data needed to evaluate the video stream transmission with the relevant evaluation tools.

5.4.5 Network Simulator Discussion

The network simulator is an integral part of CAFSS-Net as it is responsible for simulating a physical network which will be be analysed. NS-2 is capable of representing different networking conditions which include network topologies and bandwidths. However, the analysis obtained from the bandwidth discussions in section 5.3, highlighted some interesting observations with regards to the results from the previous chapter.

A significant difference was noticeable for the medium bandwidth tests between the Small Network scenario and the Peer-to-Peer scenario. As observed in section 5.3.1.2 from the medium bandwidth discussion, some irregularities exist between the physical and simulated network results for the medium bandwidth tests. The latency and jitter irregularities can be accounted for by the significant distance between the server and the client for the physical network tests. NS-2 only applies accurate distance measurements for wireless networks (as distance is a significant factor between nodes in a wireless network) but the TRACE_FILE's for wireless networks in NS-2 are presented differently to the normal networks.

However, the differences in frame loss can be as a result of various factors. One factor may be the differences in representing network bandwidths for both the simulated and physical tests. NS-2 has parameters and flags that can be used to specify the configuration of the network being simulated such as: queue size, scheduling, internal buffers and various other settings (from the section 3.4). Different combinations of these parameters and flags can affect the outcome of the simulation.

Another factor could be the physical network bandwidth representation. Since the physical network tests utilised a software-based bandwidth limiter [113] to reproduce the medium bandwidth specifications for the tests, this could have also created discrepancies in the results. The software-based bandwidth limiter only provided the following parameters to control the bandwidth connection: the protocol (TCP,UDP,IP and ICMP), the bandwidth limit and a priority level. However, it is interesting to note that with the high bandwidth tests, this discrepancy was not present. The investigation of both these factors can be possible extensions for future work.

5.4.6 Evaluation Tool-set Discussion

CAFSS-Net's evaluation tools can add further analysis to the quality evaluation of the video stream transmission. The measurement discussion in section 5.3.2 discussed both the different relationships between the different testing tools and their dependency/independence on/from the other tools. However, Figure 4.67 illustrates significant grey areas that indicate tests that were not performed. In order to enhance the accuracy of CAFSS-Net, the following future tests have been proposed, accompanied by their possible benefits for CAFSS-Net.

The Large Network tests did not include any minimum cache size tests. Future implementation of these tests can help associate a relationship between the minimum cache size and the bandwidth connection, especially in the presence of external traffic which this scenario provides.

The Small Network scenario did not present the MOS and PSNR results for both the high and medium bandwidth tests. Producing these results could possibly enhance the relationship between the frame loss scores and both the MOS and PSNR results for both the medium and low bandwidth tests.

Since no low bandwidth tests were done at all for the Small Network scenario (the physical network tests for low bandwidths), future implementation of these tests can associate the physical network results closer to the simulated results (the Peer-to-Peer tests and the Small Network tests). This can also provide more data to aid in the development of

CAFSS-Net, particularly with the representation of both the simulated and the physical network results.

5.5 Summary

This chapter began by expressing the limitations that governed the network and video components of CAFSS-Net. The video limitations consisted of the MPEG-4 encoding format, both the CIF and QCIF resolutions, pre-defined videos and the general structure of the videos. The network limitations focused on the bandwidths tested (both the simulated and the physical tests), how CBR is the means of transporting the stream and how the wireless network representations will be represented within CAFSS-Net. This section also proposed extensions to these limits as possible future work.

Section 5.3 provided the analysis of the results by discussing the different bandwidth tests in relation to the overview of the results presented in section 4.7. The low bandwidth discussion (in subsection 5.3.1.1) focused on the comparison between the Large Network and Peer-to-Peer network scenarios, whilst the medium and high bandwidth discussions (section 5.3.1.2 and 5.3.1.3 respectively) analysed all three scenarios. Interesting observations between the scenarios and within the scenarios were identified and further analysed in section 5.4.

The analysis section also presented the measurement discussion (subsection 5.3.2), where different evaluation tools were analysed in relation to their results. This discussion identified relationships between the tools themselves which yielded the following three analytical groups of tools:

1. Independent evaluation tools (PSNR, frame loss, MOS) which can independently provide an accurate measurement on the final video quality output (determine if the quality is good or bad)

2. Dependent evaluation tools (jitter, latency, cache) that need to be interpreted with other evaluation tools in order to provide an accurate measurement on the final video quality output

3. Combination of tools which need the results of another tool in order to determine their own results (the MOS test requires results from the PSNR test, the cache test requires results from the MOS test and the jitter test requires results from the latency test).

The last section 5.4 presented the implications for CAFSS-Net as a consequence of analysing the various results. This section discussed the different components of CAFSS-Net, evaluating their effectiveness in their functionality/performance. The client, streaming server and video components yielded successful evaluations whilst the video publishing component suggested that further tests be implemented using alternative compression schemes as future work.

The network simulator component discussed the irregularities between the physical and simulated results, and suggested that further testing as future work be recommended to resolve these. The evaluation tool-set discussion investigated the various tools and highlighted the relationships between them as discussed in section 5.3.2. This section presented more suggestions for possible future work to further enhance the accuracy of these evaluation tools for CAFSS-Net.

Chapter 6

Conclusions

6.1 Introduction

The literature for this study within the streaming media domain was explored and consisted of the following categories: streaming, network simulation and videos. Three existing streaming media distribution engines (systems/services) were investigated to determine the common characteristics between them to provide a foundation for CAFSS-Net. CAFSS-Net was soon comprised of the following three major components (with a total of six components): Network Simulator (with the Streaming Server and Client(s) components), the Evaluation Tool-set (with the Video Publishing Tool(s) component) and the Videos. CAFSS-Net was evaluated using three test scenarios which were: Peer-to-Peer, Small Network and Large Network. From these test scenarios, results were gathered and a subsequent discussion on the findings was presented.

This chapter summarises this research study by first presenting the objectives in section 6.2. Section 6.3 provides a brief summary that shows how CAFSS-Net is able to emulate the essential characteristics of the three existing streaming media distribution engines explored (Microsoft Windows Media Services 9, Apple's Darwin Streaming Server, and RealNetworks Helix Server). Section 6.4 presents the future work CAFSS-Net which can be implemented without significant changes to the overall framework. This is followed by the possible extensions for CAFSS-Net identified previously in Section 6.5 which require significant amendments to the overall framework. The next section 6.6 provides possible applications for CAFSS-Net and the last section (6.7) presents the conclusion of this study.

6.2 Objectives

Video streaming requires three components: The encoder which reproduces the video to be streamed, the server which transmits the video stream and the player which plays the video stream [10]. However, another important factor that governs the video stream is the network that distributes the stream. The objective was to develop a streaming media analysis framework which assists in the prediction of modifications in network design and how the different video characteristics affect the video stream.

This analysis framework examined existing commercial and open-source streaming media distribution services/systems and determined the similar features between them as a foundation for the framework. From this foundation a complete framework was defined with its relevant components and a test bed proposed to validate the framework.

The aim of the test bed was to examine the major complexities of video streaming which are lost/dropped packets, latency, jitter, noise and video specifications (such as resolution and scene complexity). From the test results collected, an analysis of these results was performed to evaluate the framework. From this evaluation, future proposal and extensions were noted and presented.

6.3 CAFSS-Net and Existing Systems

In section 2.6, existing streaming media engines (Helix [60], WMS-9 [16] and Darwin [59]) were explored to define a common foundation for the formulation of CAFSS-Net. CAFSS-Net has been defined and tested through a test bed in section 3.4 with the results provided in sections 4.3, 4.4, 4.5 and analysed in Chapter 5. This section aims to show how CAFSS-Net is able to emulate the essential characteristics of the existing commercial streaming systems in relation to the *features* presented in Table 6.1. This table shows the features that represent minor components/attributes for streaming media.

Features	**CAFSS-Net**	**Helix**	**WMS-9**	**Darwin**
Version	1.0	11.1.1	9.0	5.5.4
Available OS	Win XP with Cygwin	UNIX Windows 2K3	Win Server 2K8 Win Server 2K3	MAC-OS X Win NT
License	NA	30 Day Trial	App Service	Open Source
Unicast	Yes	Yes	Yes	Yes
Multicast	Yes	Yes	No (ver 9.0)	Yes
On-Demand	Yes	Yes	Yes	Yes
Cache	Yes	Yes	Yes	No
Proxy	Yes	Yes	No	No
Interface	Console	Web-Based	Server 2003	Web-Based
Extensions	TCPDUMP	Helix Proxy	WME-9	Broadcaster

Table 6.1: CAFSS-Net and the Summarised Features of Streaming Media Systems

6.4 Future Work

This section consists of the future work that can be implemented into CAFSS-Net without significant changes to the overall framework. The majority of these suggestions have been expressed in the previous chapter, but this section aims to summarise them. These future work suggestions are related to the different components of CAFSS-Net as analysed from the section 5.4.

Video Section 5.4.1 proposes that other video resolutions be implemented into CAFSS-Net. This would allow for more video streams with possible differences in scene complexity as not all videos conform to these two standards (CIF and QCIF).

Video Publishing Section 5.4.2 proposes that other compression schemes be implemented into CAFSS-Net. Different publishing tools may be better suited to different streaming conditions like external network traffic, noisy networks and videos with high scene complexities.

Network Simulator Section 5.4.5 proposes that the various parameters and flags for NS-2 be explored to better correlate the simulated network results to the physical network results. Another related suggestion was that the physical network tests be done using physical network connections (56K modem, ISDN line, ADSL connection and a T1 line) as opposed to using a software-based bandwidth limiter.

Evaluation Tool-set Section 5.4.6 proposes that the following tests be performed to provide a better analysis between the evaluation tool-set, the network simulation tests and the physical network tests:

- All the low bandwidth tests for the Small Network scenario
- The cache tests for the Large Network scenario
- Both the PSNR and MOS tests be for the Small Network scenario.

6.5 Possible Extensions

This section highlights some of the possible extensions to CAFSS-Net in order enhance its functionality.

CAFSS-Net is console based. However, existing streaming media distribution engines provide a Graphical User Interface (GUI) that allows a visual representation of the system. A possible extension for CAFSS-Net can be the development of a GUI that is able to provide a visual representation of the system. Since the majority of the framework is integrated through different scripts, developing a user interface as a front-end could better illustrate the behaviour/characteristics of the networks being investigated. Figure 6.1 is a possible example of a GUI for CAFSS-Net, illustrating the *input* and *output* video sources, with the network simulator below.

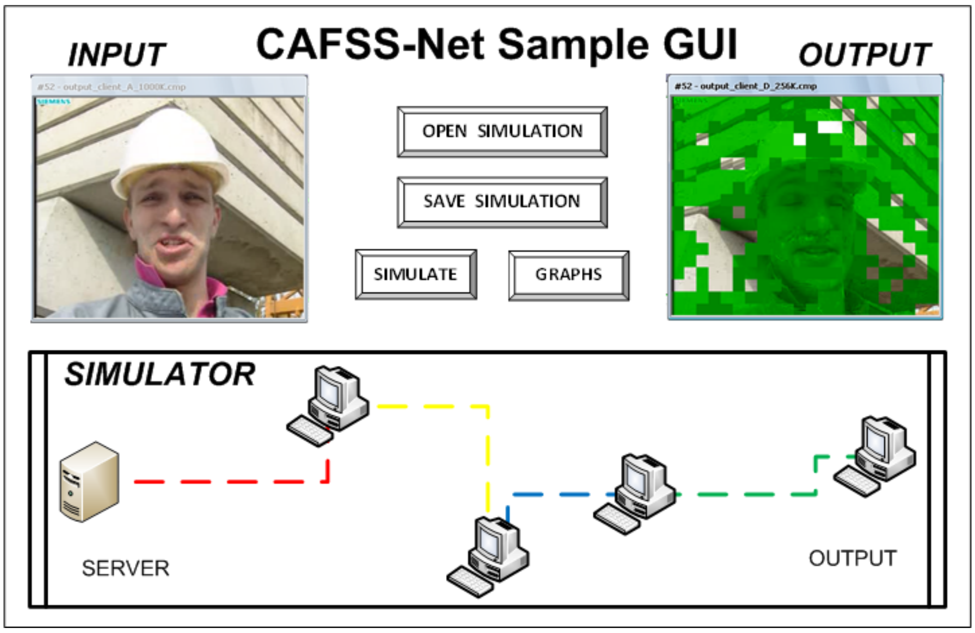


Figure 6.1: Sample GUI extension for CAFSS-Net

The video publishing component of CAFSS-Net currently utilises only the MPEG-4 compression scheme to compress and reproduce raw YUV videos. The Evalvid tool-set (CAFSS-Net's evaluation and video publishing components) provides alternative compression schemes and a suggestion to incorporate these into CAFSS-Net has already been defined as future work in the previous section. However, a possible extension to CAFSS-Net can be the evaluation between these alternative compression schemes. A possible evaluation could be aimed at determining the best compression scheme for different conditions such as: which scheme is better suited for low bandwidths, which scheme is suited for noisy networks and which scheme provides the best overall quality.

A unique extension to CAFSS-Net is the possible incorporation of a high-level caching algorithm. Section 2.5 reviewed different caching algorithms and strategies and incorporating one of these algorithms (or many other possibilities) into CAFSS-Net could provide interesting benefits in aiding in the design future networks. Some of these algorithms and strategies have already been noted (in Chapter 2) as possible extensions which are: adapting by priority drop (reviewed from [35, 36]), predictive buffering algorithm (reviewed from [7]) and the frame storage strategy for a proxy service (reviewed from [50]).

6.6 Possible Applications

Distributed media streaming requires hardware in order to successfully transmit and distribute a video stream. However, physically deploying an entire architecture without any tests may prove to be expensive in the event that the requirements are not met. CAFSS-Net is an excellent solution to aid in predicting the performance impacts of a streaming media architectures.

Large companies can use CAFSS-Net as a simulated deployment test-bed tool to further assess and analyse the quality of a video stream under varying conditions such as: video-conferencing environments, streaming between different company buildings (where noise and distance may affect the quality).

Another suitable application for CAFSS-Net is for educational purposes. CAFSS-Net can demonstrate the streaming of video through a simulated network graphically (using the NS-2 Network Animation tool), and associating both the different networking conditions and video characteristics to the final video quality. CAFSS-Net as an educational tool presents three major components working concurrently, video distribution, network simulation and quality evaluation tests.

6.7 Conclusion

Helix, WMS-9 and Darwin all provide pre-defined or real-time streaming services to networks. However, determining an appropriate quality measurement for the video distribution can only be done by streaming the video through the network. These systems/services provide short guidelines for recommended streaming configurations, but they are limited to the bandwidth connection between the source and the destination. The size of the network and different links between the source and destination are factors that are generally not available in providing a guideline.

This study analysed these existing commercial and open-source streaming media distribution services/systems and summarised the similar core features between them. From these core features, a complete Common Analysis Framework for Simulated Streaming-video Networks was defined with its six components (three major components) which were: the videos, the video publishing tools, the streaming server, the network simulator, the client(s) and the evaluation tool-set.

A test bed was developed which consisted of the three test scenarios designed to test different aspects of the framework. The Peer-to-Peer scenario was a simple client-server architecture designed to test different bandwidths through a simulated network. The Small Network scenario consisted of three tests: physical network tests, a cache comparison test and the difference in node placement test. The Large Network scenario was designed to examine the impact of external network traffic on specific nodes.

The results of these tests were collected and an analysis was presented on the interesting observations and on the irregularities. CAFSS-Net along with its components was evaluated on the analysis performed, and although various extensions and possible future tests were recommended for some of the components (the possible extensions are presented in section 6.5), CAFSS-Net was able to achieve the following:

1. Accept raw YUV videos
2. Reproduce the videos for streaming
3. Have the streaming server transmit the video through a network to client(s)
4. The network can be both simulated or physical
5. Reproduce the videos from the client(s) back to raw YUV format
6. Evaluate both the transmission and the final video output using the following metrics:

   Latency The end to end delay between frames

   Jitter The variation of latency

   PSNR The Peak Signal to Noise Ratio of the video stream

   Frame Loss The percentage of lost frames

   MOS The Mean Opinion Score rating the final video quality from 1 to 5

CAFSS-Net's other achievements consist of the integration of all its six components. The client, streaming server and video components accomplished their objectives both within the physical and simulated environments.

Although the video publishing component was able to provide the necessary compression and decompression, proposals were suggested for both the addition of alternative compression schemes (as future work in section 6.4) and possible evaluation of these alternative schemes (as an extension in section 6.5).

Both the evaluation and network simulation components accomplished their goals, but it was noted that further physical network tests be performed as recommended in section 6.4. This was a result of some discrepancies between both the simulated and physical network results for the medium bandwidth tests.

Aside from these achievements and discrepancies, CAFSS-Net was able to successfully show how the different characteristics of video (such as the motion within the scene, camera position, video resolution) affect the outcome of a distributed video stream. CAFSS-Net was also able to show how the different networking conditions (such as topology, the scale of the network, external network traffic) also determine the outcome of a distributed video stream. This study identified relationships showing different dependencies between the evaluation tools and how the combination of results of these different tools can accurately aid in the evaluation of a video stream.

CAFSS-Net's main objective is to aid in the development and deployment of streaming media networks. It alleviates the need to physically construct the distributed streaming architecture in order to evaluate the quality of the video stream output. However, since simulated networks are only abstractions of physical networks, there is a limit to the accuracy of the representation. There was a significant difference between the physical network results and the simulated network results for medium bandwidth connections, but the high bandwidth connections presented similar results.

However, the findings confirm that CAFSS-Net is able to predict the results of a video stream through different sized networks and different network configurations for physical networks by simulating the same networking conditions through its simulated networking environment. Unfortunately, CAFSS-Net's accuracy for these predictions is limited to high bandwidth connections, but proposals for further testing have been presented in section 6.4 to improve this accuracy.

In conclusion, CAFSS-Net is a suitable Common Analysis Framework for Simulated Streaming-Video Networks which can aid in the design and deployment of streaming media distribution systems/services.

Bibliography

[1] W. Kellerer, E. Steinbach, P. Eisert, and B. Girod, "A real-time internet streaming media testbed," Department of Electrical Engineering, Stanford University, May 2002.

[2] Accordent Technologies, "Enterprise webcasting and the rise of training 2.0: Examining the impact of online rich media and media management on the knowledge economy," *Streaming Media Inc: Innovation Series*, vol. Online Learning: Textbook Strategies for Video Education, no. 6, pp. 9–12, August 2007. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[3] M. Gordon, "A powerful trend that represents fundamental change: The internet streaming media boom," *Streaming Media Inc: Innovation Series*, vol. Best Practices: Media, Broadcacst & Entertainment, no. 5, p. 30, July 2006. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[4] P. de Cuetos and K. W. Ross, "Unified framework for optimal video streaming," in *INFOCOM 2004. 23rd Annual-Joint Conference of the IEEE Computer and Communications Societies*, vol. 3. Inst. EURECOM, Sophia Antipolis;, March 2004, pp. 1479–1489.

[5] H. M. Abdel-Wahab, "Working with time-based media," Computer Science - Old Diminion University, March 2004. [Online]. Available: http://www.cs.odu.edu/~cs778/spring04/lectures/lect1Introduction.html

[6] S. Jin, A. Bestavros, and A. Iyengar, "Accelerating internet streaming media delivery using network-aware partial caching," in *Proceedings of the 22 nd International*

Conference on Distributed Computing Systems (ICDCS). Washington, DC, USA: IEEE Computer Society, 2002, p. 153.

[7] P. Y. Ho and J. Y. B. Lee, "Predictive buffering for multi-source video streaming over the internet," in *Global Telecommunications Conference.* San Francisco, CA, USA: The Chinese University of Hong Kong, November 2006, pp. 1–6.

[8] P. Mulumba, P. Clayton, and G. Wells, "An example of a simple cache system for a video streaming implementation within a network simulation," in *South African Telecommunication Networks and Applications Conference (SATNAC)*, D. Browne and M. Nkomo, Eds. Rhodes University: Rhodes University, May 2008, p. 10, submitted: Still Under Review.

[9] Adobe, "A streaming media primer," Adobe Dynamic Media Group, 2000. [Online]. Available: http://www.adobe.com/products/aftereffects/pdfs/AdobeStr.pdf

[10] S. Mack, *Streaming Media Bible.* Hungry Minds, Inc., 2002. [Online]. Available: http://www.streamingmediabible.com/html/toc.html

[11] E. Schumacher-Rasmussen, "Wecasting comes of age," *Streaming Media Inc: Innovation Series*, vol. Webcast Essentials: Presenting Successful Events Online, no. 4, p. 2, December 2005. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[12] M. Cherniack, H. Balakrishnan, M. Balazinska, D. Carney, U. Cetintemel, Y. Xing, and S. Zdonik, "Scalable distributed stream processing," in *CIDR 2003 - First Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, January 2003.

[13] P. de Cuetos, M. Reisslein, and K. Ross, "Evaluating the streaming of fgs-encoded video with rate-distortion traces," Institut Eurécom, Tech. Rep. RR-03-078, June 2003. [Online]. Available: citeseer.ist.psu.edu/decuetos03evaluating.html

[14] Real Media, *Media Creation - RealProducer (RealSystem)*, RealNetworks, Inc., Seattle, WA, USA, September 2008. [Online]. Available: http://www.realnetworks.com/products/producer/index.html

[15] Apple, *Streaming Server Administrator's Guide*, streaming server administrator's guide ed., Apple Computer Inc., Apple Computer, Inc., March 2005.

[16] Microsoft, "Windows media services," 2007. [Online]. Available: http://www.microsoft.com/windows/windowsmedia/forpros/server/server.aspx

[17] Wikipedia, "Cache," Wikipedia, the free encyclopedia, July 2007. [Online]. Available: http://en.wikipedia.org/wiki/Cache

[18] ——, "Latency (engineering)," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Latency_(engineering)

[19] ——, "Gateway (telecommunications)," Wikipedia, the free encyclopedia, September 2007. [Online]. Available: http://en.wikipedia.org/wiki/Gateway_(telecommunications)

[20] TechWeb, "Jitter definition: Techencyclopedia from techweb," TechEncyclopedia, January 2008. [Online]. Available: http://www.techweb.com/encyclopedia/defineterm.jhtml?term=jitter

[21] I. NOSPIN Group, "Guide to network terminology," FREE PC TECH, January 2008. [Online]. Available: http://freepctech.com/pc/002/networks004.shtml)

[22] P. Rob and C. Coronel, *Database Systems: Design, Implementation & Management*, 5th ed., ser. Thomson Learning, J. Locke, D. Kaufmann, and J. Goguen, Eds. Course Technology, 25 Thomson Place, Boston, Massachusetts: Course Technology, December 2002. [Online]. Available: http://www.course.com/catalog/product.cfm?isbn=0-619-06269-X

[23] Wikipedia, "Broadband," Wikipedia, the free encyclopedia, August 2007. [Online]. Available: http://en.wikipedia.org/wiki/Broadband)

[24] ——, "Proxy server," Wikipedia, the free encyclopedia, August 2007. [Online]. Available: http://en.wikipedia.org/wiki/Proxy_server

[25] ——, "Bite rate," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Bit_rate

[26] O. Streaming Media, “Glossary - compression,” Streaming Media.com, February 2008. [Online]. Available: http://www.streamingmedia.com/glossary/term.asp?t=Compression

[27] ——, “Glossary - codec,” Streaming Media.com, February 2008. [Online]. Available: http://www.streamingmedia.com/glossary/term.asp?t=Codec

[28] Wikipedia, “Disply resolution,” Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Video_resolution

[29] O. Streaming Media, “Glossary - buffer,” Streaming Media.com, February 2008. [Online]. Available: http://www.streamingmedia.com/glossary/term.asp?t=Buffer

[30] Wikipedia, “Transcode,” Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Videoconferencing

[31] ——, “Videoconferencing,” Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Videoconferencing

[32] O. Streaming Media, “Glossary - webcasting,” Streaming Media.com, February 2008. [Online]. Available: http://www.streamingmedia.com/glossary/term.asp?t=Webcasting

[33] Y. hua Chu, A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, and J. Zhan, “Early experience with an internet broadcast system based on overlay multicast,” School of Computer Science, Carnegie Mellon University, December 2003. [Online]. Available: citeseer.ist.psu.edu/article/chu04early.html

[34] Kasenna and HP, “An IPTV server benchmark that scales to one million subscribers,” April 2007. [Online]. Available: http://h20219.www2.hp.com/enterprise/downloads/HP_Intel%20_Kasenna_IPTV_Benchmark_Marketing_Brief_040907.pdf

[35] C. Krasic, “A framework for quality-adaptive media streaming: Encode once – stream anywhere,” Ph.D. dissertation, OGI School of Science & Engineering, Oregon Health & Science University, February 2004.

[36] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," Dept of Electrical Engineering-Systems, University of Southern California, 2000.

[37] C. Carmack, "How bittorrent works," HowStuffWorks.com, March 2005. [Online]. Available: http://computer.howstuffworks.com/bittorrent.htm

[38] P. Shapiro, "Bittorrent: Making bulky video files easier to share," Digital Divide Network, http://www.digitaldivide.net/articles/view.php?ArticleID=26, December 2004. [Online]. Available: http://www.digitaldivide.net/articles/view.php?ArticleID=26

[39] B. Spice and A. Watzman, "Carnegie Mellon p2p system could speed movie, music downloads," Carnegie Mellon Press Release, April 2007. [Online]. Available: http://www.cmu.edu/news/archive/2007/April/april10_set.shtml

[40] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava, "On peer-to-peer media streaming," Department of Computer Sciences - Purdue University,West Lafayette, IN, 2002.

[41] V. N. Padmanabham, H. J. Wang, and P. A. Chou, "Resilient peer-to-peer streaming," Microsoft Research, Microsoft Corporation, Redmond, WA, Tech. Rep. MSR-TR-2003-11, March 2003.

[42] P. Shah and J.-F. Paris, "Peer-to-peer multimedia streaming usign bittorrent," Department of Computer Science, University of Houston, April 2007.

[43] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik, "Video coding for streaming media delivery on the internet," in *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY*, vol. 11, no. 3, 281, March 2001, p. 269.

[44] V. K. Goyal, "Multiple description coding: Compression meets the network," *IEEE SIGNAL PROCESSING MAGAZINE*, pp. 74–93, September 2001.

[45] J. G. Apostolopoulos and M. D. Trott, "Path diversity for enhanced media streaming," HP - Mobile and Media Systems Laboratory, HP Laboratories Palo Alto, Tech. Rep., July 2004.

[46] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, “Distributing streaming media content using cooperative networking,” Microsoft Research, Microsoft Corportation, Redmond, WA, Tech. Rep. MSR-TR-2002-37, April 2002.

[47] K.-L. Kao, C.-H. Ke, and C.-K. Shieh, “An advanced simulation tool-set for video transmission performance evaluation,” in *ACM International Conference Proceeding Series*, P. from the 2006 workshop on ns-2: the IP network simulator, Ed., vol. 202, no. 10. ACM New york, USA, 2006. [Online]. Available: http://portal.acm.org/citation.cfm?id=1190464

[48] T. P. Nguyen and A. Zakhor, “Distributed video streaming over the internet,” in *SPIE Conference on Multimedia Computing and Networking (MMCN).* San Jose, California: Berkeley, CA, USA, January 2002. [Online]. Available: citeseer.ist.psu.edu/nguyen02distributed.html

[49] Real Media, *Helix Proxy Configuration and Registry Rerefence*, helix proxy version 11.1 ed., RealNetworks, Inc., Seattle, WA, USA, September 2007.

[50] S. Sen, J. Rexford, and D. F. Towsley, “Proxy prefix caching for multimedia streams,” in *INFOCOM ’99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, New York, USA, March 1999, pp. 1310–1319. [Online]. Available: citeseer.ist.psu.edu/sen99proxy.html

[51] R. Takano, Y. Kodama, T. Kudoh, M. Matsuda, F. Okazaki, and Y. Ishikawa, “Realtime burstiness measurement,” in *Protocols for Fast Lond-Distance Networks (PFLDnet)*, National Institute of Advanced insustrial Science and Technology, University of Tokyo. Nara Japan: AIST, February 2006. [Online]. Available: http://www.gridmpi.org/publications/pfldnet06-takano.pdf

[52] J. M. Almeida, D. L. Eager, M. Ferris, and M. K. Vernon, “Provisioning content distribution networks for streaming media,” University of Wisconsin and University of Saskatchewan, December 2002.

[53] B. Shen, S.-J. Lee, and S. Basu, "Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks," Mobile and Media Systems Laboratory, HP Laboratories, Palo Alto, Tech. Rep. HPL-2003-261, December 2003.

[54] M. Hofmann, T. E. Ng, K. Guo, S. Paul, and H. Zhang, "Caching techniques for streaming multimedia over the internet," Bell Labs Technical Memorandum, April 1999.

[55] G. Houtzager and C. Williamson, "A packet-level simulation study of optimal web proxy cache placement," in *11th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS)*, October 2003, pp. 324–333.

[56] S. Snedaker, D. L. Shinder, and T. W. Shinder, *Best Damn Windows Server 2003 Book Period: Everything You Need To Know About Managing A Windows Server 2003 Enterprise.* Syngress Publishing, Inc. 800 Hingham Street Rockland, MA 02370: Syngress, 2004. [Online]. Available: http://www.streamingmediabible.com/html/toc.html

[57] Microsoft, "Windows media encoder 9," 2007. [Online]. Available: http://www.microsoft.com/windows/windowsmedia/forpros/encoder/default.mspx

[58] Apple, "Quicktime player," 2007. [Online]. Available: http://www.apple.com/quicktime/player/

[59] ——, *QuickTime Streaming Streaming Server and Darwin Streaming Server Administrator's Guide*, Apple Computer Inc., November 2002.

[60] Real Media, *Helix Server Administration Guide*, helix server version 11 ed., RealNetworks, Inc., Seattle, WA, USA, September 2005.

[61] S. Chen and C. Hurley, "Youtube - company history," 2007. [Online]. Available: http://www.youtube.com/t/about

[62] AKAMAI, "Identifying the right media format for broadband video initiatives: A strategic roadmap for success," *Streaming Media Inc: Innovation Series*, vol.

Online Learning: Textbook Strategies for Video Education, no. 6, pp. 21–24, August 2007. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[63] L. Bouthillier, "Streaming media delivery for higher education," *Streaming Media Inc: Innovation Series*, vol. Online Learning: Textbook Strategies for Video Education, no. 6, pp. 25–28, August 2007. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[64] M. Fritz, "Webcasting fosters better parents in washington," *Streaming Media Inc: Innovation Series*, vol. Webcast Essentials: Presenting Successful Events Online, no. 4, pp. 18–20, December 2005. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[65] J. Yu and C. T. Chou, "A dynamic caching algorithm based on internal popularity distribution of streaming media," Huazhong University of Science & Technology, China and University of New South Wales, Australia, September 2005.

[66] L. Bouthillier, "Streaming media's success story," *Streaming Media inc: Innovation Series*, vol. Solutions For Enterprise Streaming & Digital Media, no. 3, pp. 4–8, July 2004. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[67] Wikipedia, "Network simulation," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Network_simulation

[68] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in network simulation," in *Computer*. IEEE Computer Society, JulyMay 2000, vol. 33, no. 5, pp. 59–97. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[69] G. Wilkinson, "An investigation into network emulation and the development of a custom network," Master's thesis, Rhodes University, Grahamstown, South Africa, November 2007.

[70] D. C. Craig, "Extensible hierarchical object-oriented logic simulation with an adaptable graphical user interface," Science, Memorial University of NewFoundland,

Computer Based Learning Unit, University of Leeds, May 1996. [Online]. Available: http://www.cs.mun.ca/~donald/msc/thesis.html

[71] A. El-Haj-Mahmoud and A. Kayssi, "Emunet: A real-time IP network emulator," in *Symposium on Applied Computing: Proceedings of the 2004 ACM symposium on Applied computing.* Nocosia, Cyprus: ACM, 2004, pp. 357–362.

[72] P. K. Choudhary and K. S. Trivedi, "Discrete event simulation: Tools and applications," Center for Advanced Computing and Communication Department of Electrical and Computer Engineering Duke University, October 2004.

[73] Mesquite Software, "CSIM 19 product description," January 2008. [Online]. Available: http://www.mesquite.com/products/documents/CSIM19ProductDescription.pdf

[74] OPNET, "Opnet modeler: Accelerating network r&d," November 2007. [Online]. Available: http://www.opnet.com/solutions/network_rd/modeler.html

[75] Tetcos, "Netsim," November 2007. [Online]. Available: http://www.tetcos.com/index.html

[76] Scalable Network Technologies , "Qualnet developer," January 2008. [Online]. Available: http://www.scalable-networks.com/products/developer.php

[77] R. Bagrodia, K. Tang, S. Goldman, and D. Kumar, "An accurate, scalable communication effects server for the fcs system of systems simulation environment," *Proceedings of the 2006 Winter Simulation Conference*, August 2006.

[78] J. Wang, "ns-2 tutorial (2)," Multimedia Networking Group, The Department of Computer Science, UVA, October 2004.

[79] K. Fall and K. Varadhan, *The ns Manual (formerly ns Notes and Documentation)*, The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC., October 2005.

[80] G. F. Lucio, M. Paredes-Farrera, E. Jammeh, M. Fleury, and M. J. Reed, "Opnet modeler and ns-2: Comparing the accuracy of network simulators for packet-level analysis using a network testbed," in *3rd WEAS International Conference on*

Simulation, Modelling and Optimization (ICOSMO 2003), vol. 2, Crete, 2003, pp. 700–707. [Online]. Available: http://privatewww.essex.ac.uk/~fleum/weas.pdf

[81] T. Modi, "Your media, your way, anytime-anywhere," *Streaming Media Inc: Innovation Series*, vol. Best Practices: Media, Broadcacst & Entertainment, no. 5, p. 24, July 2006. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[82] P. Casella, "Enterprise IPTV: A video evolution that is shaping our lives, changing our teaching methodologies, and influencing our learning styles," *Streaming Media Inc: Innovation Series*, vol. Online Learning: Textbook Strategies for Video Education, no. 6, p. 37, August 2007. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[83] D. Glazier and B. Sikwane, "IPTV. one remote to rule them all," *iWeek*, vol. 99, pp. 16–18, May 2007. [Online]. Available: http://www.iweek.co.za/ViewStory.asp?StoryID=173609

[84] I. Pienaar, "To HD or not to HD?" *iWeek*, no. 109, August 2007. [Online]. Available: http://www.iweek.co.za/ViewStory.asp?StoryID=175848

[85] M. Billard, "QoS and cost-effective video delivery: No longer a pipedream," *Streaming Media Inc: Innovation Series*, vol. Best Practices: Media, Broadcacst & Entertainment, no. 5, pp. 28–29, July 2006. [Online]. Available: http://www.streamingmedia.com/whitepapers/

[86] Wikipedia, "Yuv," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Y'UV

[87] D. le Gall, "Mpeg: A video compression standard for multimedia applications," in *Communications of the ACM*, vol. 34, no. 4, April 1991, pp. 46–58.

[88] Wikipedia, "Disply resolution," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Image:Vector_Video_Standards2.svg

[89] ——, "Mpeg-1," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/MPEG-1

[90] S. Chamberlain, "Gnu + cygnus + windows = cygwin," Cygnus Solutions and Red Hat, October 2007. [Online]. Available: http://cygwin.com/

[91] E. Altman and T. Jiménez, "Ns simulator for beginners," University. de Los Anders, Mérida, Venezuela, and ESSI, December 2003.

[92] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options," Network Working Group, RFC 2018, October 1996. [Online]. Available: http://www.faqs.org/rfcs/rfc2018.html

[93] P. Haldar and X. Chen, "Ns tutorial 2002," Network Simulator (ns) Tutorial 2002, November 2002. [Online]. Available: http://www.isi.edu/nsnam/ns/ns-tutorial/tutorial-02/slides/NS_Fundamentals_1.ppt

[94] Information Sciences Institute, "RFC793 - Transmission Control Protocol," Defense Advanced Research Projects Agency, University of California, RFC 793, September 1981. [Online]. Available: http://www.faqs.org/rfcs/rfc793.html

[95] J. Postel and J. Reynolds, "RFC959 - File Transfer Protocol," Network Working Group, ISI, RFC 959, October 1985. [Online]. Available: http://www.faqs.org/rfcs/rfc959.html

[96] J. Postel, "RFC318 - Telnet Protocol," Network Working Group, UCLA-NMC, RFC 318, April 1972. [Online]. Available: http://www.ietf.org/rfc/rfc318.txt

[97] ——, "RFC768 - User Datagram Protocol," Information Sciences Institute, UCLA-NMC, RFC 768, August 1980. [Online]. Available: http://tools.ietf.org/rfc/rfc768.txt

[98] Wikipedia, "Pareto distribution," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/Bit_rate

[99] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid - a framework for video transmission and quality evaluation," in *13th International Conference on Modelling Techniques*

and Tools for Computer Performance Evaluation. Urbana, Illinois, USA: Technical University of Berlin, Telecommunication Networks Group (TKN), September 2003.

[100] A. Lie and J. Klaue, "Evalvid-ra: trace driven simulation of rate adaptive mpeg-4 vbr video," *Multimedia Systems*, vol. 14, no. 1, pp. 33–50, June 2008. [Online]. Available: http://www.springerlink.com/content/e5557223016234kv/fulltext.pdf

[101] K. Chih-Heng, "How to evaluate mpeg video transmission using the ns2 simulator," Kinmen Institute of Technology Information Management, Taiwan, May 2006. [Online]. Available: http://hpds.ee.ncku.edu.tw/~smallko/ns2/Evalvid_in_NS2.htm

[102] ——, "Ns2 tutorial," EE Department, NCKU, November 2005. [Online]. Available: http://hpds.ee.ncku.edu.tw/~smallko/ns2/ns2_051126.ppt

[103] G. Combs, "Wireshark: Go deep," November 2007. [Online]. Available: http://www.wireshark.org/

[104] N. Singh, "Performance analysis for objective methods of video quality assessment," Flextronics Software Systems, TechOnline, Online, October 2005. [Online]. Available: http://www.techonline.com/learning/techpaper/192200257

[105] Wikipedia, "Peak signal to noise ratio," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/PSNR

[106] ——, "Awk," Wikipedia, the free encyclopedia, January 2008. [Online]. Available: http://en.wikipedia.org/wiki/AWK

[107] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF, Standards Track RFC1889, January 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1889.txt

[108] T. Nakashima, "Propagation property of jitter with self-similarity," *International Journal of Innovative Computing, Information and Control*, vol. 2, no. 3, pp. 567–580, June 2006.

[109] R. Widoyo, “Measurement of streaming media applications running over wireless lans,” SENG 9338 Network Project, Tech. Rep., October 2002.

[110] S.-P. Chan, C.-W. Kok, and A. K. Wong, “Multimedia streaming gateway with jitter detection,” in *IEEE TRANSACTIONS ON MULTIMEDIA*, vol. 7, no. 3, 2005, pp. 585 – 592.

[111] J. Chung and M. Claypool, “Ns by example,” Worcester Polytechnic Institute WPI, October 2007. [Online]. Available: http://en.wikipedia.org/wiki/Comparison_of_streaming_media_systems

[112] V. Jacobson, C. Leres, and S. McCanne, “Tcpdump/libpcap public repository,” http://sourceForge.net, November 2007. [Online]. Available: http://www.tcpdump.org/

[113] bandwidthLimiter.com, “Traffic shaper xp,” Bandwidthlimier.com, July 2008. [Online]. Available: http://bandwidthcontroller.com/index.html

[114] U. Arizona State, “Video traces research group,” Arizona State Univeristy, 12 2007. [Online]. Available: http://trace.eas.asu.edu/index.html

[115] J. Klaue, “Evalvid with gpac - usage,” Evalvid - A Video Quality Evaluation Tool-set, November 2007. [Online]. Available: http://www.tkn.tu-berlin.de/research/evalvid/EvalVid/docevalvid.html

Appendix A

NS-2 Simulation Script - TCL

```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Open the traffic trace file to record all events
set nd [open out.tr w]
$ns trace-all $nd

#Define a 'finish' procedure
proc finish {}
{          global ns nf nd
           $ns flush-trace
           #Close the NAM trace file
           close $nf
           close $nd
           #Execute NAM on the trace file
           exec nam out.nam &
           exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

#Give node position (for NAM)
$ns duplex-link-op
$n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

#Setup a TCP connection
set tcp [new Agent/TCP]

$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink $tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP

#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null

$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp $cbr
set type_ CBR $cbr
set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false
```

```
#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
```

Appendix B

Packet Loss script

```
#This program is used to calculate the packet loss
#for both CBR packets and TCP packets

BEGIN
{
  # Initialization. Set the variables
  numPackets = 0;      #total number of packets
  CBRDrops = 0;        #total number of lost CBR packets
  tCPDrops = 0;        #total number of lost TCP packets
  videoDrops = 0;      #total number of dropped Video Packets
  total = 0;           #total number of lost packets
}
{
  ##################################################################
  # eg trace : +  0.1  1  2  cbr 1000 ------- 2  1.0  3.1  0  0 #
  # values   : $1 $2   $3 $4  $5  $6          $8  $9            #
  ##################################################################

  action = $1; # r Recv, d Drop, e Err, + Enque, - Deque [string]
  time = $2;        # Realtime value of network events    [int]
  from = $3;        # Source Node                         [int]
  to = $4;          # Destinaion Node                     [int]
  type = $5;        # Packet Name (cbr,tcp,ack)           [string]
  pktsize = $6;     # Packet Size                         [int]
                    # Optional flags would be ------      [String]
  flow_id = $8;     # Flow ID - Flow of traffic           [int]
  src = $9;         # Source Address                      [int]
  dst = $10;        # Destination Address                 [int]
  seq_no = $11;     # Sequence Number                     [int]
  packet_id = $12; # Unique Packet ID                     [int]
  # count packtes from node 1 to node 2 & also check enquque
  if (from==0 && to==1 && action == "+")
         numPackets++;
  if (flow_id==2 && action == "d")
         cBRDrops++;
  if (flow_id==1 && action == "d")
         tCPDrops++;
  if (flow_id==0 && action == "d")
         videoDrops++;
  if (action == "d")
         total++;
}

END
{
  printf("number of packets sent:%d \n", numPackets);
  printf("number of Video packets lost:%d \n", videoDrops);
  printf("number of CBR packets lost:%d \n", cBRDrops);
  printf("number of TCP packets lost:%d \n", tCPDrops);
  printf("number of total packets lost:%d \n", total);
  printf("Packet Loss percentage: %d% \n", videoDrops / numPackets * 100);
}
```

Appendix C

Latency script

```
#This program is used to calculate the end to end delay
#for all packets within the network

BEGIN
{
  # Initialization. Set variable fsDrops: packets drop.
  #                    numFs: packets sent
  highest_packet_id = 0;  # Highest Packet Identifier
  delayed = 0;            # total number of delayed packets
  total = 0;              # total number of packets
}

{
  ###############################################################
  # eg trace : +  0.1  1  2  cbr 1000 ------- 2  1.0  3.1  0  0 #
  # values   : $1 $2   $3 $4  $5  $6          $8  $9            #
  ###############################################################

  action = $1; # r Recv, d Drop, e Err, + Enque, - Deque [string]
  time = $2;       # Realtime value of network events    [int]
  from = $3;       # Source Node                         [int]
  to = $4;         # Destinaion Node                     [int]
  type = $5;       # Packet Name (cbr,tcp,ack)           [string]
  pktsize = $6;    # Packet Size                         [int]
                   # Optional flags would be ------      [String]
  flow_id = $8;    # Flow ID - Flow of traffic           [int]
  src = $9;        # Source Address                      [int]
  dst = $10;       # Destination Address                 [int]
  seq_no = $11;    # Sequence Number                     [int]
  packet_id = $12; # Unique Packet ID                    [int]
  # The program creates an arraylist of start times and an
  # arraylist of end times to keep track of the times for
  # each packet. Since this language only has only
  # 1-dimensional arraylists, we have to seperate arraylists
  # to keep track of the packet information.

  # Keeps track of the highest packet id
  if ( packet_id > highest_packet_id)
          highest_packet_id = packet_id;

  # Ensures the begin_time[packet_id] is 0 or new start time
  if (begin_time[packet_id] == 0)
          begin_time[packet_id] = time;

  # Checks that packets are not dropped and if recieving..
  # Gets the new end_time value for that particular packet
  if (flow_id == 0 && action != "d")
  {
    if ( action == "r" )
    {
       end_time[packet_id] = time;
    }
  }

  # if the packet does not meet the above set the negative
  # flag to prevent it from being printed out.
  else
  {
      end_time[packet_id] = -1;
  }
}

END
{
  for (packet_id = 0; packet_id <= highest_packet_id; packet_id++)
  {
    #Now iterate through the list and compare times.
    start = begin_time[packet_id];
    end = end_time[packet_id];
    packet_duration = end - start;
    if (start < end)
    {
      printf("Packet ID: %d, Start time: %f,  packet_id, start);
      printf("Latency: %f \n", packet_duration);
    }
  }
}
```

Appendix D

DVD-ROM

Thesis Hard-copy

An electronic version of this thesis is provided in the *thesis* folder.

Images and Graphs

Organised according to the following directory structure within the *Results* folder.

Scenario 1	**Scenario 2**	**Scenario 3**
Akiyo CIF and QCIF	Akiyo CIF and QCIF	Akiyo CIF and QCIF
Foreman CIF and QCIF	Foreman CIF and QCIF	Foreman CIF and QCIF
Hall CIF and QCIF	Hall CIF and QCIF	Hall CIF and QCIF

Simulation and Evaluation Scripts

Located in the *CAFSS-Net* folder and follows the same directory structure as the table above.

Literature and References

Located in the *SOURCES* folder according to Author(s) and Year.

Video Samples

Located in the *VIDEOS* folder and represents only the first scenario.