An Adaptive SVC-video Streaming Scheme for Mobile WiMAX Networks

Khotso Hamilton Keta

This thesis is submitted in partial fulfillment of the academic requirements
for the degree of
Master of Science in Electrical Engineering
in the Faculty of Engineering and The Built Environment
University of Cape Town

October 2011

As the candidate's supervisor, I have approved this dissertation for submission.

Name: Dr. Alexandru Murgu

Signed: _____

Date: _____

Declaration

I hereby declare that this dissertation is my own unaided work, both in conception and execution, and that apart from the normal guidance of my supervisor. I have received no assistance except as stated in the text of this document. Neither the substance nor any part of the thesis has been submitted in the past, or is being, or is to be submitted for a degree in the University or any other University.

I am presenting the thesis for examination for the degree of MSc in Electrical Engineering. I also grant the University free license to reproduce the above thesis in whole or in part, for the purpose of research.

Khotso Hamilton KETA

_____ Date _____

Abstract

The advances in video technology and increased telecommunications network capacity have made it technologically and economically feasible to deliver high quality video content to a vast number of users over fixed and mobile broadband networks, such as WiMAX. These advancements have made it possible for mobile users to access multimedia services, such as mobile TV, Internet TV, and iTunes multimedia download.

The problem that is being addressed in this study is bandwidth fluctuation that occurs during a video stream delivery over the Mobile WiMAX network, which results in video quality degradation. To address this problem, H.264AVC/SVC coded video is used. In the literature, SVC coded video has been found to offer improved visual quality compared to the preceding standards and proved to be suitable for dynamic bandwidth environments, such as Mobile WiMAX network. However, SVC video exhibits significant bit rate variability, which brings in bandwidths efficiency problem. This study is aimed at addressing the bandwidth fluctuation problem and the bandwidth efficiency problems.

The objectives of this thesis are to investigate the challenges of video streaming over Mobile WiMAX networks, to explore the mechanisms used to offer scalable video over wireless networks, and to develop a video streaming scheme that minimises packet loss and optimises visual video quality for Mobile WiMAX networks. The major focus of this study is on developing a streaming server that is bandwidth adaptive.

We propose a video streaming algorithm that is based on the pipelining design technique and the algorithm is compared to the most prominent video scheduling algorithm, Earliest Deadline First. For performance evaluation, JSVM tool was used for video encoding; NS-2 for network simulation and EvalSVC for video evaluation. The two algorithms were compared with respect to both quality of service and quality of user experience metrics. The pipelining-based scheduling algorithm was found to outperform the Earliest Deadline First as it experienced lower packet losses, higher average PSNR values with smaller standard deviations, lower packet losses and significantly reduced SVC traffic variability. Therefore, pipelining-based streaming scheme is most suitable for SVC video streaming over Mobile WiMAX networks.

Acknowledgements

I would like to sincerely thank my supervisor Dr. Alexandru Murgu for his willingness to supervise my work and the opportunity he granted me of being part of the communication research group (CRG). The time he provided for discussions and answers to my questions proved to be invaluable.

I would also like to thank members of the CRG group as well as network applications research group for their time, advices and contributions which helped me make this work a success.

To my friends and family, thank you so much for your financial and emotional support. I appreciate all the compromises and the sacrifices you have made. If it was not because of your sacrifices, my stay in Cape Town could not have been possible. May God bless you.

I would like to also thank Mrs Alison Gwynne-Evans for prove reading my thesis.

Last but not the least; I would like thank the Giver of life for all the blessings including the great gift of life.

Table of Contents

List of Figures

List of Abbreviations

WiMAX	Worldwide Interoperability for Microwave Access
SVC	Scalable Video Coding
AVC	Advanced Video Coding
QoS	Quality of Service
QoE	Quality of user Experience
GOP	Group of Pictures
FEC	Forward Error Correction
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase-Shift Keying
TDD	Time Division Duplexing
FDD	Frequency Division Duplexing
OFDMA	Orthogonal Frequency Division Multiple Access
ITU-T	International Telecommunication Union
NALU	Network Abstraction Layer Units
TID	Temporal Identifier
QID	Quality Identifier
MPEG	Moving Pictures Experts Group
I-frame	Intra-coded frame
P-frame	Predictive frame
B-frame	Bi-predictive frame
MGS	Medium Granular Scalability
EDF	Earliest Deadline First
SD	Standard Definition

MDC	Multiple Description Coding
TV	Television
VBR	Variable Bit Rate
MAN	Metropolitan Area Network
PSNR	Peak Signal-to-Noise Ratio
VoIP	Voice over Internet Protocol
MPLS	Multiprotocol Label Switching
CID	Connection Identifier
SFID	Service Flow Identifier

Chapter 1 Introduction

1.1 Video Streaming over Mobile WiMAX

The rapidly growing demand for mobile broadband data has resulted in the development and deployment of Mobile WiMAX. Mobile WiMAX is a broadband wireless solution that enables convergence of mobile and fixed broadband networks through a common wide area broadband radio access technology [1]. Currently, over 500 commercial WiMAX networks, both fixed and mobile WiMAX, have been deployed in over 149 countries around the globe, offering coverage to 630 million people [2]. In addition, WiMAX is rapidly introduced in developing countries where cellular penetrations are still low and the demand of broadband wireless access is considered key for development [3]. Today, Mobile WiMAX subscriptions are over 10 million globally and are projected to reach 130 million by 2014 [2]. Figure 1 shows a worldwide WiMAX deployment map, in which the blue and red pins represent Mobile WiMAX and Fixed WiMAX deployments.

Figure 1.1 Worldwide WiMAX deployment map [4].

Mobile WiMAX networks features coupled with the growing availability of feature-rich mobile devices have made it technically and economically feasible to offer mobile multimedia services to users in many different sectors, including enterprise, health, education and the military [2, 3]. Furthermore, mobile users are now able to access multimedia services, such as Mobile TV, Internet TV, and iTunes multimedia downloads. These services are delivered to the users using either of the following fundamental approaches: download-and-play or video streaming [5].

Download-and-play is a video delivery approach in which a video file is entirely downloaded to the client device before the play-out process can begin. The major setbacks to this approach are the need for file storage and a long play-out delay [5]

In contrast, video streaming is an approach in which video content is delivered to a viewing device for immediate display [5]. Video streaming eliminates the challenges of down-and-play approach. However, it has its own challenges. The major challenge of this approach is that the available network bandwidth should be greater than or equal to the bit rate of the bit stream that is being delivered over the network [5]. Otherwise, the quality of the video received at the viewing device is drastically degraded. This research investigates the problem of video quality degradation while streaming video over Mobile WiMAX networks.

According to Cisco Systems [6], video streaming will grow at a faster rate over the next few years. It predicts that video traffic will exceed 91% of all global traffic by 2014. This growth is attributed to popularity of video streaming services, such as Mobile TV, video on demand and high definition TV [6]. Therefore, video streaming will be increasingly important.

To provide an overview on how video streaming works, Figure 1.2 illustrates a video streaming system.

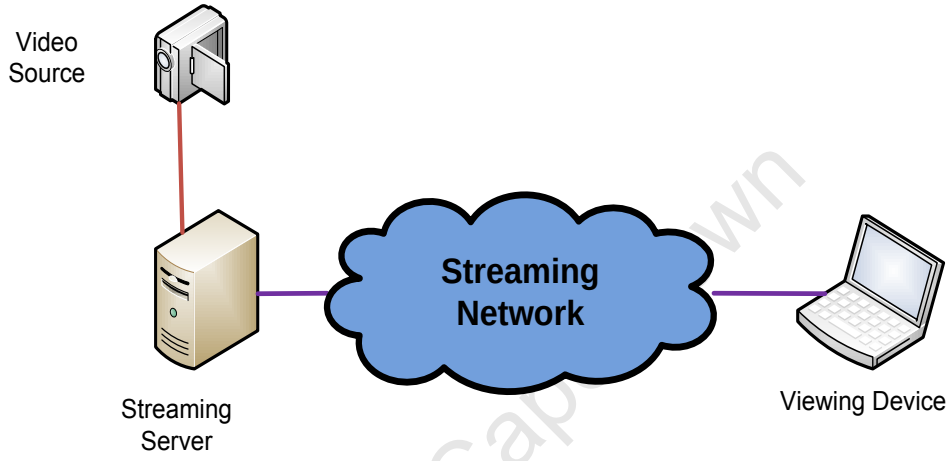

Figure 1.2 A Typical Video Streaming System [5]

The four major components of a typical streaming system are: a streaming network, a video source, streaming server, and viewing device. These components are briefly described below.

The streaming network provides the communication link between the streaming server and the viewing device. The streaming network can be a composite of various networks. In this study, last mile connectivity is provided by the Mobile WiMAX network. This is because Mobile WiMAX has a number of features that support video streaming. Some of its prominent features are high data rates, quality of service provisioning, and mobility support. According to the WiMAX technical report [1], peak data rates of up to 63 Mbps on the downlink and of up to 28 Mbps on the uplink, per sector in a 10 MHz channel are obtainable. However, Mobile WiMAX networks are characterized by time varying transmission bandwidth which makes video streaming over them a challenge [7].

Another component of the streaming system is a video source. The video source or camera produces raw video which is sent to the streaming server. The raw video consumes a

large amount of bandwidth. To achieve transmission bandwidth efficiency, raw video is compressed prior to its transmission over the streaming network [5]. Video compression modes are briefly discussed below.

The following two modes are used for video compression: non-scalable video coding and scalable video coding. In non-scalable video coding, coding efficiency is the most important factor and the compression is optimized at a rate known prior to encoding [8]. The major setback of this method is that it is difficult to adaptively stream non-scalable video streams over time-varying communication channels, such as Mobile WiMAX [9]. In contrast, with scalable video coding, a video bit-stream can be truncated to obtain sub-set bit-streams of lower temporal resolution, lower quality signal and/or lower spatial resolution. Scalable video coding techniques provide functionalities such as transmission bandwidth and video quality adaptation [8, 9]. For this reason, in this study we are primarily concerned with scalable video coding (SVC) technique.

As mentioned earlier, a streaming server is one of the key components of a streaming system. A streaming server is responsible for sending video streams to the viewing device. It takes video content that has been stored internally and creates a stream for each viewer request [5]. In addition, a streaming server is used to adapt the rate of the stream that is being sent to a viewer based on the changing network conditions. The server carries out bit rate adaptation in different ways [9], which are detailed in the next chapter. This study is aims at developing a server-based bandwidth adaptive video streaming scheme.

The last component of a video system we look at is a viewing device. A viewing device processes the bit-stream that is being received and displays video on its screen. It utilizes player software that resides in it. A number of functions, such as re-synchronizing the incoming streams, are carried out by this software and its performance can have a major impact on user satisfaction [5]. However, in this study the performance of the viewing device is not considered.

In the documentation of current studies, work has been conducted [7, 10-12] which was aimed at addressing challenges of video streaming over Mobile WiMAX networks. The major findings are discussed in Chapter Two. The next subsection describes some of the challenges and defines the problem addressed in this study.

1.2 Problem Definition

As was introduced earlier, this study investigates the problem of video quality degradation while streaming video over Mobile WiMAX networks. Video streaming over Mobile WiMAX faces challenges due to the characteristics of the network. In essence, Mobile WiMAX networks are characterized by time varying transmission bandwidth due to a number of factors, including user mobility, network congestion and other network imperfections [13]. The time varying transmission bandwidth may result in increased network congestion, packet loss and packet delay variation which, if not controlled, drastically degrades video quality.

Since the bandwidth problem is inherent in Mobile WiMAX, it is difficult to solve. The effect is minimized by improving the streaming system. This can be done by minimizing packet loss that occurs when available transmission bandwidth decreases. Packet loss is not only minimized, but the streaming system must provide the optimum video quality with the available bandwidth. This is the core of this research work.

To get a better understanding of the problem addressed in this study, consider a user travelling by bus accessing Mobile TV services over Mobile WiMAX networks. The user is watching a news bulletin, but as the bus moves the video that is being displayed may become jerky and annoying to the user [14]. Consequently, the user may abandon the service. This would be undesirable to the service providers as it affects their revenue.

What happens technically in the scenario described above in terms of bandwidth dynamics is illustrated in Figure 1.3. The figure depicts a typical bandwidth graph that belongs to a video streaming client in a Mobile WiMAX network. At period Δt, bandwidth (BW) is decreasing going below the rate (optimum bandwidth) required to successfully the video sequence. If at this period, the streaming server does not decrease sending rate, it will inject large amount of video traffic that will be lost and hence video quality degradation. In addition, a non-bandwidth adaptive streaming server can worsen network congestion.

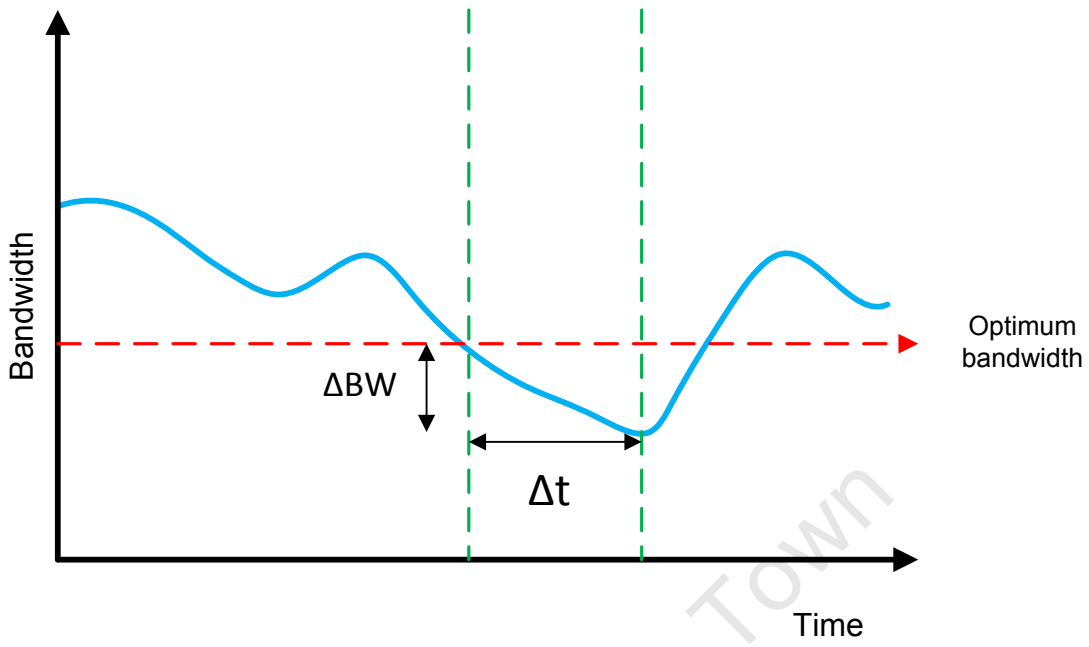

Figure 1.3 Bandwidth Pattern

As has already been mentioned, Scalable Video Coding (SVC) is used in this study. SVC is an extension of the H.264/MPEG-4 Advanced Video Coding (AVC) standard. It offers high compression efficiency, improved rate-distortion and a wider range of scalability modes compared to the preceding video coding standards [15]. Due to higher compression gains, SVC saves bandwidth compared to preceding standards. H.264/AVC standard has made it possible to stream real-time standard definition at bit rates as low as 512kbps and is suitable for transmission over unpredictable wireless networks, such as Mobile WiMAX [2][9]. However, one of the major drawbacks of SVC is that it suffers from a high frame size variability which makes its network transportation challenging [16].

Although work has been done on video streaming over Mobile WiMAX and SVC video traffic variability in the literature, there is a need for research into optimizing visual video quality when streaming video Mobile WiMAX using bandwidth adaptive servers. These servers will improve bandwidth efficiency and offer quality of user experience while streaming video over Mobile WiMAX networks.

1.3 Research Objectives

The main objectives of the study are to:

i. Investigate the challenges of video streaming in the Mobile WiMAX networks.

ii. Design a video streaming scheme that supports quality of user experience (QoE) while streaming video over the Mobile WiMAX network.

iii. Implement and analyse the proposed video streaming scheme.

The specific objectives of the video streaming scheme are as follows:

- To offer graceful degradation of video quality as user bandwidth decreases in Mobile WiMAX networks.
- To minimize packet loss that occurs when available bandwidth decreases during a streaming session.
- To improve network bandwidth efficiency.

1.4 Research Scope and Limitations

The video streaming field is broad. As a result, performance enhancement of the video streaming systems can be carried out at different layers of the Internet protocol (IP) stack. For instance, Mobile IP, a layer three protocol, is used to address the seamless mobility problem during a video streaming session and may be seen to improve the performance of the video streaming system. Therefore, this study is limited to video streaming at the application layer.

In addition, a study on the Mobile WiMAX network can be carried out in different parts of the network, that is, either in core or access networks, or both. This study is limited to the access part of the Mobile WiMAX network.

Another restriction is on the video application supported in this study. The Mobile WiMAX network architecture supports different types of video streaming applications, including stored video and live video. The type of video streaming application influences the design of a video streaming scheme. Therefore, this investigation is restricted to stored video. This is because, in the proposed scheme, the streaming server needs to have information of all frames

constituting a group of pictures before sending the first frame in the group

Besides the application type, another limitation is on the video coding standard. The proposed scheme is limited to scalable coded video. This is because the scheme segments video frames and some of the segments of the frame may not be transmitted when transmission bandwidth is not enough. Unlike the non-scalable video coding techniques which drop the frame if it is not completely received, scalable vide coding (SVC) allows decoding of partially received frames.

There are a number of additional factors that affects the quality of the video that is displayed on the viewing device, such as the screen size. In this work, the focus is on the time dynamic transmission bandwidth of the Mobile WiMAX network.

1.5 Research Contributions

The contribution of this study is twofold. In the first place, the study contributes to knowledge as it shows how pipelining, a method that is normally used in computer architecture, can be incorporated into video steaming to address the bandwidth problem in the Mobile WiMAX network [17]. Pipelining is the process in which instructions are broken down into sub-instructions which are interleaved during execution with the aim of saving execution time.

Secondly, there is the practical application of the proposed scheme. The outcome of this study can be applied in Mobile WiMAX networks to offer quality of service as well as quality of user experience when offering video streaming services to mobile users.

1.6 Thesis Outline

The remainder of this thesis is organized as follows:

- Chapter 2 presents the relevant background information on the Mobile WiMAX standard, scalable video coding and video streaming. The chapter also discusses features of Mobile WiMAX that supports video streaming, the challenges of video streaming over Mobile WiMAX and the approaches used in the literature to address the challenges.

- Chapter 3 presents the proposed video streaming scheme. It begins by introducing pipelining; a method used to schedule video frames in a manner that addresses the bandwidth problem. This is followed by a discussion on the priority planning. Then, the design details of the proposed video streaming scheme on both the server and receiver sides are presented.
- Chapter 4 provides details on the evaluation framework used in this study. It starts by presenting the objectives of the framework, and then provides an overview of the software tools used. The chapter then proceeds to discuss how the proposed video streaming scheme was implemented in Network Simulator - 2. The simulation configuration and the experimental procedure are also presented.
- Chapter 5 presents the evaluation results and a discussion on the results obtained. It starts by defining the performance metrics of interest. This is then followed by the presentation and analysis of the results obtained using the ns-2 simulator and video evaluation tools.
- Chapter 6 presents a set of conclusions relating to the suitability of pipelining in video streaming to minimise packet loss and optimise visual quality when streaming video over the Mobile WiMAX networks. Then, the recommendations for future improvements and investigations are presented.

Chapter 2 Background and Literature Review

2.1 Introduction

The main goal of this chapter is to familiarize the reader with the subjects at hand. Thus, it presents background information on Mobile WiMAX, scalable video coding and video streaming.

The chapter begins by discussing the Mobile WiMAX standard and its features. Then, it presents an overview of the H.246 AVC Scalable Video Coding extension. It briefly covers video streaming, as well as the requirements and recommendations of the ITU-T and the WiMAX Forum on supporting video streaming applications. The chapter also provides a brief overview of different video adaptation methods found in literature as well as discussion on each.

2.2 Mobile WiMAX

2.2.1 WiMAX Overview

There has been a growing demand for broadband mobile services. Conventional wired high-speed broadband access networks are costly to deploy in rural areas and they also lack terminal mobility support required by mobile services. Mobile WiMAX - Worldwide Interoperability for Microwave Access - was developed to provide a flexible, efficient and cost-effective solution to these problems [13].

The first WiMAX standard (IEEE 802.16d-2004) offers wireless broadband access to fixed and nomadic users. The physical (PHY) layer defined by this standard uses orthogonal frequency division multiplexing (OFDM) to provide strong performance in multipath and non-line-of-sight (NLOS) environments. The standard defines only the physical and medium access control layer of the wireless MAN networks [13] [2].

In 2005, the amendment was made to the IEEE 802.16d-2004 standard, and the resulting standard is called IEEE 802.16e-2005. IEEE-802.16e PHY layer is based on orthogonal frequency division multiple-access (OFDMA) and the standard added new features, including support for mobility. This standard is commonly known as Mobile WiMAX [3].

Beside support for mobility and handover, Mobile WiMAX brings in the features such as scalable OFDMA, advanced antenna systems, services flows, QoS, spatial multiplexing, encryption etc [14]. The next two subsections discuss the Mobile WiMAX PHY and MAC layers.

2.2.2 The PHY layer

2.2.2.1 Introduction

This sub-section is aimed at discussing the following features and attributes of the Mobile WiMAX PHY layer:

- Scalable channel bandwidth
- Adaptive modulation
- Forward Error correction
- Frame structure
- Data rates

2.2.2.2 Scalable channel bandwidth

The PHY layer of Mobile WiMAX is based on the concept of scalable OFDMA. Scalability is one of the key features of OFDMA for support of a wide range of bandwidths and high throughputs. The scalability in OFDMA is achieved through adjusting the FFT size from 128 to 512, 1024 and 2048 to support channel bandwidths of 1.25 MHz, 5 MHz, 10 MHz and 20 MHz respectively [15].

The channel bandwidth determines the number of OFDM subcarriers that are usable in an OFDMA scheme. The parameters selected in IEEE 802.16e-2005 are such that the subcarrier spacing is fixed (10.94 KHz) and what varies with channel bandwidth is the number of subcarriers. Fixing the sub-carrier spacing means that useful symbol time duration and guard time remain fixed so as to minimise the impact of bandwidth scaling to higher layers [15].

2.2.2.3 Adaptive Modulation

Mobile WiMAX provides for the use of adaptive modulation and coding schemes to optimise the throughput based on the channel conditions. When the channel conditions are good, the higher density modulation schemes (such as 64QAM) are used. The high-density modulation scheme support higher data rates but are not tolerant to interference or noise. The lower-density modulation schemes (such as QPSK) are the opposite of the higher-density ones. The support of the different modulation schemes is a key feature for maximising bit-rates in different operation environments [3].

The Mobile WiMAX base stations support QPSK, 16QAM and 64QAM. The mobile stations are required to support QPSK and 16 QAM and 64QAM optionally. Switching between modulation schemes happens on a frame to frame basis. As a result, depending on the error rate, the base station can switch from 64QAM to QSPK or from 64QAM to 16QAM, as shown in Figure 2.1 [3, 13]

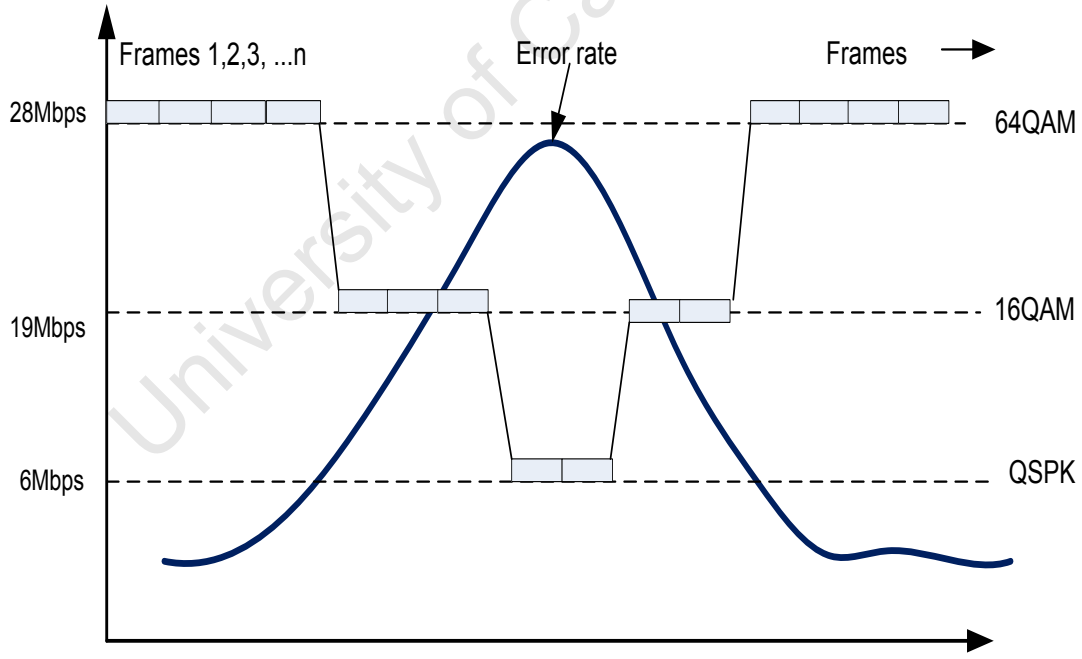

Figure 2.1 Adaptive modulation schemes maximize bit rate in low error rate environments[3]

Determination on the modulation scheme is done with the help of the channel quality indicator (CQI) channel, which updates the base station periodically about the status of the

channel. CQICH is part of the Mobile WiMAX frame.

2.2.2.4 Forward Error Correction (FEC)

Forward error correction is a commonly used error correction method used in digital transmission and broadcasting applications. It allows error correction by using redundancy information carried in the transmitted signal, and it helps maximise throughput by avoiding frame re-transmission. The redundancy bits can be carried out in a number of ways, such as parity bits or simple repetition of the transmitted signals. For efficiency, the FEC is tailored for the characteristics of the transmission system. Mobile WiMAX uses convolution coding with Reed Solomon (RS) error coding techniques [3].

2.2.2.5 Frame structure

Mobile WiMAX PHY supports both Frequency Division Duplexing (FDD) and Time Division Duplexing (TDD). However, in the initial Mobile WiMAX profiles, only TDD has been implemented. Therefore, the frame structure discussed in this subsection belongs to the TDD. One of the major advantages of TDD is that it enables the adjustment of the uplink to downlink frame ratio in order support asymmetric traffic in either direction, which is not the case in FDD as bandwidth is equally shared between the downlink and the uplink [3]. In addition, TDD needs a single channel for both the uplink and the downlink, and this provides flexibility for adaptation to varying global spectrum allocation. But FDD needs two separate channels for uplink and downlink [1].

The frame structure consists of two sub-frames, the downlink and uplink sub-frames. The sub-frames are separated by the Transmit/Receive or Receive/Transmit transmission gaps to avoid downlink and uplink collisions.

Figure 2.2 depicts the Mobile WiMAX frame structure. The control information used in the frame is as follows [1, 3]:

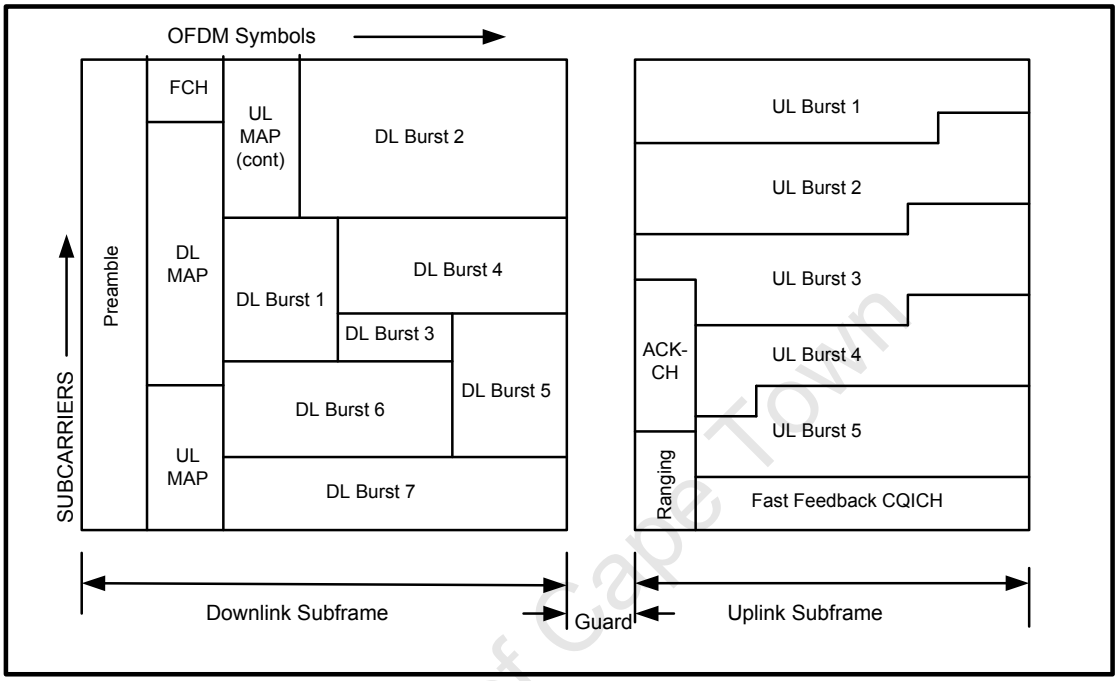

Figure 2.2 Mobile WiMAX frame structure

Preamble: The downlink sub-frame begins with the preamble, which is used for synchronization purposes.

Frame control header (FCH): The FCH succeeds the preamble and offers information about the length of MAP messages, coding scheme and the usable sub-channel.

DL-MAP and UL-MAP: The MAPS convey information about the sub-frame structure that will be used, the time slots, as well as the sub-channels assigned to the terminal.

UL Ranging: The UL ranging sub-channel is used by the mobile station (MS) to acquire the correct timing offset and power adjustments such that its transmission is aligned to the timing of the base station.

UL CQICH: is used by the mobile station to send channel state information to the base station.

UL ACK: is used by the Mobile station to feedback the downlink HARQ acknowledgements.

2.2.2.6 Data Rates

In Mobile WiMAX, data rates that can be supported depend on two parameters: channel quality and channel bandwidth. Channel quality is important in determining the appropriate modulation scheme as well as the Forward Error Correction (FEC). Mobile WiMAX (Profile 1) supports QSPK, 16QAM and 64QAM in the downlink and QSPK, 16QAM (and optionally 64QAM) in the uplink. Bandwidth support ranges from 1.25MHz to 20MHz. As explained in subsection 2.2.2.2, subcarrier spacing and symbol time are fixed. Therefore, increase in bandwidth implies increase in number of subcarriers. The subcarrier spacing and symbol time are used to determine the total amount of bandwidth available [3].

2.2.3 The MAC layer

2.2.3.1 Introduction

The MAC layer of Mobile WiMAX follows the basic functions of establishing connections, resource allocation, and ensuring quality of services as in Fixed WiMAX. This layer is the service interface of a WiMAX system [15] . This section covers the following three aspects of the MAC layer:

- Quality of service
- Service classes
- MAC scheduler

2.2.3.2 Quality of service support

WiMAX is connection oriented and supports QoS through the use of service flows. With the service flows, WiMAX is able to deliver services, such as voice or video streaming within agreed parameters. The QoS parameters associated with the service flow define the transmission ordering and scheduling on the WiMAX air interface [3].

Figure 2.3 illustrates QoS support in Mobile WiMAX. Before a service is offered, the base station and the user-terminal establish a unidirectional logical link (called a MAC connection) between the MACs of the two devices and the link is identified by its connection identification (CID). The "*sending"* device MAC associates the packets traversing it with

service flows to be delivered over the connection. This means that all protocol data units that flow to the "receiving" device contain the CID and the service flow identification (SFID) that has been established for the connection [1][3].

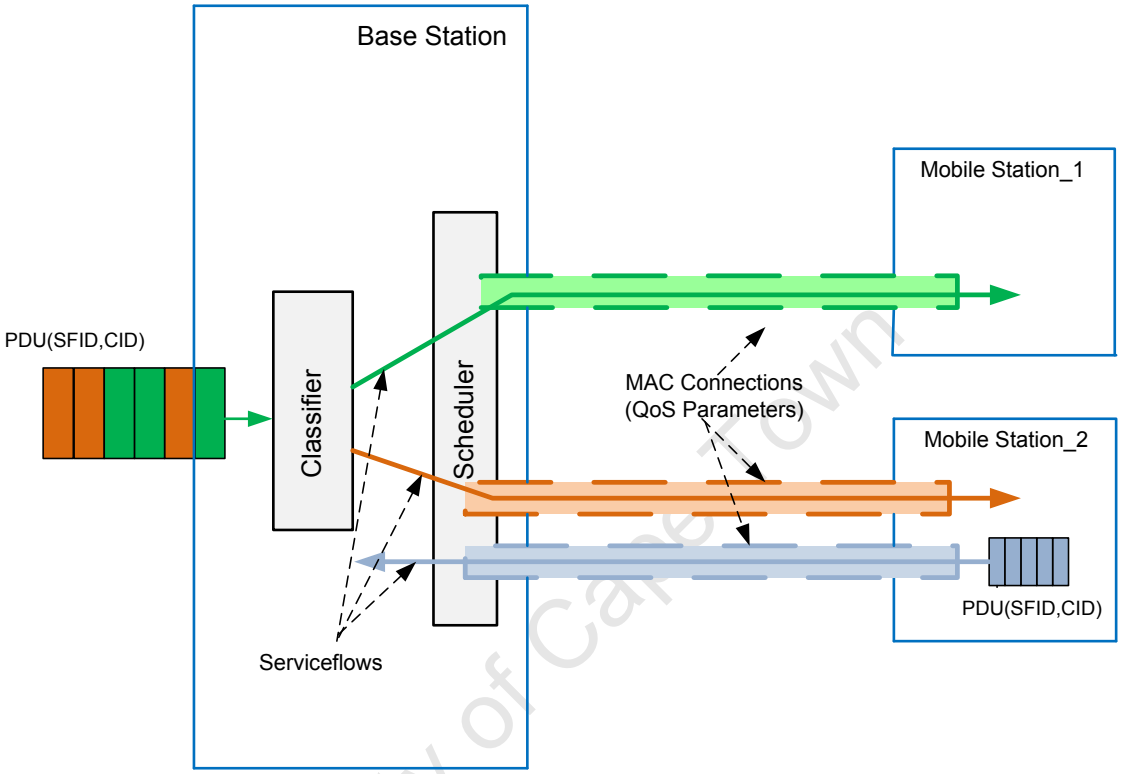

Figure 2.3 QoS support in Mobile WiMAX [16]

2.2.3.3 Service classes

As has been mentioned in the previous subsection, Mobile WiMAX supports a wide range of services and applications with different QoS requirements. It offers the following service classes [1-3]:

Unsolicited Grant Service (UGS): The UGS is designed to support real-time services that periodically generate fixed-size data packets. It provides fixed bit rate circuit emulation services such as T1 or E1. The base station schedules the capacity in the frames without any explicit request for each packet.

Real-Time Polling Service (rt-PS): The rt-PS is designed to support services that periodically generate variable-size data packets. Resources for the rt-PS are reserved based on connection type and service flow parameters, but are made available only on request.

Non-Real-Time Polling Service (nrt-PS): The nrt-PS is designed to support applications where the delays due to the network are not critical. Request for bandwidth is required and it is allocated based on meeting requirements of higher priority services.

Extended Real-Time Variable-Rate Service (ERT-VR): The ERT-VR is a combination of UGS and rt-PS. Hence, it has characteristics of both. This class is designed to support real-time service flows that periodically generate variable-size data packets..

Best Effort (BE): The BE is designed to support data streams that do not have minimum transmission delay. This class makes no guarantees on packet loss, delay or jitter.

2.2.3.4 MAC scheduler

The vital functional block of the MAC layer is the MAC scheduler. The MAC scheduler is responsible for ensuring service flows as well as quality of service. It determines which packets would flow over the air interface from among many applications, each with its class of service. The scheduling in carried out in both downlink and uplink and the quality channel indicator (QCI) plays an important role during the scheduling process. The scheduler also performs resource allocations and delivers them as MAPS messages. The resource allocations can be altered on a frame basis [1, 17].

2.3 Scalable video coding (SVC)

2.3.1 Introduction

The H.264/AVC video coding standard has provided significant video compression improvements over earlier versions [18]. To enhance the standard, the joint team of the ITU-T VCEG and the ISO/IEC MPEG completed the Scalable Video Coding (SVC) extension of the standard in 2007 [19]. The main objective of the extension has been to enable encoding of a high-quality bit stream consisting of multiple sub-streams, base layer (coarse visual quality sub-stream) and the enhancement layers (quality improvement sub-streams). This allows the scaling of the bit-stream to various conditions, such as bandwidth and terminal capability. In SVC, scalability refers to the removal of parts of the bit stream while the resulting bit stream remains decodable [8, 9]

SVC is a highly attractive solution to the challenges of high-quality video streaming over varying bandwidth networks, such as Mobile WiMAX [11]. This is because it can offer graceful quality degradation when the available network bandwidth decreases during a video streaming session by allowing the dropping of one or more enhancement layers to scale the bit stream to the available bandwidth [20]. In addition, priority layers can be protected in order to guarantee some level of QoE. The usual modes of scalability are temporal, spatial, and SNR/quality scalability. Temporal scalability is enabled through hierarchical prediction, whereas the spatial scalability and quality scalability are provided using a layered approach [21]. Further details on the modes of scalability are provided in subsection 2.3.3.

Besides varying or limited capacity networks, SVC can also be applied in [9, 20]:

- Heterogeneous devices: in order to provide a range of picture quality suited to the heterogeneous requirements and capabilities of the receiving terminals (such as display resolution, processing power or battery power).
- Surveillance video storage: base layer of the video is archived to save storage space.

The next subsection discusses the conceptual layers of SVC.

2.3.2 Conceptual layers

As mentioned, SVC is an extension of the H.264/AVC standard. As a result, it borrows much from the main standard. The conceptual design of SVC, which is similar to that of the H.264/AVC standard, is briefly discussed.

The design has two conceptual layers: a video coding layer (VCL) and a network abstraction layer. The VCL produces the coded representation of the source video data, while the NAL formats these data and provides header information to facilitate usage of VCL data in the variety of systems [21]. The two layers are briefly discussed.

2.3.2.1 Network abstraction layer (NAL)

The coded content is encapsulated in NAL units. A NAL unit resembles a transport packet, consists of a header and a payload part. It contains an integer number of bytes and the first three bytes represent the header and the remaining bytes represent the payload data [21]. Figure 2.4 illustrates the NAL unit header, showing the field name and number of bits in each

field.

R	I	PRID	N	DID	QID	TID	U	D	O	RR
1	1	6	1	3	4	3	1	1	1	2

Figure 2.4 SVC NAL unit header [22]

Below is a brief description of each field of the header: [22]:

- **R (reserved_one_bit):** A bit reserved for future extension of the header.
- **I (idr_flag)** : This field indicates whether the NAL unit is from an IDR frame (see section 2.3.3)
- **PRID (priority_id):** specifies a priority identifier for the NAL unit. A lower value of PRID indicates a higher priority.
- **N (no_inter_layer_pred_flag):** specifies whether inter-layer prediction may be used for decoding the coded slice.
- **DID (dependency_id):** the field indicates the inter-layer coding dependency level of a layer representation. (see subsection 2.3.3)
- **QID (quality_id):** this field indicates the quality level of an MGS layer representation (see subsection 2.3.3.)
- **TID (temporal_id):** This field indicates the temporal level of a layer representation. The temporal_id is associated with the frame rate, with lower values of a temporal_id corresponding to lower frame rates. (See subsection 2.3.3)
- **U (use_ref_base_pic_flag):** indicates that reference base pictures are used during the inter prediction process.
- **D (discardable_flag):** indicates discardable NALU units.
- **O (output_flag):** affects the decoded picture output process.
- **RR (reserved_three_2bits):** Reserved bits for the future extension.

NAL units fall into two categories: VCL NAL units and non-VLC NAL units. VCL NAL units contain coded slices while non-NAL units contain additional information that helps carry out other processes, such as decoding. A set of NAL units with the same properties constitutes an access unit, and decoding an access unit yields a picture [21].

Figure 2.5 depicts an access unit made up of two NAL units.

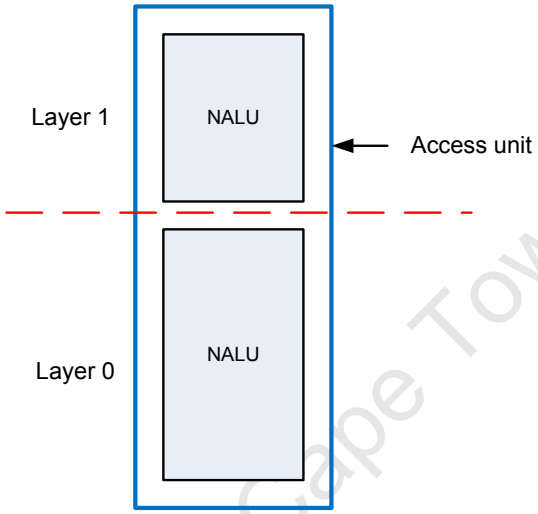

Figure 2.5 H.264 /AVC Access unit [23]

2.3.2.2 Video coding layer (VCL)

The video coding layer is an output of the encoding process, which is a sequence of bits representing the coded video data. To code data, this standard uses the block-based hybrid coding approach as in the preceding standards. However, the major improvement compared to the preceding standards is compression efficiency. In addition, the standard is also more flexible and adaptable [21].

The H.264/AVC standard divides pictures into macro-blocks and slices. Macro blocks of a picture are organized into one or more slices. SVC supports three types of slices: I-slice, P-slice and B-slice. The slices are organized into a frame of the slice type. Details of the macro-blocks and slices are beyond the scope of this work.

Below are the three frame types supported in SVC [8, 24]:

- IDR-frame: Instantaneous Decoder Refresh is a self-contained frame that is decoded

independently. This is normally the first frame of the video sequence and is inserted with the video sequence for re-synchronization purposes in case the transmitted bit steam is damaged. It is used for referencing when decoding other frames of the bit-stream. The major drawback of an IDR-frame is its size, it consumes many more bits but, conversely, it does not produce many artifacts.

- P-frame: The Predictive-inter frame is encoded or decoded by making reference to the previous IDR and/or P frame(s). This frame improves compression by storing the difference between the two consecutive images. In terms of size, the P frame is much smaller than the IDR frame. However, the drawback is that it is sensitive to transmission errors due to its dependence on earlier IDR and P frames. Prediction is represented by arrows in Figure 2.6.
- B-frame: The Bi-predictive inter frame is similar to the P-frame except that it makes reference to both an earlier and future frames. This is the smallest frame when compared to the other two frames. Importantly, 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-Frame computationally expensive to encode and decode.

In SVC, the B frame significantly improves the rate distortion compression efficiency. However, the cost of efficiency is traffic variability, and coding delays introduced by hierarchical B frames prediction [7]. The P-frames are used as key frames, which also helps in minimizing the number of IDR frames in the bit stream. The key frame along with all frames referencing it constitutes a group of pictures (GOP), which is illustrated in Figure 2.6. Loss or error in the key frame affects all frames of the GOP.

2.3.3 Modes of scalability

2.3.3.1 Temporal scalability

Temporal scalability provides a low frame-rate base layer and enhancement layers that build up to a higher frame rate [22]. A bit stream enables this type of scalability when a set of corresponding frames in a GOP is partitioned into a temporal base layer and one or more enhancement layers. The enhancement layer frames are usually coded as B frames that can be

dropped to scale down the video bit stream.

Figure 2.6 depicts a hierarchical prediction structure [21].

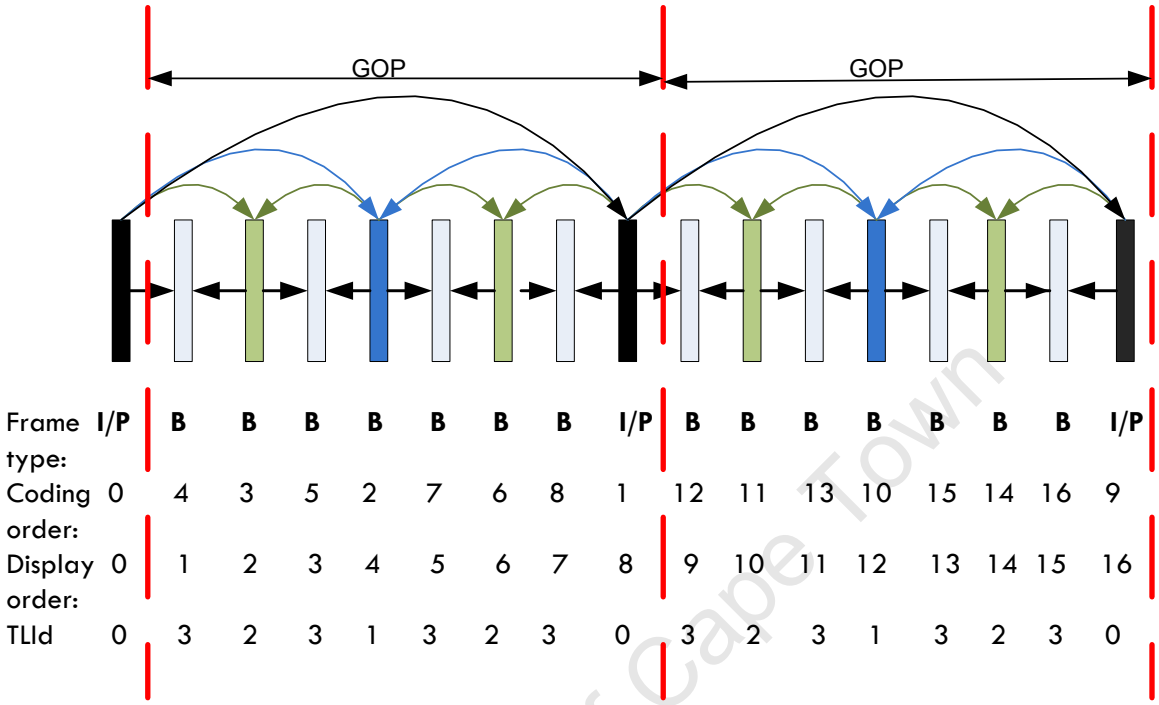

Figure 2.6 Temporal scalability [21]

To achieve temporal scalability, the construction of frame hierarchies is essential, as it will enable dropping of frames not used for prediction. The temporal identifier (TID) is used to determine the temporal level of the frames and the dropping of frames is done starting with the higher temporal level frames. For example, in Figure 2.6, the frames with temporal level equal to 3 can be dropped to reduce the frame rate by a factor of two, and further decrease the frame rate by a factor of two by removing temporal level 2 frames. This type of scalability is inherent in other modes of scalability [21, 25]

2.3.3.2 Spatial scalability

Spatial scalability enables coding of a picture as a hierarchy of spatial resolutions as shown in Figure 2.7. Decoding the base layer yields a low-resolution video sequence and encoding the base layer and one or more enhancement layers increases the resolution. Each layer corresponds to a supported spatial resolution and a spatial layer is referred to by a dependency identifier (DID). A DID of the base layer is equal to zero and it is increased by one from one

spatial layer to the next, as illustrated in Figure 2.7 [22, 25].

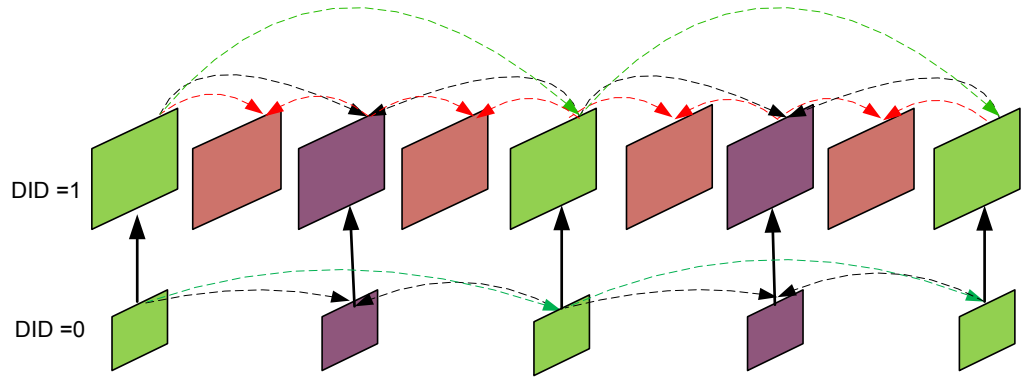

Figure 2.7 Spatial scalability [25]

2.3.3.3 SNR/Quality scalability

In quality scalability, the enhancement layers improve the visual quality of the bit stream. Quality scalability can be considered as a simple case of spatial scalability, where prediction dependencies exist between hierarchical frames. Instead of a different resolution, frames have different qualities. A quality identifier (QID) refers to a quality layer. Quality scalability has two modes: coarse granular scalability (CGS) and medium granular scalability (MGS) [22, 25]

In SVC, a CGS layer should be completely retained or removed when scaling down the bit stream. This is because partially received layers are not decodable. Hence, this limits the adaptability of the bit stream to the available bandwidth, as just a few bit rates are possible [21].

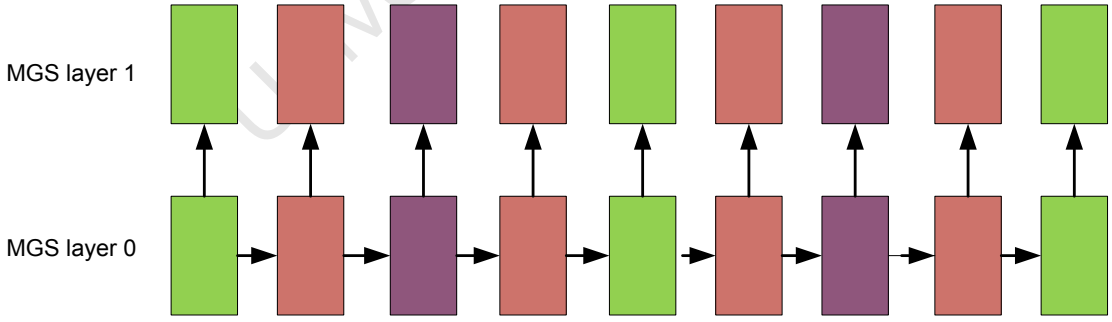

Figure 2.8 Medium granular scalability (MGS) bit stream [21]

In contrast to SVC, in MGS, any enhancement layer NAL unit can be discarded. A flag is used to indicate the discardable NALUs, (refer to subsection 2.2.3) provided there are no NALUs depending on it for prediction. Therefore most of the NALU which could have been

dropped in the CGS mode do not need to be dropped. Clearly, this mode provides more bit rate possibilities and better visual quality than the CGS mode. Figure 2.8 depicts an MGS coded bit stream [21]. The bit stream has two layers: MGS layer 0 and MGS layer 1.

2.4 Video streaming

2.4.1 Introduction

The previous two sections have covered the background information on Mobile WiMAX and scalable video coding, respectively. This section covers:

- An adaptive video streaming system overview
- The requirements of multimedia streaming applications
- Features of Mobile WiMAX that support video streaming
- Challenges of video streaming over Mobile WiMAX
- Methods that can be used to adapt or scale video to the available network bandwidth

2.4.2 Adaptive video streaming system overview

Video streaming refers to the real-time transport of live or stored video. In live video, video capturing, compression and distribution are done in real-time and end users can view video instantly. In stored video, video is captured, compressed and stored. Video distribution occurs later on at the end user's request [4]. Typical examples of video streaming applications include webcasting, Mobile TV, Video on Demand and high definition television broadcasting (HDTV) [3].

An adaptive video streaming system, illustrated in Figure 2.9, can be divided into the following six areas [26]:

- **Video compression**: To save bandwidth, video is compressed prior to transmission over the network. Video compression is classified into: non-scalable and scalable video compression coding. This work focuses on scalable video compression as it can cope

with bandwidth fluctuations.

- **Streaming server**: Streaming servers are required to perform under timing constraints in order to offer quality streaming services. They are required to offer VCR-like functions, such as fast forward, pause, etc.
- **Application layer QoS/QoE control**: Various mechanisms are used to make video cope with varying network conditions and to ensure QoE. These include application-layer control techniques, such as congestion control and forward error correction (FEC). In this work, network monitoring is done so that the streaming server can scale the bit stream to available bandwidth in order to packet loss and significant video quality degradation.
- **Media distribution network**: Enough support from the network is required to minimise packet loss and delay in order to offer reliable stream delivery. Mobile WiMAX meets this requirement.
- **Protocols for video streaming**: Transport, network and session layer protocols are usually required to stream video from the server to the receiver over the transport network. However, protocols are not in the scope of this work.
- **Media synchronization at the receiving side**: The receiver should be able to correct small scale time packet variations through buffering, and to decode received frames and synchronise video and audio tracks.

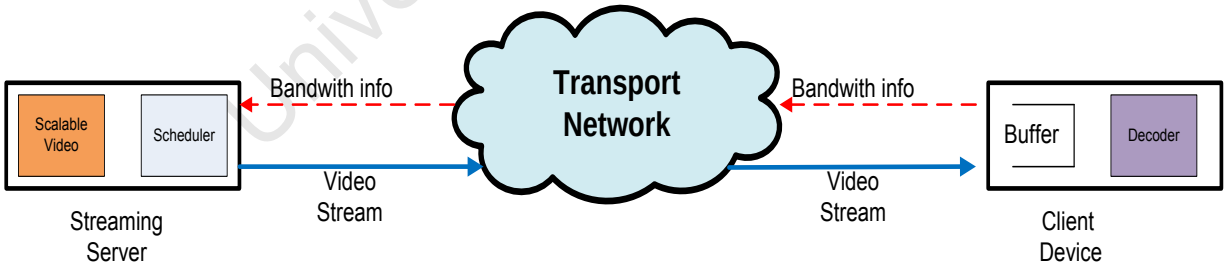

Figure 2.9 Adaptive video streaming system

2.4.3 Requirements of the video streaming applications

Multimedia streaming applications typically have strict delay, jitter and packet loss

requirements. Below are movie streaming application performance requirements from the WiMAX Forum [16]:

- **Packet delay variation within a flow**: must be less than 2sec
- **Information loss (PER):** must be below 0.5 %
- **One-way radio access network transfer delay**: must be below 250msec

The ITU-T also has its recommendations and for the MM4b video class – the class in which all frames are encoded and the frame rate is ~30 or ~25 frames per second - the recommendations are as follows [9]:

- **Format**: Common Intermediate Format
- **Delay:** ≤ ~150 ms
- **Latency variation**: ≤ ~50 ms
- **Nominal bit rate**: ~0.7 Mbps

Observing these requirements and recommendations helps in assuring QoE while streaming a movie over the Mobile WiMAX network.

2.4.4 Features of Mobile WiMAX supporting video streaming

Among the features of Mobile WiMAX, the following two are important in offering video streaming services over the Mobile WiMAX network [1]:

- **High data rates**: Mobile WiMAX offers high data rates, peak data rates up to 63 Mbps on the DL and up to 28 Mbps on the UL, per sector in a 10 MHz channel. This is achieved by incorporation of multiple input multiple output (MIMO) antenna techniques, flexible sub-channelization schemes and advanced coding and modulation techniques. These connectivity speeds are adequate for users to upload and download video in real time, with each user using a small fraction of the cell resources. Hence, a large number of users in a given area can access multimedia services.
- **Quality of service (QoS):** QoS is deemed as a fundamental premise of the IEEE 802.16

MAC standard. For QoS provision, connection IDs and service flow IDs are used. The service flows can be mapped to DiffServ code points or MPLS flow labels in order to obtain end-to-end IP based QoS. There exist five service classes which have different levels of QoS and support various services such as video streaming, voice services in any form including VOIP, multimedia instant messaging, presence or Internet browsing, amongst others.

2.4.5 Challenges of video streaming over Mobile WiMAX

Although Mobile WiMAX offers support for video streaming, some challenges still exist due to the characteristic features of Mobile WiMAX. The challenges are briefly discussed [3, 20]:

- The data rate that a user obtains depends on the transmission channel conditions and consequently the modulation type that is maintainable. Switching modulation schemes results in a change in available bandwidth.
- When handoff occurs, the next base station may not have adequate resources to meet the demands of the newly joined mobile terminal, hence there can be drastic changes in available bandwidth.
- Signal strength decreases with the increase in distance between the base station and the mobile terminal. This also triggers change in the modulation scheme, from high density but less robust such as 64QAM to the more robust but low density schemes such as QPSK. Hence, this results in a decrease in available bandwidth.
- Mobile WiMAX offers QoS support. A scalable bit-stream can be streamed via different QoS service classes. For example the base layer can be assigned to the rt-PS service class while the enhancement layer is assigned to the nrt-PS. When congestion level increases within the Mobile WiMAX network, the enhancement layer may be affected.

Bandwidth fluctuations pose a serious problem to real-time video transmission over the WiMAX network. In this study, the bandwidth fluctuation problem is addressed without going into the specific cause of the problem.

2.4.6 Approaches used to address bandwidth fluctuation problem

In the literature, the following approaches were used to address the bandwidth fluctuation problem when streaming video over either the internet or wireless data networks:

- **Pre-coding:** This is a video adaptation method in which the streaming server switches between various pre-coded bit streams in order to meet the current requirements (bandwidth conditions, terminal capacity, etc). Multiple pre-coded bit streams have different quality, spatial resolutions, temporal resolutions, format, etc [9, 26] .
- **Trans-coding:** In trans-coding, a video signal is decoded and re-encoded to another coding format, bit-rate, spatial resolution, etc. Initially, the technique was used to map the bit rate of the video stream to the available channel capacity. However, lately, the technology supports heterogeneous devices, various frame rates, spatial resolutions and bit-rate adaptation of a video sequence [21, 27, 28]
- **Multiple description coding (MDC):** MDC encodes video into two or more independently decodable streams called descriptions. The descriptions are sent to the decoder over separate communication paths and in the event of one, or more, descriptions failing to arrive, the decoder uses the available descriptions to approximate the original signal. The quality of the received video is proportional to the number and size of the received descriptions. MDC allows video to be adapted, by providing the client with only the relevant descriptions required to achieve the desired quality, spatial resolutions and/or temporal resolutions [29, 30] .
- **Scalable video coding (SVC):** In SVC, a video is encoded into a scalable bit stream in which videos of lower qualities, spatial resolutions and/or temporal resolutions can be generated by truncating the scalable bit-stream. The scalability makes it easy to meet the bandwidth conditions, terminal capability, etc [9].

Below is an overview of work that has been done in an effort to minimize packet loss when bandwidth decreases. The advantages and disadvantages of each video adaptation approach are given.

S. Coulombe *et al.* compared pre-coding and trans-coding. They found pre-coding to have several advantages over trans-coding. The advantages include, lower processing requirements and no rights or author's approval issues. On the other hand, creating multiple representations requires a significant effort [28].

B. Chen *et al.* compared pre-coding and trans-coding in order to find the one appropriate for various IMS scenarios, such as link capacities, device heterogeneity and different formats. Trans-coding was found to be more suitable for IMS than pre-coding. The reason for this is due to the flexibility of trans-coding in adapting the video to any codec, spatial resolution, temporal resolution and/or bit-rate the client requires or the bandwidth affords. In contrast, the major disadvantage of pre-coding is a variety in codec, bitrates, etc makes it difficult to provide suitable bit-streams for all conditions. It is even more difficult in wireless networks where available bandwidth is dynamic [27] .

P. Xia *et al.* conducted a study on MDC over wireless data networks. They investigated how to assign bandwidth for descriptions in order to make full use of available bandwidth. The major disadvantage is that the minimum bandwidth has to be known prior to coding. Otherwise, if the bit rate of the descriptions is higher than the bandwidth on some parts of the network, the loss rate of the descriptions becomes high. On the other hand, if the descriptions have very low bit rates, the higher the number of descriptions required, which increases the coding cost [29].

G. Sun *et al.* investigated MDC in providing error-resilience in video transmission over wireless networks. The major advantage of MDC, according to the authors, is multi-path transmission, since it tolerates packet losses and delays due to network congestion [31].

D. Miras *et al.* compared pre-coding to scalable video coding. The authors give an account as to how the pre-coding can support dynamic switching between multiple streams, through synchronisation points in the bit-streams. The major disadvantage of the method is that it complicates the encoding process. In addition, the pre-coding requires extra space to store the multiple bit-streams and the number of available streams limits the granularity of the rate adaption [27].

The authors also mentioned the advantages of and disadvantages of scalable video coding. Advantages according to the authors are that scalable encoding allows the video bit-rate to be adapted to the network conditions by dropping or adding layers, as well as improving the reliability of video playback by prioritizing lower layers. Additionally, unequal error protection can be employed to protect the lower layers of the video bit-stream. Disadvantages include deciding what partition method to use, the efficiency cost, deciding how many layers to use and/or how to distribute the bandwidth among the layers [27].

J. Kim *et al*. proposed the use of SVC in Mobile IPTV services. The major advantage of SVC is can be applied in multiple scenarios such as multi-resolution content analysis, content adaptation, complexity adaptation, and bandwidth adaptation. The authors mention a disadvantage of SVC as the requirement of a relatively high complex encoder [32].

H. Sun *et al.* provided an overview of scalable video streaming. In their work, they compare SVC to pre-coding. They recommended SVC over pre-coding since it easier to scale SVC bit streams to the fast changing network bandwidth. The major disadvantage of SVC is the complexity of the decoder. However, the authors suggested that all four methods are not competing but complementing each other. The choice of which one to use should depend on the scenario at hand [9] .

2.5 Summary

The chapter presented the background information on Mobile WiMAX, SVC and video streaming. The focus on Mobile WiMAX was on both PHY and MAC layers as features that support video streaming. The key features of Mobile WiMAX supporting video streaming are QoS provisioning and high data rates. The challenges of video streaming over Mobile WiMAX were also presented and the major challenge was found to be varying transmission bandwidth.

On the SVC, an overview of the extension was presented as well as a discussion on the conceptual layers of the SVC and the modes of scalability including temporal, spatial and quality. Quality scalability, which allows the dropping of individual NALUs to scale down the bit stream to the available bandwidth, was covered in more detail. The base layer can be protected to guarantee a certain level of QoE to users.

A typical bandwidth adaptive streaming system was presented. The major components of the streaming system are the streaming server, the transport network and the receiving client. The streaming server streams the video and adapts it to the available bandwidth using one of the following methods found in the literature: pre-coding, trans-coding, SVC and MDC. Network monitoring is carried out by the streaming server and the receiver and the available bandwidth information is used to scale the bit stream.

Chapter 3 Pipelining-based Video Streaming Scheme

3.1 Introduction

The proposed pipelining-based video streaming scheme for Mobile WiMAX networks is presented in this chapter. The scheme consists of the following three activities: scheduling, network monitoring and video reception. The activities are shared between the video streaming server and the video client. The core of this work is on scheduling which is done by the streaming server using bandwidth signals sent by the video client to scale the bit stream to the available bandwidth.

The chapter begins by introducing the pipelining design technique as used in computer architecture and shows how the technique can be adopted to video streaming. Following this, the assumptions made when developing the scheme are presented, followed by a discussion of the design details of the proposed scheme with respect to the video streaming server and the video client.

3.2 Pipelining Design Technique

3.2.1 General concept

Pipelining is an implementation technique in which a number of instructions overlap in execution. The instructions are broken down into sub-instructions that are executed in parallel to save the execution time. The technique takes advantage of the parallelism that exists between actions required to execute the instructions. Today, pipelining is a key implementation technique widely used in computer processing units to improve their performance [33].

In pipelining, the idea is to execute more than one instruction concurrently .To carry out pipelining, the two approaches, parallelism or time-slicing are often used. In parallelism, a given task is broken down into a number of subtasks that are executed in sequence. Each subtask is performed by a given functional unit. The functional units are connected serially and they work simultaneously [33, 34].

Figure 3.1 illustrates the difference between parallelism-based pipelining and the traditional sequential execution of tasks. In the diagram, there are four functional units serially connected. The functional parts are: fetching (F), decoding (D), execution (E) and writing the results (W). Four instructions (Instr1, Instr2, Instr3, Instr4) are broken down are processed simultaneously. From the figure, the total time required to process the four instructions is seven time units when using pipelining as compared to sixteen time units in the case of sequential processing [33].

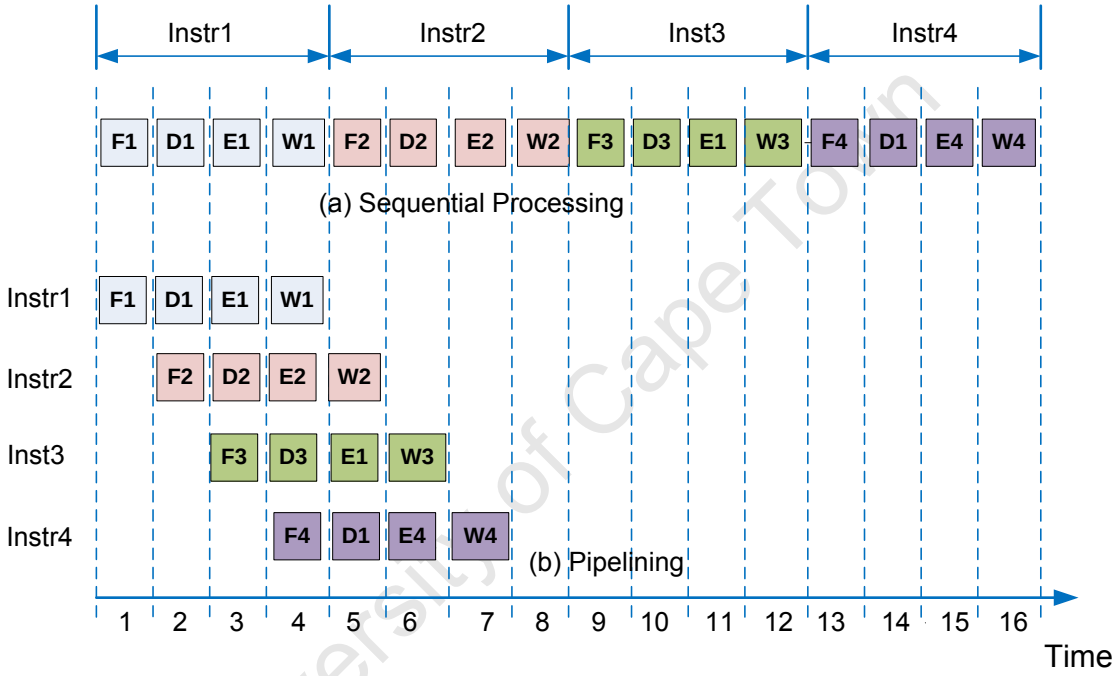

Figure 3.1 Pipelining versus sequential processing [33]

As mentioned, the other approach that can be used to carry out pipelining is time-slicing. In time-slicing, the focus is on processing a number of instructions in a time-multiplexing manner [34]. That is, a number of instructions are processed in a time period which is normally used to process a single instruction. This pipelining approach is used in this study. This is because the main objective of this study is not to save execution time (achieved through parallelism), but to enhance QoE by scheduling part of the network abstraction layer units (frame parts) that have more impact on video quality when available bandwidth is not sufficient for all frames.

Figure 3.2 shows video scheduling through the default scheduling mechanism, Earliest Deadline First and scheduling through pipelining. The four frames (I, P, B_1, B_2) are processed by each scheduler. Earliest deadline is shown first in Figure 3.2(a) where frame scheduling is done based on their deadlines or decoding order. In this method, each frame is scheduled within its own period. One of the challenges of this approach is that when bandwidth is not sufficient for delivering all packets, the packets get dropped by the network, which may result in drastic video quality degradation. Figure 3.2(b) illustrates how pipelining can be used in video scheduling so that in each period base layer NALUs are scheduled first, this helps refrain from scheduling enhancement NALUs (depending on their priorities) if available bandwidth is insufficient for all NALUs in a given scheduling plan. Another advantage of pipelining is that it helps in smoothing the SVC traffic.

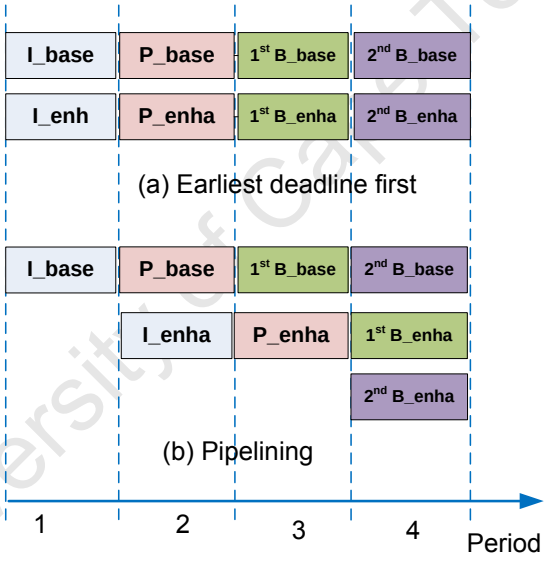

Figure 3.2 Pipelining in video scheduling

3.2.2 Pipelining performance evaluation

In computer architecture, the following three performance measures are usually used to measure the impact of the pipeline [33]:

- Speedup
- Throughput

- Efficiency

3.2.2.1 Speedup

Speedup determines the improvement of pipelining in terms of time units required to execute a given number of tasks. It is given by:

$$speedup = \frac{Time\ using\ sequential\ processing}{Time\ using\ pipeline\ processing}$$

In video streaming, video quality is more important than saving the execution time. This is because reducing the execution time does not have a direct relationship on quality of experience (QoE) of the received video. The aim is to send video frames in a manner that minimizes packet loss but optimizes QoE. Therefore, in this study speedup is not used, instead a video quality metric: peak-signal-to-noise-ratio (PSNR) is used to compare the performance of the earliest time and the proposed pipelining-based scheduling scheme.

3.2.2.2 Throughput

In computer architecture, throughput refers to number of tasks executed per unit time. The aim of pipelining is to increase throughput. However, in the proposed scheme, the idea is not to increase throughput but to minimise packet loss. Throughput in this case is not very important, as scheduling a large number of packets does not imply improved QoE as packets may be lost during the transmission process.

3.2.2.3 Efficiency

In computer architecture, efficiency refers to the ratio of actual speed-up to the maximum speed-up. This measure is also not used in this study for the same reasons as speedup. Instead, efficiency is considered in terms of bandwidth efficiency of the scheduling algorithm.

As mentioned in Chapter 2, SVC has a higher frame size variability compared to the preceding standards. One of the objectives of the proposed pipelining-based scheme is to reduce traffic variability. Therefore, the traffic profiles of the pipelining-based scheme and Earliest Deadline First (EDF) are compared.

3.3 Design Assumptions

When developing the proposed pipelining-based video streaming scheme, the following design assumptions were made:

I. During the streaming session, the available bandwidth is always greater than or equal to the minimum bandwidth. This is to allow successful transmission of the base layer at all times.

II. The base layer is protected by error concealment methods, such as forward error correction (FEC) in order to guarantee error free transmission.

III. A packet belongs to a single network abstraction layer unit (NALU). Hence, a dropped or lost packet directly affects only one NALU.

IV. All configurations required for video streaming have been done at the WiMAX core network.

V. The first packet of each pipelining cycle is delivered successfully. This is because this packet is used during network monitoring.

3.4 Design Topology

The network model assumed in this study is depicted in Figure 3.3. It is an end-to-end video transmission from a remote video streaming server to a subscriber stations in the Mobile WiMAX network. The network consists of the streaming server, subscriber stations, internet, and the Mobile WiMAX network access network. The streaming server and the Mobile WiMAX network are interconnected by the internet.

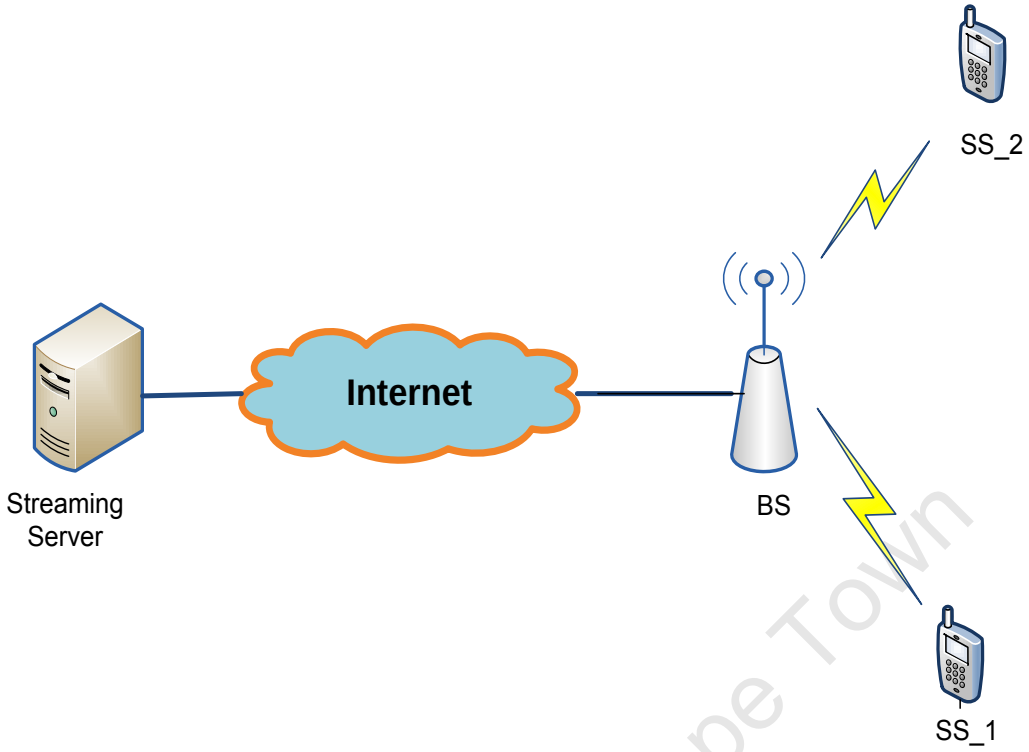

Figure 3.3 End-to-end video transmissions over WiMAX network

In Figure 3.3, the subscriber station SS_1, sends a request to the streaming server and the streaming server responds by delivering the video stream to SS_1. SS_1 estimates the available bandwidth periodically during the streaming session and sends the bandwidth information to the streaming server so that the streaming server can scale the video bit-stream to the available network bandwidth. Bandwidth information feedback helps the streaming server avoid flooding the network with video traffic which fails to arrive at the subscriber station, SS_1.

The focus of this work is on the streaming server and the subscriber station. This is because the streaming server carries out bandwidth video frames scheduling and bandwidth adaptation. Hence, the server hosts the proposed pipelined streaming scheduling scheme. The subscriber station receives video traffic, estimates available bandwidth and sends bandwidth signals to the streaming server. Design of the proposed video streaming scheme with respect to two network nodes is discussed in the next two sections.

3.5 The Streaming Server

The main function of the server is to stream stored video files over the Mobile WiMAX network to the receiving clients. The stored video files are encoded with SVC and the resultant video frames are delivered to the subscriber stations over the Mobile WiMAX network. However, before video frames are scheduled, they are broken down into network abstraction layer units (NALUs). The NALUs are assigned priorities based on their impact on received video. The NALUs are then buffered and later scheduled based on their priority and their parent frame index. The following three activities take place in the streaming server:

I. Priority planning for QoE optimisation

II. Server side NALUs buffering

III. Pipelining scheduler

3.5.1 Priority planning for QoE optimisation

As mentioned in section 3.1, the proposed scheme is based on the pipelining design technique. In pipelining, instructions are broken down into several sub-instructions and the sub-instructions are executed in parallel. In this case, video frames are broken down into NALUs. The NALUs are assigned priorities and are scheduled based on their priorities. Scheduling NALUs based on their priorities results in scheduling of NALUs from different frames in one period. This is in contrast to what normally happens in SVC video frames scheduling, where NALUs of one frame are scheduled in a single period. This subsection details priority planning for perceptual video quality optimisation.

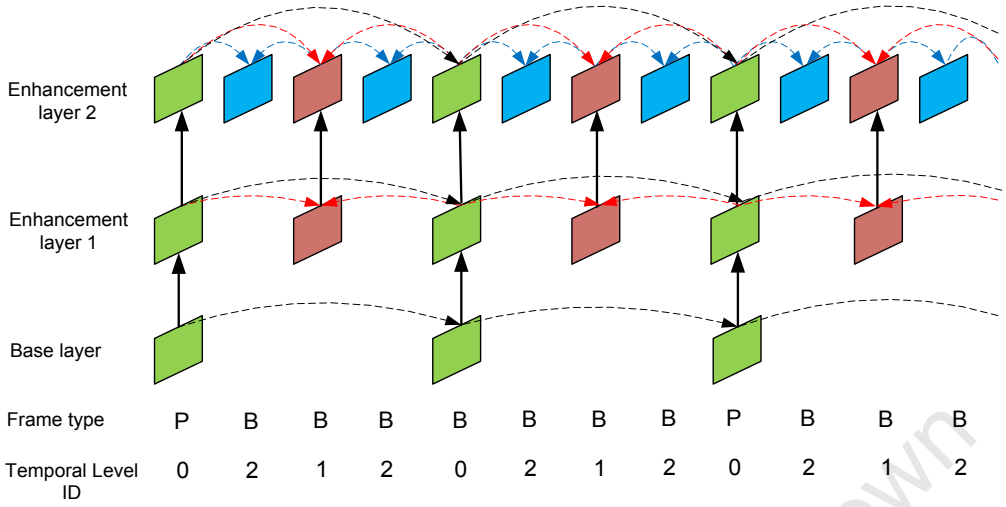

Figure 3.4 Hierarchical SVC bit stream

Figure 3.4 depicts an SNR video sequence encoded into three layers, a base layer and two enhancement layers. To recall, SVC uses temporal layers in both spatial and SNR scalabilities. In Figure 3.4, temporal level identifier (TID) 0 frames have NALUs in all layers. TID 1 frames have NALUs in layer 1 and layer 2. TID 2 frames have NALUs in layer 2. The arrows show the frames' predictions. In this study, we take advantage of frame segmentation that exist in SVC layered video to drop less important NALUs when bandwidth is not sufficient for transmission of the whole video sequence. We determine the priority of the NALUS as follows:

Consider an individual layer, L, consisting of m frames. L belongs to the bit-stream of n layers. The layers of the bits-stream can be represented in terms of the frames that make up the layer as follows:

L_0 , the base layer can be represented as:

$$L_0 = F_{01} + F_{02} + \cdots + F_{0m} \quad (1)$$

L_1 , the 1st enhancement layer can be represented as:

$$L_1 = F_{11} + F_{12} + \cdots + F_{1m} \quad (2)$$

Then the i^{th} layer is given by:

$$L_i = F_{i1} + F_{i2} + \cdots + F_{im} \quad (3)$$

Therefore, in general, the i^{th} layer can be expressed as:

$$L_i = \sum_{n=1}^{m} F_{in} \quad (4)$$

Where, F_{in} is the i^{th} layer NALU of the nth frame. Note that in the above equations, it is possible to have no representation of the k^{th} frame in one or more layers. This is demonstrated in Figure 3.4 where TID 1 and TID 2 frames do not have base layer NALUs.

We assign each layer a priority depending on its layer number as follows:

$$Priority(L_i) = i \quad (5)$$

From Eq. 4 and Eq. 5, it can be deduced that:

$$Priority(F_{in}) = i \quad (6)$$

Simply put, Eq. 6 means that the priority of the NALU is equal to the NALU's layer number. Eq. 6 defines what is referred to as the NALU's vertical priority.

A frame like a layer can be represented in terms of its NALUs that make up the frame. Following from Equation 1, the frame can be represented as follows:

$$F_i = \sum_{j=1}^{n} F_{ij} \quad (7)$$

Where: F_i is the i^{th} frame of the bit-stream.

n is a number of layers in the bit-stream.

F_{ij} is the j^{th} NALU of the i^{th} frame. A video frame is made up of one or more NALUs.

The relationship between the NALUs of an individual frame and the vertical priorities is illustrated in Figure 3.5. The frame consisting of k NALUs (from k layers) is mapped to k levels of priority (from equation 6). Therefore, NALU 0 has priority 0 and NALU 1 has priority 1 and so on. Priority decreases as we go up on the scale, that is, zero has higher priority than one.

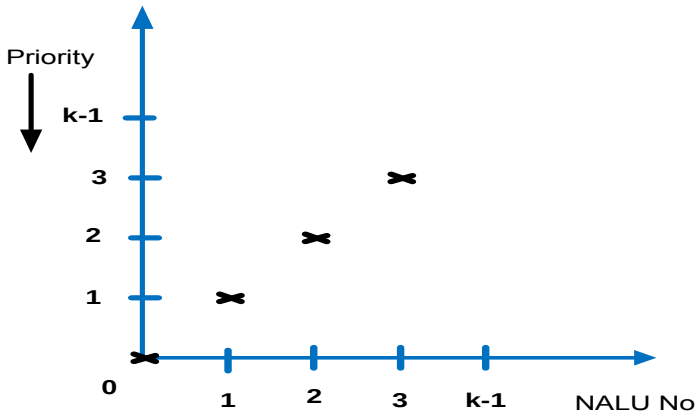

Figure 3.5 Frame vertical priority graph

However, for transmission reasons, the NALUs are broken down into equal size packets. Packets inherit priority of their NALUs and each packet contains one or more NAL units. The relationship between the packets and the priority of packets is depicted in Figure 3.6.

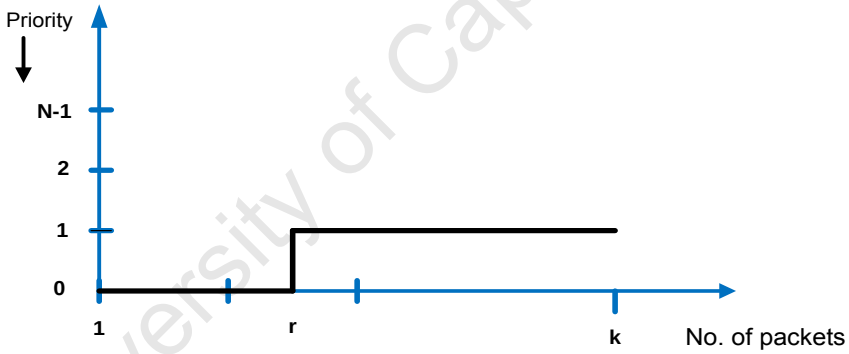

Figure 3.6 Vertical priority of the packets

In this study a two layered bit-stream is used. This is the minimum number of layers required to carry-out pipelining, more can be used but for proof of concept two layers will suffice. Hence, only two levels of vertical priorities exist, as shown in Figure 3.6. It also means that the frames of the bit-stream have at most two NALUs, for the two layers. In Figure 3.6, packets (1 to r) which are packets of the base layer NALU are assigned priority zero and packets (r+1 to k), packets of the enhancement layer are assigned priority one.

The priority that has been discussed so far is vertical priority. However, there exists what is referred to as horizontal priority. The importance of the packet is not only determined by the layer to which it belongs but also by the type of the frame. We refer to priority based on the

frame type as horizontal priority.

The horizontal priority assignment is done with respect to TID of the frame. Key frames (I/P) have TID of zero and all NALUs of these frames are assigned the horizontal priority of zero. B frames have different TIDs. NALUs belonging to TID 1 B frame are assigned priority of 1 and those belonging to TID 2 are assigned priority of 2 as shown in Figure 3.4. We use both horizontal and vertical priorities to determine the overall priority of the NALU.

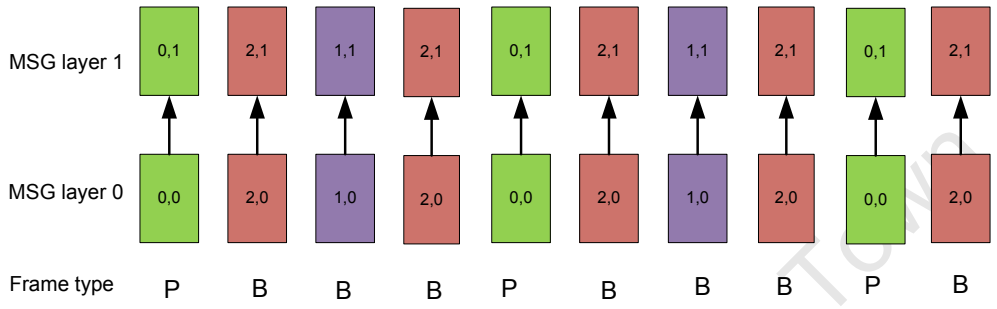

Figure 3.7 Horizontal and vertical priority coding

Besides the hierarchical representation, a video sequence can be represented in terms of quality layers (MGS) as shown in Figure 3.7. In this figure, NALUs belonging to the MSG layer 0 (QL_Id=0) have vertical priority of zero and those belonging to the MSG layer 1(QL_Id=1) have vertical priority of one. As for horizontal priority, TID 0 frames have priority 0, TID 1 frames have priority 1 and TID 2 frames have priority 2 (as shown in Figure 3.7). All in all, each NALU has both vertical and horizontal priority values. Table 3.1 summarizes the dual priorities used in this study:

Table 3.1 Dual priority summary

Frame TID	**Layer 0**	**Layer 1**
0	(0,0)	(0,1)
1	(1,0)	(1,1)
2	(2,0)	(2,1)

Using the priorities in Table 3.1, the following expression is utilised to compute the overall priority of an individual NALU shown in Table 3.2:

$$Priority(NALU) = 2 * TLId + 3 * QLId$$

Table 3.2 Overall NALU priorities

Priority combination	Overall priority
(0,0)	0
(1,0)	2
(0,1)	3
(1,1)	5
(2,0)	4
(2,1)	7

This section has discussed the priority planning for quality of experience optimisation. The next subsection details the NALU buffering process.

3.5.2 Server side NALUs buffering

After segmentation and priority assignments, NALUs are buffered. Buffering offers support to pre-scheduling activities such as traffic smoothing and forward error correction. It also allows planning for pipelining of video frames which is proposed in this study.

Buffering of the NALUs is carried out based on the TID of the NALUs. Four buffers are used, as shown in Figure 3.8. TID 0 NALUs are stored in buffer 1. TID 1 NALUs are stored in buffer 2. TID 2 NALUs are stored in buffers 3 and 4.The NALUs belonging two conservative TID 2 frames and are stored in separate buffers in order to allow interleaving of NALUs belonging to these frames.

The scheduler accesses buffers periodically and schedules NALUs according to their priorities and parent frame index. NALUs belonging to two adjacent buffers are scheduled in a single period. The scheduler makes a decision on which NALUs to schedule or drop NALUs using the algorithm discussed in the next subsection.

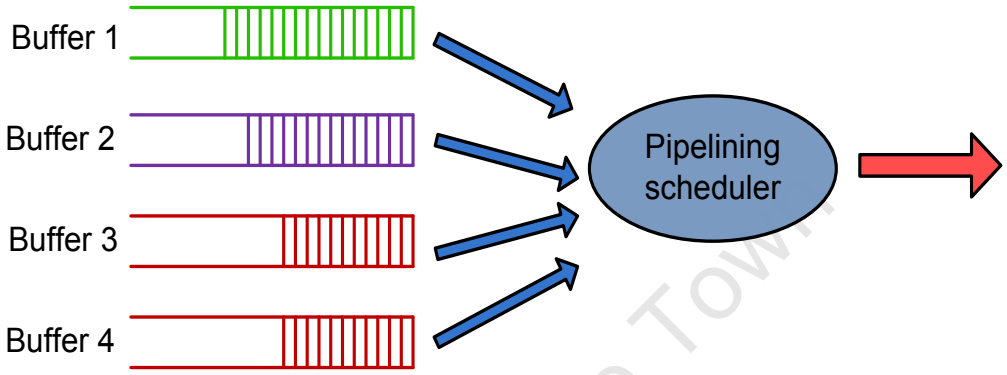

Figure 3.8 Video NALUs buffering

It is important to mention that packet buffering also occurs on the receiving side. On this side, buffering is important for compensating small time scale packet delay and packet delay variations. More details on buffering on the receiving side are discussed in details in subsection 3.6.1

3.5.3 Pipelining Scheduler

For transmission over the network, NALUs are broken down into equal size packets. Note that the packets inherit attributes of their parent NALUs, such as overall priority and frame index. The Scheduler determines the order and number of packets to be scheduled. As mentioned, the proposed scheduling of video frames is based on the pipelining technique. Hence, the name pipelining scheduler is given.

The scheduler utilizes two time granularities: frame period and the pipelining cycle. In a frame period, the scheduler sends packets belonging to two adjacent frames. The pipelining cycle is defined as the time frame in which the server schedules a given number of packets without

adjusting the scheduling rate.

The activity of the scheduler is represented by the Finite State Machine (FSM) diagram shown below:

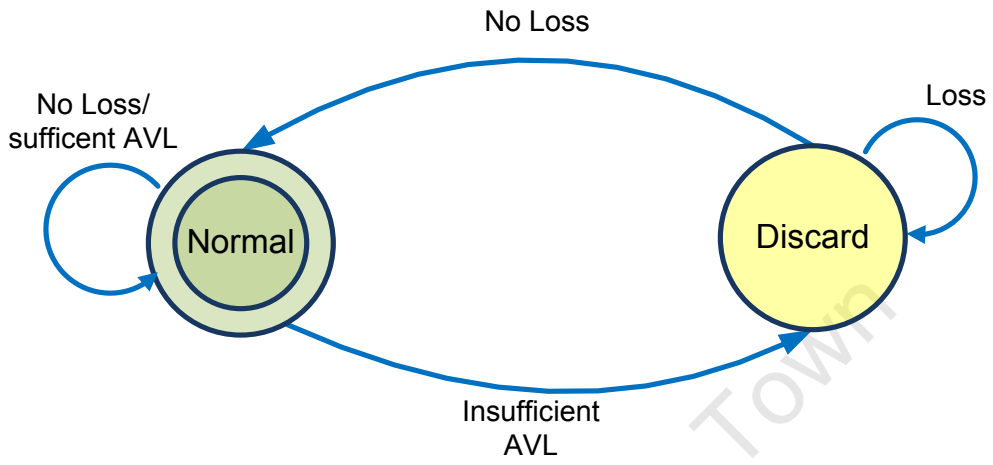

Figure 3.9 FSM diagram of the Scheduler

Key:

AVL: Available bandwidth

Table 3.3 shows the truth table which summarises the FSM of the scheduler. The table also shows the inputs that trigger transitions to different states.

Table 3.3 FSM Truth Table

Current state	**Input**	**Next state**
Normal	No loss/ Sufficient AVL	Normal
Normal	Insufficient AVL	Discard
Discard	No loss	Normal
Discard	Loss	Discard

The number of packets scheduled in a pipelining cycle, n, depends on the available

bandwidth, which is computed using information provided by the receiving node (section 3.5.2). An increase in available bandwidth results in an increase in the number of packets scheduled in the pipelining cycle. Also, a decrease in available bandwidth results in reduction in number of packets scheduled and hence, the sending rate decreases. Thus, the scheduler scales the bit-stream up or down and therefore adapts the bit-rate to the available bandwidth.

The proposed Pipelining Scheduler is illustrated in Figure 3.10 .The scheduler operates as follows:

- The scheduler receives the bandwidth information which it uses to determine the number of packets to be scheduled in the current pipelining cycle. In the first pipelining cycle, the scheduler sends all packets of the pipelining cycles to the receiver. The receiver estimates available bandwidth and sends bandwidth information back to the streaming server.
- Upon receiving bandwidth information, the scheduler checks if the available bandwidth has decreased in the previous pipelining cycle. If it did not decrease, the scheduler sends all the packets of the current pipelining cycle (referred to as *default* in Figure 3.10) then moves to the next pipelining cycle.
- If the bandwidth has decreased the scheduler checks if the available bandwidth is enough for scheduling all packets of the pipelining cycle and schedules all packets if capacity is enough. If it is not, the scheduler decreases the number of packets to be scheduled in accordance with available bandwidth (AVL). This is to minimize packet loss and by doing this the scheduler adapts the bit-stream to the available bandwidth.

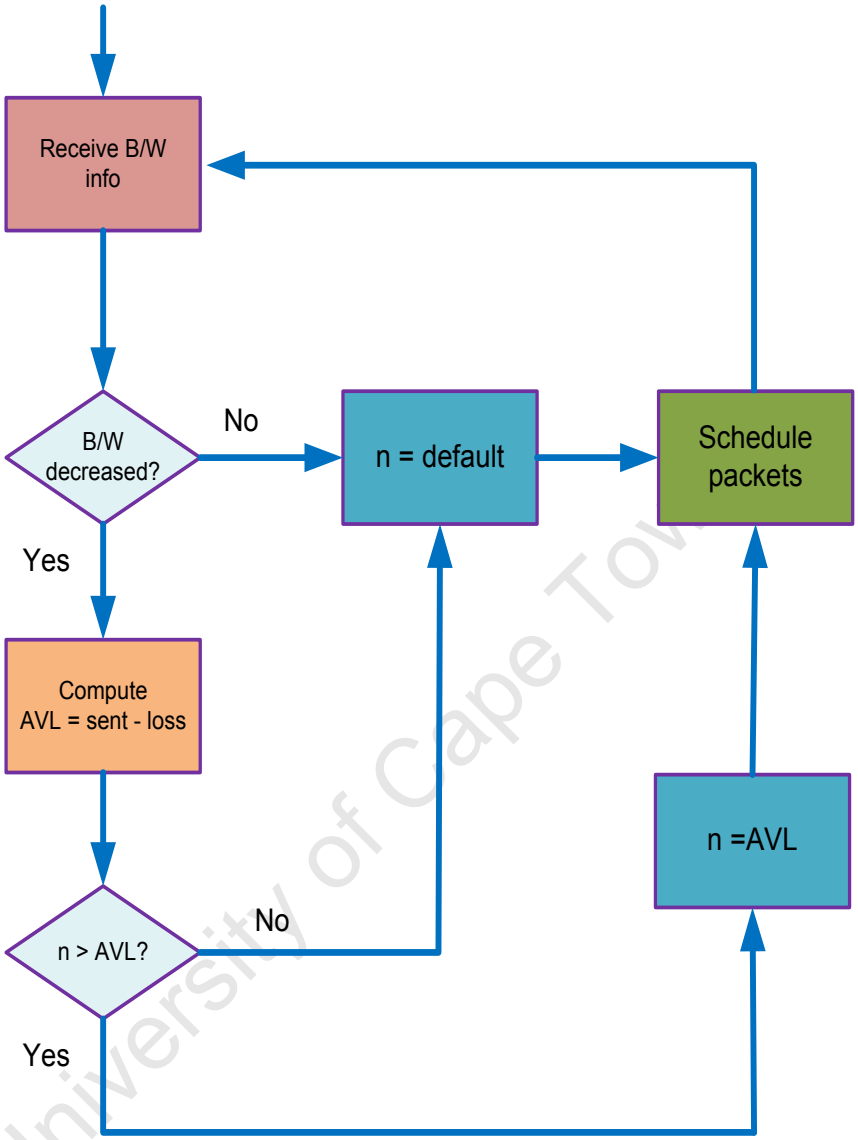

Figure 3.10 Pipelining Scheduler

Available bandwidth (AVL) is the number of packets that can be transmitted at a given transmission bandwidth. It is computed by subtracting the number of losses (P_loss) from the number of packets sent (P_sent) at a given pipelining cycle:

$$AVL = P\_sent - P\_lost$$

The actual scheduling of packets is done by the Pipeliner(the green block in Figure 3.10) shown in Figure 3.11.

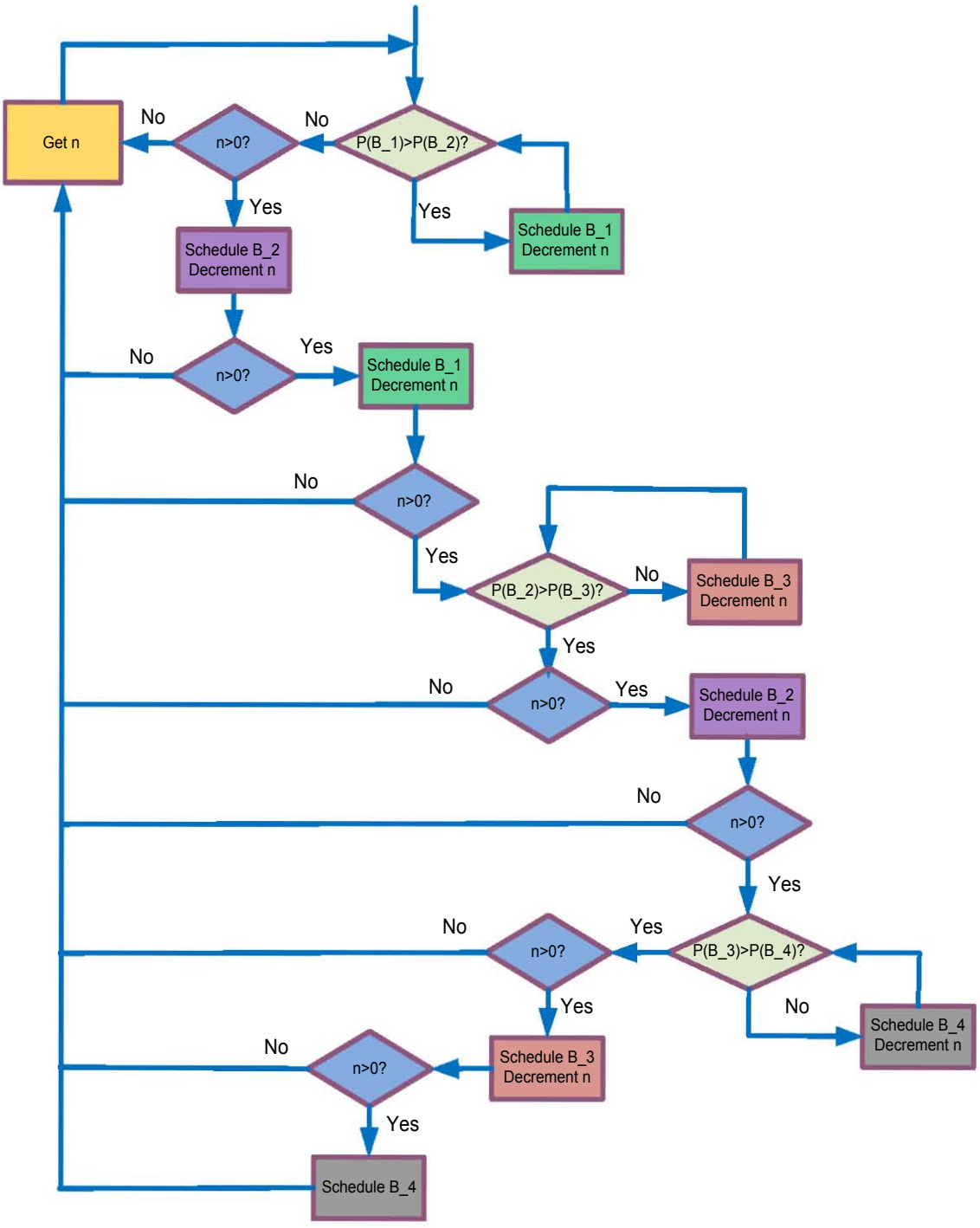

Figure 3.11 Pipeliner

The Pipeliner works as follows:

- The Pipeliner compares priorities of the in front NALUs in Buffer 1 and Buffer 2.
    - If buffer 1 NALU has higher priority, the NALU is scheduled and n is decremented. Priority comparison is done again between the NALUs in front of Buffer 1 and the Buffer 2 and the Buffer 1 NALU is scheduled if it has higher priority.
    - If buffer 2 NALU has higher priority, the bandwidth availability is checked (i.e. whether n>0?).
        - If bandwidth is insufficient the pipelining cycle is exited and all NALU which were supposed to be scheduled in the current pipelining cycle are discarded. The scheduler goes to the beginning and starts scheduling packets of the new pipelining cycle.
        - If bandwidth is sufficient the NALU from buffer 2 is scheduled and n is then decremented. The lower priority Buffer 1 NALU is also scheduled, followed by decrementing n and these will happen provided there is enough capacity. Otherwise, the pipelining cycle is exited.
- In the availability of bandwidth, the scheduler compares priorities of the first NALUs in Buffer 2 and Buffer 3. Then, the same procedure as in Buffer 1 and Buffer 2 NALUs' described above is repeated.
- Again if capacity is still sufficient, the Pipeliner compares priorities of the first NALUs in Buffer 3 and Buffer 4
    - Buffer 4 NALU is scheduled if it has higher priority and the comparison is done again between the next NALU in Buffer 4 and the front NALU of Buffer 3.
        - Bandwidth availability is confirmed and the Buffer 3 NALU is scheduled if it has higher priority. Otherwise the pipelining cycle is exited as illustrated in Figure 3.9.
        - Buffer 3 and Buffer 4 NALUs are then scheduled and prior to scheduling of each of them, a bandwidth availability check is carried out. Scheduling of Buffer 4 NALU completes the pipelining cycle.

3.6 Video receiver

The functions of the receiver include video packets buffering, packets re-ordering and re-combining packets to form frames. The frames are then decoded and displayed on the screen as video. The focus of this work on the receiving side is on the following two activities:

1. Packet buffering
2. Network monitoring

The two activities are in turn discussed in the next two subsections respectively.

3.6.1 Packet buffering

Due to the pipelining behavior of the scheduler, a frame is sent to the receiving device at two different periods. Simply put, a frame has part A (NALU belonging to the base layer) and part B (NALU belonging to the enhancement layer). These two parts of the frame are scheduled at subsequent periods. In the situation where packet delay is less than the frame frequency, the frame parts arrive at the receiving device at two different periods.

At the receiving side, packets are buffered before the decoding process can take place. Buffering is used to correct some of the network effects, such as small time scale delay variation and re-ordering of packets. Since, in this case parts of one frame arrive at different periods, buffering is needed so that frame parts (NALUs) can wait for each other and be re-combined prior to a decoding process.

When using the pipelining scheduler, part B of the frame is scheduled with part A of the succeeding frame and this makes part A wait for at least one period in the buffer for part B to arrive. If part A packets are lost during transmission, they can be re-transmitted by a re-transmission component (re-transmission is not part of this work and can be carried out by the special re-transmission module) provided they will reach the receiver within their playback deadline. Upon arrival of the corresponding part B of the frame, part A is pre-fetched and put together combined with part B. Part A undergoes buffering and pre-fetching which occur at different decoding periods.

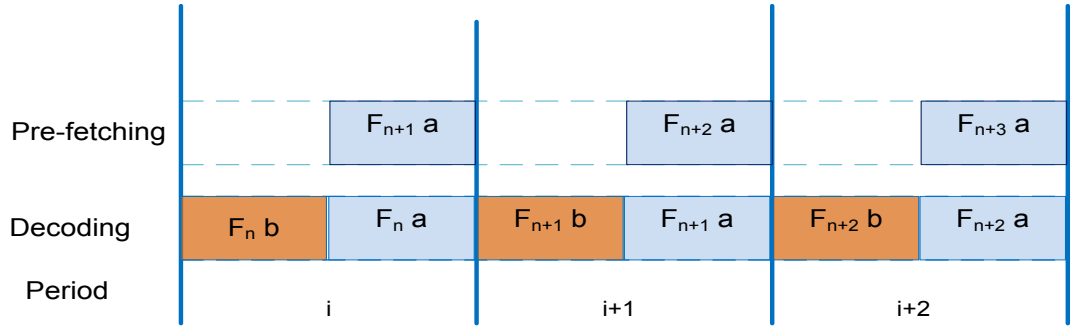

Figure 3.12 Two-stage pipelining graph

Figure 3.12 illustrates the pipelining process on the receiving side. At period *i* the lower priority NALU (part B) of F_n and F_{n+1} (part A of succeeding frame) are received. Part B that has just been received is combined with the pre-fetched part A which was probably received earlier so that decoding can take place. The procedure (arriving of part B and pre-fetching of part A of the frame) occurs in the next periods, i+1 and so on.

The proposed streaming scheduling scheme resembles pipelining in the following ways: in pipelining, instructions are broken down into sub-instructions and interleaved during execution; in this case, video frames are broken down into NALUs (sub-frames). The frames are interleaved during scheduling and at the receiving side they pass through pre-fetching and decoding stages which form a two stage pipeline graph (Figure 3.12).

Unlike in computer architecture, where the pipelining technique is instrumental in saving the execution time, in this case it is used to optimise quality of user experience while streaming video over the Mobile WiMAX networks.

This section has discussed the buffering process on the receiving side. The next section details the bandwidth estimation process.

3.6.2 Network monitoring

Bandwidth estimation is a broad research area on its own and substantial work [44] has been done in this field. The objective of this study is not to delve deep into bandwidth estimation research but just to develop a simple bandwidth estimation technique in order to prove the proposed bandwidth adaptive streaming scheme. Bandwidth estimation is essential because the proposed scheme requires bandwidth feedback in order to adapt the bit-stream to the

available transmission bandwidth. This section details algorithms that are used for bandwidth estimation in this study.

In order to effectively estimate bandwidth and scale the bit-stream to the available bandwidth, the following two algorithms are used:

a) Throughput estimation algorithm

b) Packet loss estimation algorithm

A. Throughput calculation algorithm

This algorithm computes throughput, then sends information about the bandwidth trend to the streaming server. In order to calculate throughput, the following throughput expression is used:

$$Thoughtput = \frac{packet\ \_size * (count\ +1)}{(t_n - t_1)} \quad (9)$$

Where:

t_n arrival time of the last received packet

t_1 arrival time of the first packet

count is the received packet counter

packet_size is the maximum packet size in bytes

Normally, the formula works by calculating amount of data received in a given time period. However, it was modified for MPEG traffic used in this work. Throughput is calculated per pipelining cycle and this is to ensure that throughput is estimated with some accuracy. Modification is done by tagging the first packet of the pipelining cycle with information indicating the number of packets to be sent at the current pipelining cycle. The receiver uses the tagged information together with the arrival times of the first and last received packet of pipelining cycle to compute the throughput.

Pipelining cycles are used to avoid averaging the throughput since MPEG frames are scheduled periodically and there exists silent periods between the subsequent pipelining cycles. The throughput calculation algorithm is depicted in Figure 3.13 and below is the summary of

how throughput is calculated.

- When the packet is received, it is checked whether it is the first packet in the group of packets that have been in the same period with it.
    - If it is the first packet, its arrival time is stored.
    - If it is not the first, it is checked if it is the last packet of the pipelining cycle:
        - If it is not the last, the packet counter is incremented
        - If it is the last packet, the arrival time of the packet is used to compute the throughput using Equation 9.

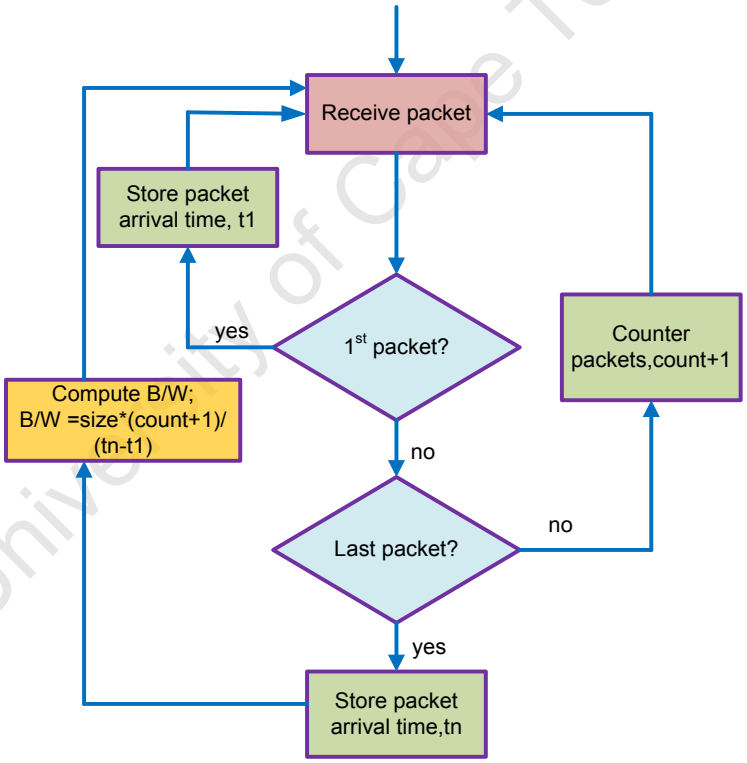

Figure 3.13 Throughput calculation

Since traffic that is being transmitted is VBR, change in throughput is not only related to network dynamics but to the nature of SVC traffic as well. This makes it difficult for the scheduler to adapt the bit-stream to the available bandwidth. Therefore, the second algorithm aimed at complementing this algorithm is proposed.

The second algorithm tracks change in transmission bandwidth with respect to packet loss. The algorithm is called loss estimation and it is briefly discussed below.

B. Loss estimation

The loss estimation algorithm tracks packet loss and informs the streaming server about the bandwidth status. The receiving device periodically (every 40ms) sends the bandwidth information signals to the streaming server. This is to allow fast adaptation to the fluctuating WiMAX bandwidth. In essence, the feedback signals contain loss flags and number of losses. Figure 3.14 depicts the loss estimation algorithm.

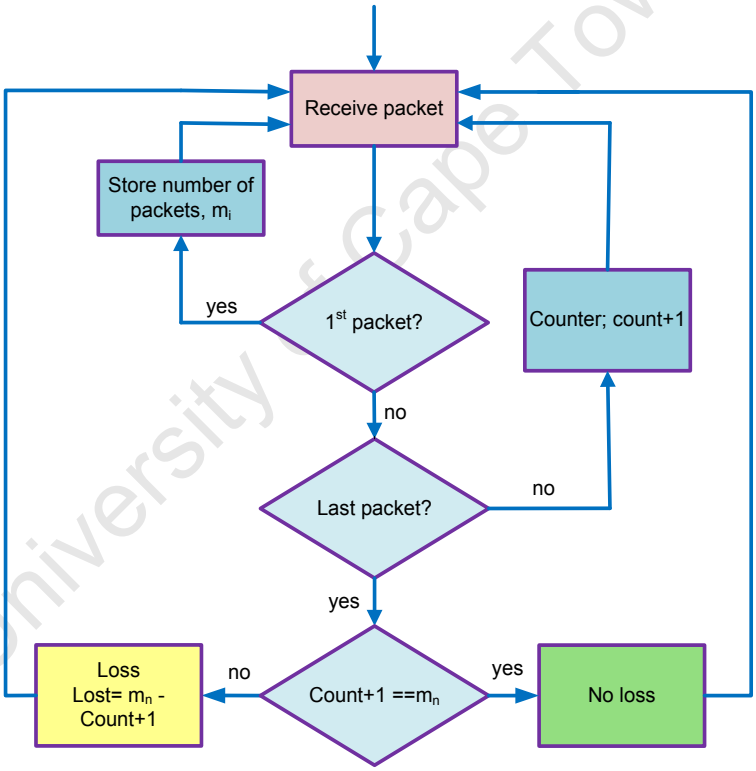

Figure 3.14 Loss estimation algorithm

The algorithm works as follows:

Upon receiving the packet the algorithm checks if it is the first packet. Recall that the first packet has a tag indicating the number of packets sent during the same pipelining cycle.

- If it is the first packet, the number of packets expected (obtained from the tag) is stored.

- If it is not the first, it means it is either the middle or the last packet. So the packet will be checked if it is the last packet. The packet sequence numbers are used to determine if it is the last packet.
    - If it is not the last, the packet counter is incremented
    - If it is the last packet of the pipelining cycle, packet loss is determined. Packet loss is determined by comparing the number of packets received to the number of packets that came with the first packet's tag. And if they are not equal, the number of packets lost is calculated and the loss flag is set to one. The streaming server uses this information to scale the video bit-stream accordingly.

Table 3.4 provides a summary of estimated bandwidth information contained in feedback signals that are sent to the streaming server periodically.

Table 3.4 Bandwidth information summary

Status	**Flag**	**Packets lost**
Loss	1	No. of lost packets (Lost)
No loss	0	0

3.7 Summary

This chapter has discussed the pipelining design technique as used in computer architecture. It also discussed how the technique can be adapted into video streaming. The technique coupled with network monitoring is used to design a scalable video streaming server for Mobile WiMAX networks. The main aim of the scalable server is to minimise packet loss while ensuring optimal visual video quality when available bandwidth decreases during a streaming session.

One of the advantages of the proposed scalable video streaming server is that the streaming system is bandwidth efficient. This is because the streaming server schedules video traffic based on bandwidth feedback signals to avoid flooding the network with traffic that will eventually get lost due to congestion. In addition, it saves bandwidth because if congestion occurs towards the end of the route, it means the video packets that get dropped by the network

have been consuming bandwidth that could have been used by other users.

Another advantage of the proposed scheme is its ability to smooth video traffic. There is high variability in SVC traffic due to its nature of being variable bit rate traffic, as well as its improved compression efficiency. The use of pipelining helps reduce burstiness during scheduling of the SVC traffic, thus making efficient use of the network bandwidth.

SVC is more suitable for video streaming over varying bandwidth networks, such as Mobile WiMAX. This is because in SVC, partially received frames are decodable as long as the base layer network abstraction layer units (NALUs) are completely received. Therefore, during scaling the streaming server ensures that it schedules a complete NALU to avoid scheduling traffic which will not improve received video quality as incompletely received NALU are discarded during decoding.

The next chapter presents the evaluation framework that was used and discusses how the proposed video streaming scheme was implemented.

Chapter 4 Experiment Implementation in NS2

4.1 Introduction

Chapter 3 presented the design details of the proposed pipelining-based video streaming scheme. This chapter discusses the implementation details of the proposed scheme. The chapter begins by introducing the objectives of the evaluation framework used to implement the proposed scheme. Then, the software tools that were used to construct the evaluation framework are discussed followed by a discussion of the implementation of the proposed algorithms in the respective nodes in the NS-2 simulator. Thereafter, the experiment procedure is presented.

4.2 Evaluation Framework Objectives

The key objectives of the evaluation framework are as follows:

- To determine whether it is feasible to use pipelining as a scheduling technique in video streaming in order to minimise packet loss while optimising perceptual video quality when transmission bandwidth decreases during an SVC-video streaming session.
- To evaluate the performance of the pipelining scheduler and compare it to the Earliest Deadline First video scheduling algorithm.
- To determine the distortion that a video streaming application is likely to experience when Mobile WiMAX network available bandwidth decreases. A suitable framework must adequately model a real-world scenario that a streaming application, say Movie streaming, encounters.

4.3 Software Tools

The following software tools are used in this work: video coding tools, network simulation tools and video evaluation tools.

4.3.1 Video Coding Tools (JSVM 9.18 Software)

The Joint Scalable Video Model (JSVM) has been used as a video coding tool in this study. JSVM is a software tool-kit that is used for encoding and decoding SVC video. It is a deliverable of the project by the Joint Video Team (JVT) of the ISO/IEC Moving Pictures Experts Group (MPEG) and the ITU-T Video Coding Experts Group (VCEG) [19].

The software is written in C++ and can be built on either Windows 32/64 bit or Linux 32/64 bit platforms. In this study, it was built on an Ubuntu 64 bit platform. The software consists of a number of tools. However, only the following three were used [19]:

i. H264AVCEncoderLibTestStatic
ii. H264AVCDecoderLibTestStatic
iii. BitStreamExtractorStatic

The tools are briefly discussed below and sample commands are provided.

i. H264AVCEncoderLibTestStatic: is an encoder which is used to generate AVC and SVC bit-streams. The encoding mode is specified by the parameter *AVCMode* inside the main configuration file. When *AVCMode* is not present or equal to zero, the encoder is run in scalable coding mode. Otherwise, the encoder is operated in single-layer coding mode [19].

Usage:

H264AVCEncoderLibTestStatic –pf <mcfg> [command line options]
Mcfg: represents the filename of the main configuration file

To encode video into an SVC bit-stream, an encoder uses a number of configuration files: the main configuration file and the configuration files of each layer. The configuration files consist of a number of configuration parameters with each parameter specified in one line. Each of the parameter has a default value, and when the parameter is absent in the file, the default value is used [19]

Some of the key configuration parameters used when encoding video in this study and their corresponding values are given in Table 4.1.The complete configuration files can be found

in the software folder accompanying this document.

Table 4.1 SNR encoding configuration parameters

Parameter	Value	Description
FramesToBeEncoded	300	Number of frames
CgsSnrRefinement	1	SNR refinement, 1:MGS
GOPSize	4	GOP size
BaseLayerMode	1	Base layer mode,1:AVC compatible
NumLayers	2	Number of layers
FrameRate	25	Displayed frames per second

ii. **H264AVCDecoderLibTestStatic:** is a decoder and is used for decompressing both AVC and SVC bit-streams depending on the associated command line options.

Usage:

H264AVCDecoderLibTestStatic <str> <rec> [-ec <ec>]

str: bit-stream file (input)

rec: reconstructed video sequence (output)

ec: error concealment method (1-3)

iii. **BitStreamExtractorStatic:** this tool is used to extract sub-streams of an AVC or SVC bit stream. The sub-streams represent bit streams with a reduced spatial and/or temporal resolution and/or a reduced bit-rate.

Usage:

BitStreamExtractorStatic [-pt trace] <in> [<out> [options]]

Options:

-pt trace: generates a packet trace file from the given stream

-sl SL: extract the layer with layer id = SL and the dependent lower layers

4.3.2 Network Simulation Tools

Among available network simulators, Network Simulator-2 and OPNET (Optimized Network Engineering Tool) are the two simulators suitable for modelling and analysis of the proposed video streaming scheme. The major advantage of the two simulators over others is that they both have WiMAX modules. In addition, the two event-driven network simulators have been used extensively in industry and academia [24]

NS-2 is a simulator developed in C++ and TCL scripting language. The front-end of the simulator is developed with TCL for easy configurations and C++ is used at the back-end for efficiency [35]. NS-2 has extensively been used in academic networking research. One of its major advantages over other simulators is its open source development. Hence, it is very extensible and allows users to either alter existing protocols or create new ones. In addition, NS-2 is well documented and has online support [35].

OPNET offers a comprehensive development environment for the specification, simulation and performance analysis of communication networks [36]. For this reason, OPNET is a leading network simulation service provider in the market. In OPNET, a large range of communication systems from a single LAN to global satellite networks are supported [36].

However, in this work, NS-2 is used. The major reason for choosing NS-2 was due to its support for SVC-video and WiMAX as well as the availability of QoE evaluation tools for received video. The NIST mobility module for NS-2 [37] and EvalSVC module [12] were added to the NS-2 for support of WiMAX and SVC video, respectively.

4.3.3 Video Evaluation Tool-set

EvalVid is a framework that provides tools for the evaluation of video that has been transmitted over a physical or a simulated communication network. Researchers have used the framework intensively in order to evaluate their video transmission network designs. Video evaluation is done in terms of user perceived video quality in addition to other network performance metrics, such as packet loss, frame loss, and end-to-end delay [24, 24, 38].

EvalVid takes a video input, transmits it over the network and produces received video with errors and defects that happened during the network transmission. By comparing the

received to the transmitted video, the framework is able to provide visual quality evaluation. However, if degradation is too small to be evaluated using subjective measures (user-perceived quality); the tool-set provides the means to evaluate the received video using other objective video quality measurement metrics such as the peak-signal-to-noise-ratio (PSNR) [38].

The major challenge of Evalvid version 2.7 is that it does not support SVC video. However, the support for SVC video is achieved through an extension of EvalVid called EvalSVC. Figure 4.1 illustrates an integration of the EvalSVC framework into NS-2 simulator. In the figure, the video encoder and decoder blocks represent the encoding and decoding processes carried out by JSVM software described in subsection 4.3.1. The enclosed area denoted as the NS-2 environment represents the network and the green blocks are the EvalSVC video evaluation tools.

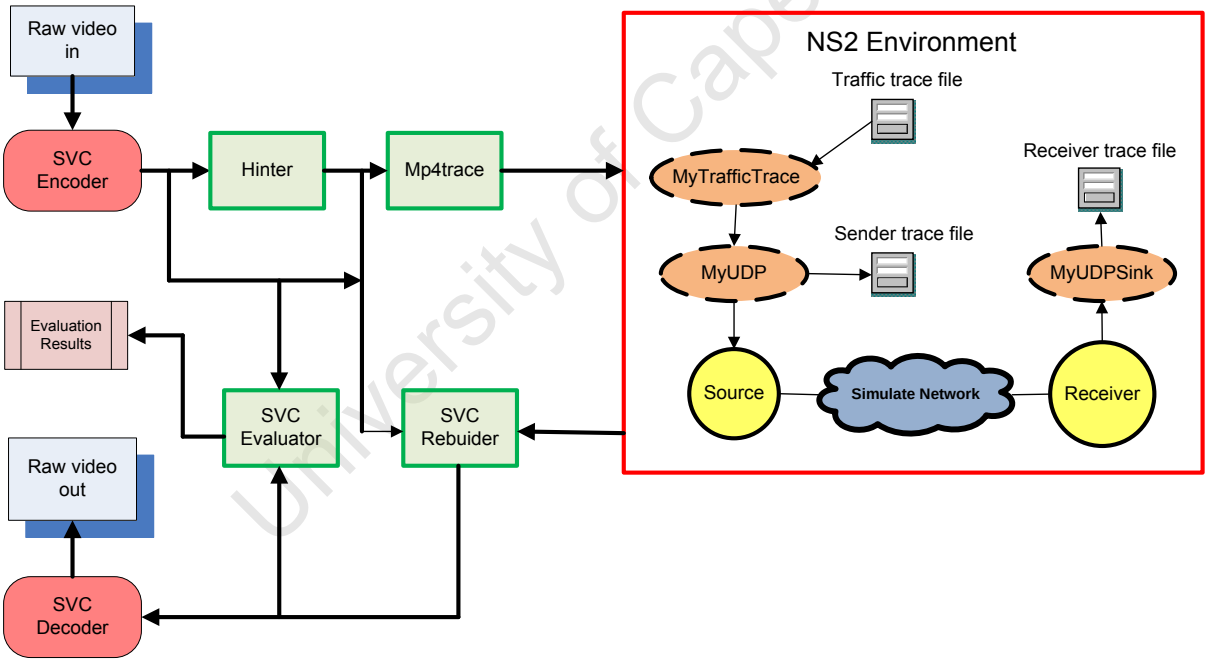

Figure 4.1 Integration of EvalSVC into NS-2 [12]

4.3.3.1 EvalSVC Tools

Below is a brief overview of video evaluation tools that are used in this study [12]:

- Hinter: The MP4Box tool from the GPAC library [39] is used as a hinter. The main function of this tool is to create ISO MP4 files containing the SVC NALUs and a hint

track, which describes how to packetise the frames for RTP transport.

- Mp4trace: This component acts as a video sender. The tool is used to send the hinted SVC bit-stream out to the network using the packetisation information that was produced by the Hinter. It logs (into a trace file) the sequence numbers, types, and sizes of the video frames, and the number of UDP packets required to transmit each frame. The resulting trace file is used in NS-2 to generate SVC traffic.
- SVC Re-builder: This tool is the core of EvalSVC tool-kit. The main role of the SVC Re-builder is to reconstruct video. The tool uses sender trace file, receiver trace file, traffic trace files and the hinted file (.mp4 file) to reconstruct a possibly-corrupted SVC bit-stream at the receiver. When encountering a missing packet, or a missing frame, the SVC re-builder truncates the SVC video frame. The tool also generates files which can be used to compute QoS metrics, such as end-to-end delay, jitter, and loss rate.
- SVC Evaluator: The tool compares the re-constructed video to the reference video then produces QoE results. The tool generates the objective and subjective quality evaluation (PSNR and MOS respectively) metrics of the reconstructed SVC video.

4.3.3.2 NS-2 Environment

Within the NS-2 Environment shown in Figure 4.1, there are five NS objects of the following type: traffic generator, transport agent and node. The three object types are briefly described below [12]:

- Traffic Generator: The traffic generator used is called MyTrafficTrace. It generates SVC traffic according to the information contained in the video source trace file generated by the Mp4trace tool. MyTrafficTrace schedules video frames (as packets) based on their encoding order. The packets are sent from MyTrafficTrace to the transport agent for transmission over the network.

    - Transport Agent: Transport agents used are MyUDP and MyUDPSink. MyUDP sends traffic generated by MyTrafficTrace to the corresponding receiving side agent, MyUDPSink. When sending the packets, MyUDP logs (into a trace file as well) the sending time, sequence number and the packet size of every packet. When MyUDPSink receives the packets sent by MyUDP it logs the arrival times, sequence numbers and the

packet sizes. The logs are stored in a receiving trace file. The SVC Re-Builder when re-constructing the received video uses the trace files.

- Node: The nodes, the source and the receiver send and receive traffic from the low-level network, respectively.

4.4 Implementation of the proposed algorithms in NS-2.

4.4.1 Video streaming server

To implement the proposed scheduling scheme on the streaming server, the default traffic generator (MyTrafficTrace) and agents (MyUDP) were modified as follows:

i. MyTrafficTrace traffic generator which scheduled frames based on their encoding order was modified. The resulting traffic generator is called Pipelining_TrafGen.

ii. MyUDP agent was modified to support the Pipelining_TrafGen. The resulting agent is called Transport_Agent.

Figure 4.2 illustrates the proposed video streaming server model implementation in NS-2. The server has three levels: application, agent and low-level network. Application level is where the traffic generator resides. Analogous to other applications, such as CBR, Pipelining_trafGen sits on the top level and models user demands to transmit SVC video traffic periodically. The demand is passed to the intermediate level, which in this case is the Transport_Agent.

The Transport_Agent acts as a bridge that connects an application to the low-level network. It also constructs the UDP packets based on the user demand provided by an application, then stores source and destination IP addresses and transport layer ports in the packet header, and forwards the packet to the attached node. The node puts the packet into the low-level network for delivery to the video receiver.

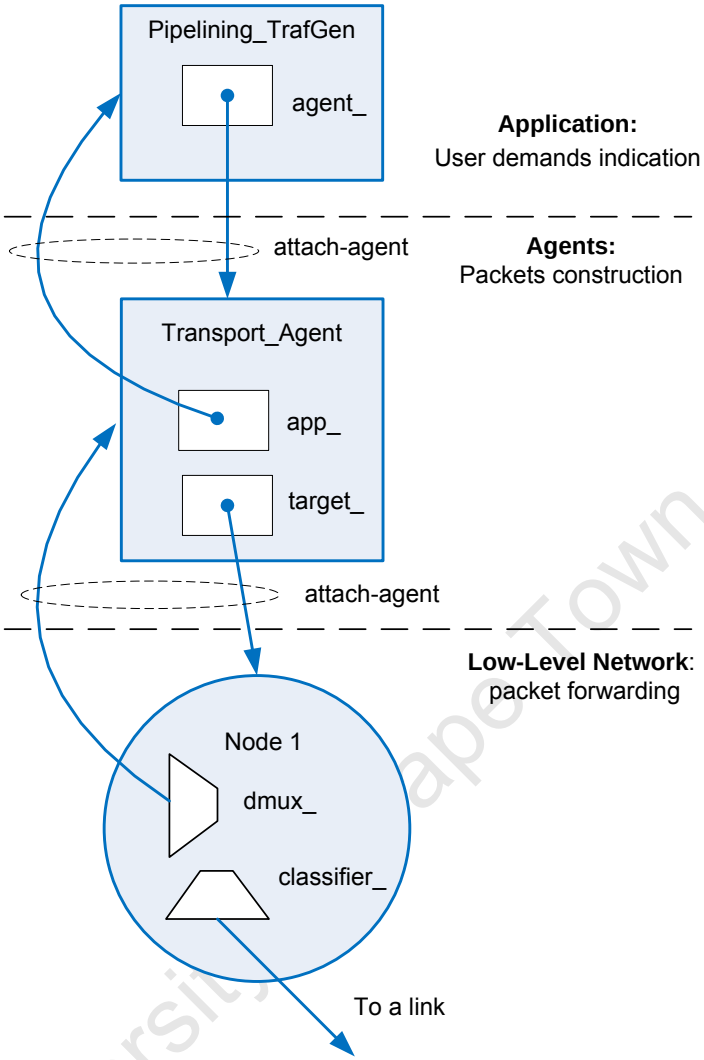

Figure 4.2 Video streaming server model

The next subsections discuss the Pipelining_TrafGen and the Transport_Agent. The proposed pipelining-based scheduling scheme was implemented using C++ in the Pipelining_TrafGen.

4.4.1.1 Pipelining Traffic Generator (Pipelining_TrafGen)

Pipelining_TrafGen generates SVC traffic according to the given trace file. The trace file contains a series of inter-burst transmission intervals and payload burst sizes, frame type, frame size and the number of RTP packets required to transmit the frame. The pipelining scheduler, presented in chapter three, has been implemented within the Pipelining_TrafGen.

The scheduling algorithm is represented by the flow chart shown in Figure 4.3. The flow

chart shows how the scheduling algorithm, proposed in Chapter 3, was implemented in NS-2.

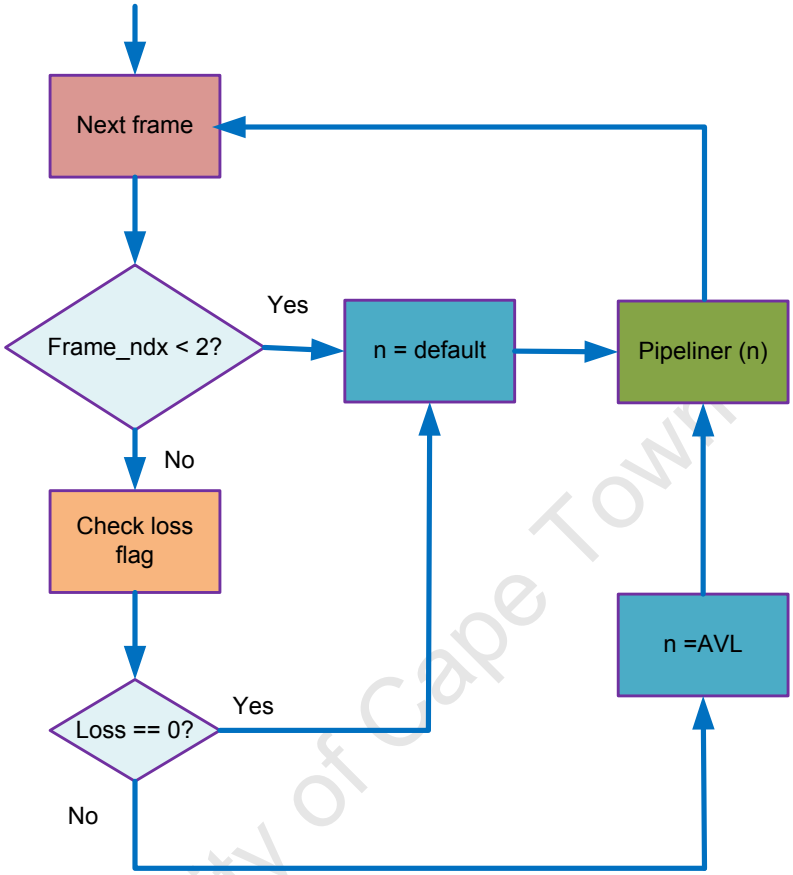

Figure 4.3 Pipelining scheduler

The pipelining scheduler works as follows:

- It receives the bandwidth information and the current pipelining cycle information. The bandwidth information is used monitor the servers sending rate so that bit-stream is scaled to the available bandwidth.
- For the first two periods, the scheduler does not use bandwidth info to schedule the packets. This is because we have assumed that it takes at most 80ms for bandwidth feedback to reach the streaming server.
- Thereafter, the scheduler checks the loss flag to determine if there has been loss. In the absence of loss, all packets of the current pipelining cycle are scheduled. Otherwise,

packets in accordance with the number of previously received packets, n, are scheduled.

- The scheduling of the packets is carried out by the function called Pipeliner (a green block in Figure 4.3). The details of this function are discussed below.

The Pipeliner function

The function used the following variables:

n: number of packets to be sent in a pipelining cycle. It can be considered as the "sending quota". The value of n is directly proportional to the available bandwidth and its minimum value at the start of each pipelining cycle is equal to Iy, that is, the number of packets in the base layer NALU. Therefore, at the lowest value of the available bandwidth, only the base layer packets are scheduled in the pipelining cycle. n equals to zero signifies the end of the pipelining cycle.

Iy: number of packets constituting the I-frame base layer NALU. This value differs with each I-frame but in this study, for simplicity, it is assumed to be the same for all frames. To ensure that the base layer packets are never missed, the size of the largest NALU has been used to compute Iy.

Pe and **Be:** number of packets constituting the medium granular scalability layer 0 NALU for P-frame and B-frame respectively. As in the case of Iy, the values of Pe and Be are not accurate, as the largest NALUs were used to determine the variables in both cases.

The values of Iy, Pe and Be varies for each video sequence, hence a different set of values was computed for all video sequences used.

Count: the number of segments used to transmit the complete frame. It is computed by dividing the frame size by the segment size.

C++ functions used by the Pipeliner function

Table 4.1 describes the functions used by the "pipeliner" function. The detailed descriptions are attached in Appendix A.

Table 4.1 Pipeliner supporting functions

Function	Description
Sendmsg(nbytes) **e.g:** **send Iy packets**	• Send a message with "nbytes" bytes • Informs attached transport layer agent about user's demand • nbytes is data payload (250+overhead)
get_next(ndx,trace) **e.g:** **Move to next frame**	• Takes frame index and trace as inputs • Retrieves relevant information of the frame pointed by the index
next_interval(size)	• Takes payload size "size" an input, returns delay time after which a new payload is generated • Invokes the get_next( ) function to get attributes of the frame

The operation of the pipeliner function has been represented by a flow chart as well. The flow chart of the function has been divided into three parts: Pipeliner (when frame type=1), Pipeliner (when frame type=2) and Pipeliner (when frame type=3). Each part is presented in its own page due to the limited page space.

The function received frame attributes, such as frame index, frame type, and frame size. Then it checks the frame type (I=1, P=2, and B=3). If frame type is equal to one, the function continues on the Pipeliner (when frame type=1), that is the flow chart in Figure 4.4. If the frame type is equal to two, the function goes to Pipeliner (when frame type=2) flowchart part shown in Figure 4.5. If the frame type is equal to three, the function goes to the Pipeliner (when frame type=3) part of the flow chart shown in Figure 4.6.

The various blocks of the flow chart are described below:

- **Load preceding frame**: the frame index is decremented, then get_next() function is called and the attributes of the frame are obtained.
- **Send packets**: the sendmsg(nbytes) function is called, thus sending packets to the transport agent. Then, n is decremented by the number of packets that have been sent.

- **Move to next frame**: In this process, the frame index is incremented. The index is required by the **Load preceding frame** to obtain the frame attributes.
- **Reset n**: sets n to zero, to allow exiting of the current pipelining cycle.
- **(n=>count – Iy) or(n=> count – Pe) or (n=>count – Be):** The conditional tests are carried out to ensure that a complete NALU is scheduled. This is because incompletely received NALUs are discarded. Hence, they do not add to the quality of the received video and are a waste of the channel bandwidth.
- **Next pipelining cycle:** is an exit to the current pipelining cycle. It gives control back to the main function, the Pipeliner.

It is important to note that as proposed, the priorities are used to determine which NALUs are scheduled or dropped. However, in the implementation, the priorities are not explicitly shown. This is because the frames from the trace file come in order, and the scheduler is able to determine the number of packets to schedule using the bandwidth information and the frame type. The diagrams illustrating the pipeliner follow:

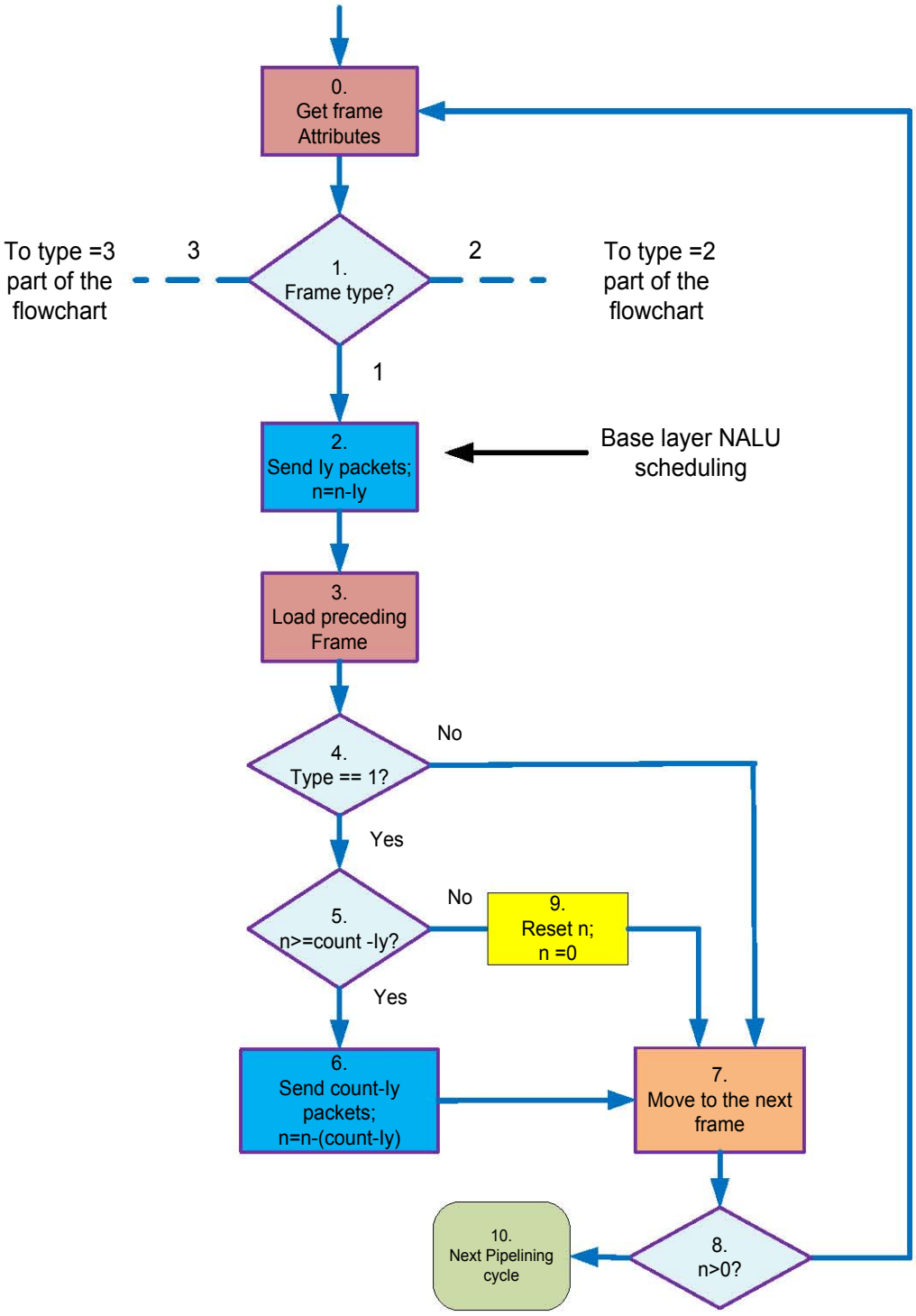

Figure 4.4 Pipeliner (when frame type=1)...

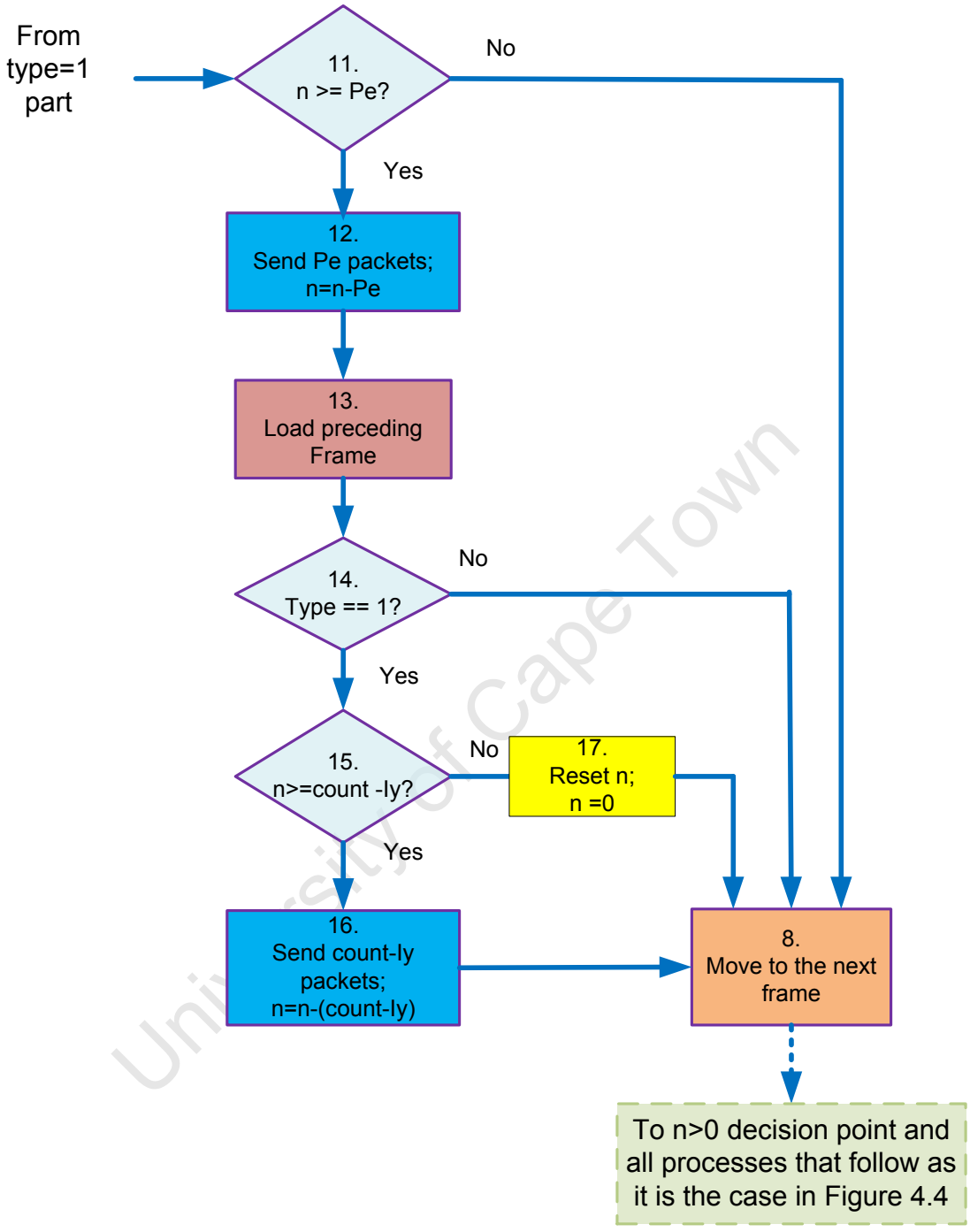

Figure 4.5 Pipeliner (when frame type = 2)...

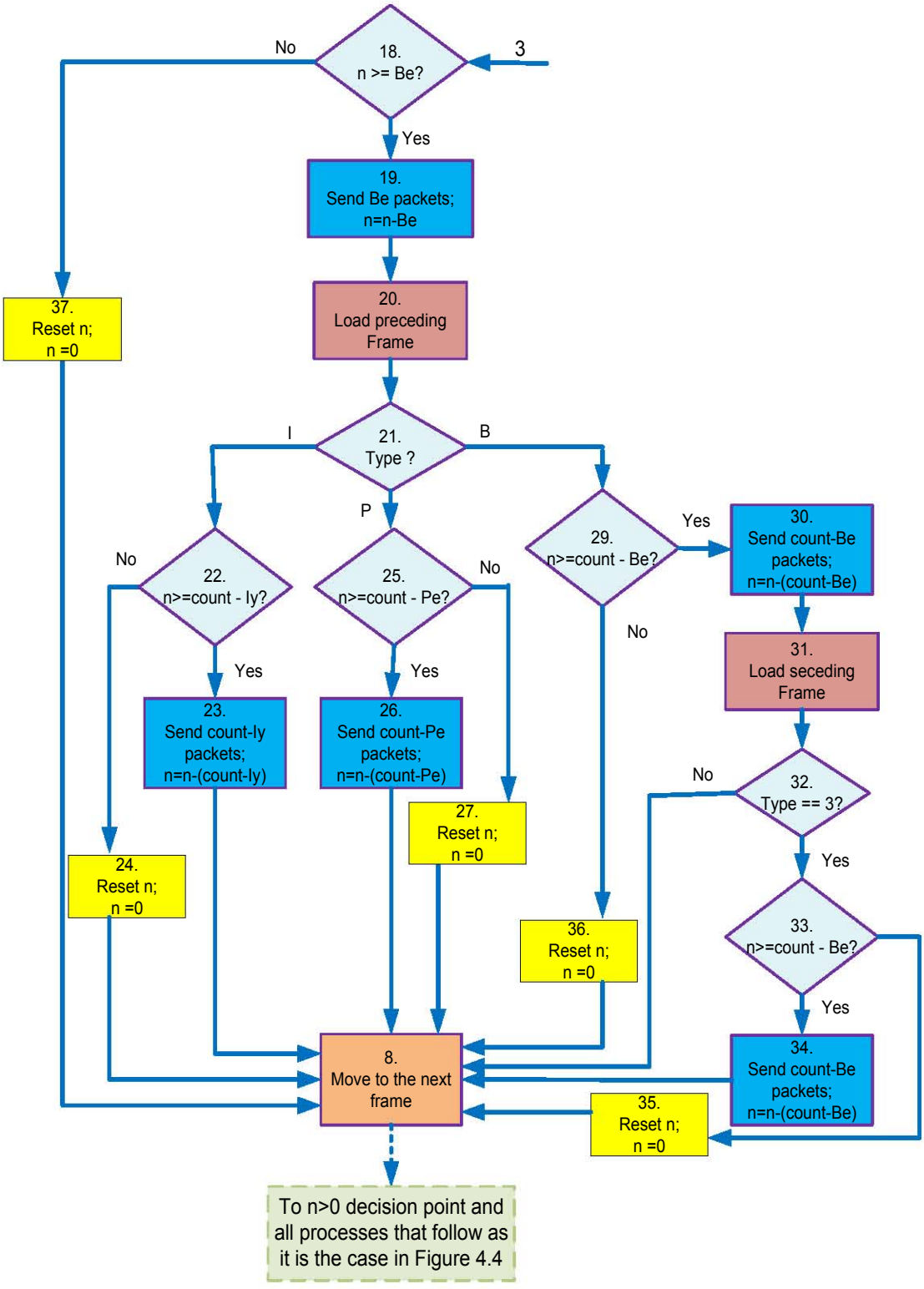

Figure 4.6 Pipeliner (when frame type = 3)

4.4.1.2 Transport agent

The major changes made in the Transport_Agent concern the of recording the number of packets sent to the lower level network during each pipelining cycle. Packet marking is done in this agent in order to allow identification and ordering of packets belonging to individual frames. Packet marking is important because the pipelining scheduler sends packets of the frame out of the normal order due to the use of priorities. The frame ID was used for the identification of packets belonging to the same frame.

4.4.2 Video receiver

To implement components of the proposed scheduling scheme on the video receiver, the receiving agent (MyUDPSink) was modified. As mentioned in subsection 4.33, the major responsibility of MyUDPSink is to log received packets information (arrival time, sequence number and packet size) into a trace file, and to de-allocate received packets. The agent was modified to provide the means of estimating available transmission bandwidth. The resulting agent is called the Receiving_Agent.

The Receiving_Agent implemented the proposed bandwidth estimation algorithms. The algorithms estimated available bandwidth and sent feedback signals to the streaming every 40ms. The proposed network monitoring algorithms were implemented without any modifications. Refer to subsection 3.6.2 for details of these algorithms.

Figure 4.7 shows an overview of a video receiver model in NS-2. Unlike the streaming server (Figure 4.2), the receiver does not have the traffic generator object since it does not generate any traffic. Trace files produced by the receiver were used to carry out performance evaluation.

Since the pipelining scheduler sent packets of the frame out of order, packet re-ordering was required prior to video reconstruction and decoding. Packet re-ordering, frame decoding and video playback processes were executed offline.

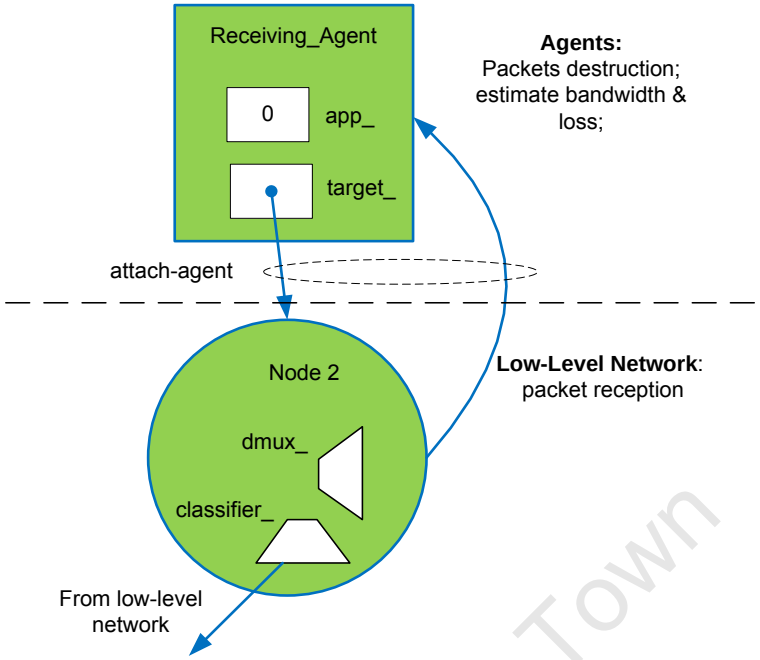

Figure 4.7 Video receiver model

4.5 Simulation configurations

4.5.1 Network Model

The simulated network model consisted of six network nodes. The nodes were a streaming server, sending SVC video to a receiving subscriber station (SS_1) in the WiMAX network. The CBR traffic source and its corresponding receiving subscriber station (SS_2). The other nodes were the WiMAX base station and the router - interconnecting the traffic sources and the WiMAX network. SVC video was the traffic of interest in this study while CBR modelled the cross-traffic.

The simulated network model is illustrated in Figure 4.8.

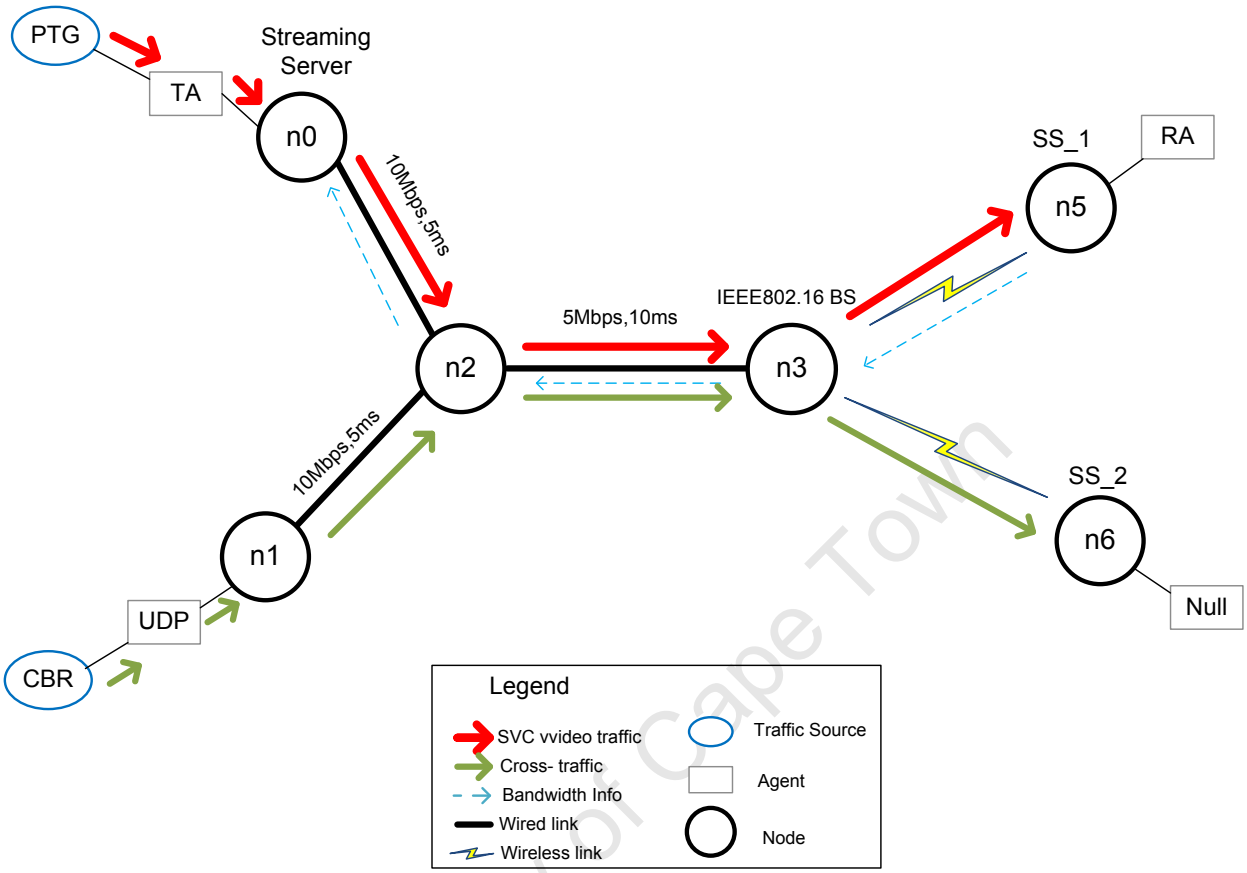

Figure 4.8 Simulated network model

In contrast to the network model discussed in Section 3.4, the internet cloud was eliminated during implementation. The assumption made was that the video caching server existed between the internet and the WiMAX network. Therefore, it took the role of the streaming server. Internet cloud was eliminated to avoid problems such as long delays introduced by the internet so that the focus is on the WiMAX network. The delay affects the performance of the streaming scheme. The delay should be shorter than the interval at which bandwidth signals are sent to the streaming server. Otherwise, the bandwidth signals would arrive late and become useless.

A Summary of the network model is presented in Table 4.5, which shows the network nodes and their corresponding network objects as defined in NS-2.

Table 4.5 Network model summary

Network Node	NS-2 Objects			
	Node no.	Traffic generator	Sending agent	Receiving agent
Streaming server	0	Pipelining_TrafGen	Transport_Agent	N/A
CBR_Source	1	CBR	UDP	N/A
Router	2	N/A	N/A	N/A
Base Station	3	N/A	N/A	N/A
SS_1	4	N/A	N/A	Receiving_Agent
SS_2	5	N/A	N/A	NULL

4.5.2 Traffic sources

4.5.2.1 SVC traffic

Three different video sequences are used to generate SVC traffic. The videos are selected based on their motion degree (which is due to camera motion and rate of change in scenes). The reason for using different videos is that achievable received quality depends on the properties of video that is transmitted. For instance, 2% packet loss does not equally degrade visual quality of low and high motion degree videos.

The following video sequences outlined below were chosen and downloaded from the trace library [40]. All videos are presented in high quality standard, CIF, and in YUV format.

i. News_cif.yuv

This is a relatively low motion video. It is captured during the news bulletin. The camera used to capture video is fixed and the change in pictures is relatively small. This is evident from the almost identical picture frames from time to time.

ii. Paris_cif.yuv

This is considered to be a moderate motion video. It was captured during a TV programme. The settings are the same as in News video described above, except that in this video there is slightly higher degree of motion.

iii. Foreman_cif.yuv

This is a relatively high motion video. This video shows a supervisor and his subordinates at a construction site. The movement in this video is significantly greater than that of the news video. The camera position capturing this scene is more erratic compared to the Highway video and there is a higher degree of change between video frames

4.5.2.2 Cross traffic

The cross traffic was sent from n1 to n5 as shown in Figure 4.8. This cross traffic was characterized by large amounts of data that was sent in short intervals (5ms), which alternated with no-traffic intervals (3ms). The application/traffic generator that was used to create bursts was CBR exponential traffic application. The same packet size (1000B) was used for all simulation runs and the CBR traffic was transmitted for 20s. The sending rate was altered once (from 1.6Mbps to 3.8Mbps in increments of 100kbps) per streaming session in order to create different congestion levels.

4.5.3 Mobile WiMAX network settings

Table 4.6 shows the default and non-default configuration parameters used in the WiMAX segment of the simulated network.

Table 4.6 WiMAX network configuration parameters [19]

Default settings		
Parameter	**Value**	
Duplexing type	TDD	
System Channel Bandwidth (MHz)	10	
FFT Size	1024	
Sampling frequency (MHz)	11.2	
Sub-carrier frequency spacing	10.94 kHz	
Useful symbol period (T_b=1/f)	91.4 microseconds	
Guard Time (T_g=T_b/8)	11.4 microseconds	
OFDMA Symbol Duration (T_s=T_b+T_g)	102.9 microseconds	
Frame duration	5 milliseconds	
Number of OFDMA Symbols	48	
	DL PUSC	**UL PUSC**
Null Sub-carriers	184	184
Pilot Sub-carriers	120	280
Data Sub-carriers	720	560
Sub-channels	30	35
Non – default settings		
Topography dimensions X(m): Y(m):	 2500 2500	
Modulation Scheme	QPSK	
Minimum Bandwidth(kbps)	Base layer bitrate (<200kbps)	

QPSK was chosen as the appropriate modulation scheme since it offered low bandwidth but enough capacity for high definition video streaming. As a result, it was easier to create a network bottleneck when using lower capacity modulation schemes than when using the higher capacity ones, such as 64QAM. The minimum bandwidth for each connection depended on the base layer bit-rate since, minimum bandwidth was the capacity required to transmit the base

layer.

The service classes are not configured since they are not yet implemented in the NIST mobility module.

4.5.4 Experiment procedure

All the tools discussed work together as shown in the following steps in order to carry-out the tests.

Step 1: Encoding

H264AVCEncoderLibTestStatic -pf snr_main.cfg -lqp 0 30 -rqp 0 32 -lqp 1 24 -rqp 1 26

Step 2: Determine Iy, Pe, and Be.

BitStreamExtractorStatic -pt trace snr_svc.264

Step 3: Hinting

mp4box -hint -mtu 250 -add snr_svc.264 snr_svc.mp4

Step 4: Recording reference PSNR

psnr 352 288 420 news_cif.yuv snews_cif.yuv > ref_psnr.txt

Step 5: Creating video frames trace file

mp4trace -t UDP -f -m cam -s 157.159.16.152 12346 snr_svc.mp4 > st_snr

Step 6: Running network simulation

ns network.tcl

Step 7: Re-order packets

A separate script was written to perform this task. This step is required because during video frames scheduling, pipelining interleaves packets from various frames.

Step 8: Reconstructing video

etmp4 -F -0 sd_snr rd_snr st_snr snr_svc rsnr_svc

Step 9: Produce perceptual video evaluation results

psnr 352 288 420 news_cif.yuv rnews_cif.yuv > ref_psnr_news_cif.txt

4.6 Summary

This chapter has discussed the implementation of the proposed video streaming scheme. The Joint Scalable Video Model software has been used for the video coding part of the evaluation framework, while NS-2 has been used for the network simulation part. To gather results for video evaluation, EvalSVC was used. The implementation of the proposed pipelining-based video streaming scheme has been done on the back end of the NS-2 simulator, using C++.

All software tools were integrated and the framework works as follows. The raw video (.yuv format) from a file is encoded using the SVC encoder to produce an SVC bit stream (H.264 format). Medium granular scalability mode of the SNR encoder is selected when encoding the bit stream due to its scalability. The Hinter is then used to packetise SNR encoded bit stream in RTP packets, as well as adding the hint track. The resulting file has mp4 type. The mp4trace tool then generates the video trace file of the bit stream consisting of the frame index, type, size, etc. The trace file is used to generate video traffic which is delivered over the simulated network. The received packets are logged into a trace file, which together with other trace files and the hinted track are used by the SVC Re-builder to construct the received bit stream, as well as video transmission results. An SVC decoder is then used to decode the received video bit stream.

Chapter 5 Performance Evaluation

5.1 Introduction

The previous chapters have covered the design and implementation of the proposed pipelining-based video streaming scheme for the Mobile WiMAX networks. This chapter presents the performance evaluation of the proposed scheme. The performance of the pipelining-based scheme is compared to that of the most commonly used video scheduling algorithm: Earliest Deadline First (EDF).

The chapter begins by describing the performance metrics used to evaluate the scheme as well as their relevance in this study. Then, the simulation results and their discussions with respect to each of the performance metrics are presented.

5.2 Performance Metrics

To evaluate the proposed pipelining-based video streaming scheme, the following metrics are used:

- Packet loss ratio
- PSNR
- Frame loss
- Traffic burstiness
- Rate adaptation
- Frame latency
- Frame jitter
- Available bandwidth

The remainder of this section is aimed at describing the above listed performance metrics.

5.2.1 Packet loss

The Packet Loss Metric consists of dropped and/or lost packets. An example of packets that are dropped is in a congested network where there is insufficient bandwidth for transmission of all scheduled packets. The major challenge of packet loss in video streaming is that it results in visual quality degradation as well as bandwidth inefficiency, depending on the node in the network at which packet loss occurs. If the bottleneck is adjacent to the destination, as it is the case in this study, the channel is occupied from the source to the bottleneck node by the packets that eventually get lost, hence bandwidth inefficiency.

In this study, packet loss is mainly attributed to network congestion. The following expression is used to calculate the packet loss ratio:

$$PL = \frac{\mathrm{P_{sent}} - \mathrm{P_{rec}}}{\mathrm{P_{sent}}}$$

Where P_{rec}: number of packets received

P_{sent}: number of packets sent by the streaming server

It is important to note that in the context of video streaming, it is not only the number of packets lost that matters, but also the kind of data that is contained in the lost packets. For example, MPEG-4 defines different types of frames (I, P, B) and some generic headers. Therefore, packets belonging to different frame types impact video quality differently.

5.2.2 PSNR

The Peak Signal Noise Ratio (PSNR) is a scientific/mathematical formula used to measure video quality. This metric is a derivative of signal to noise ratio, which compares the signal energy to noise energy. The equation below defines the PSNR between the luminance component of the image/frame (Y), the source video frame (S) and destination video frame (D) [12, 24, 38]:

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

Where:

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

n= video frame

N_c= number of column pixels within the frame

N_r =number of row pixels within the frame

k = number of bits per pixel

i,j = pixel combination for frame

The PSNR is calculated frame by frame, which can be very tedious if the bit stream consists of hundreds or thousands of frames. In this study, the average PSNR and its standard deviation are used to compare the video test sequences transmitted using both pipelining-based and Earliest Deadline First streaming algorithms.

When computing PSNR, if a given frame is lost, the previous frame is copied and this is used as a method for error concealment. However, this may result in inaccurate video visual quality assessment. To complement this, frame loss is coupled with PSNR to assess the visual quality of the transmitted video streams.

5.2.3 Frame loss

Frame loss is defined as the total number of frames lost during the streaming session. In this study, the assumption made is that the key frames are protected against network losses. Hence, the frame loss metric consists of only P and B frames in the case of video scheduling using the pipelining-based streaming scheme. The following expression is used to calculate frame loss:

$$FL = \frac{F_{sent} - F_{rec}}{F_{sent}}$$

Where: F_{rec}: number of packets received

F_{sent}: *number of packets sent by the streaming server*

As mentioned, this metric is used together with the PSNR metric to assess quality of user experience of the transmitted video sequences.

5.2.4 End-to-end frame latency and frame jitter

Video streaming applications are sensitive to delays. In this study, the focus is on frame delay, because the proposed pipelining-based video streaming scheme does not affect end-to-end packet delay. Relatively small frame delays and packet delays, that is, delays within the range recommended by the ITU-T [41] are desirable.

End-to-end delay of frame X is calculated by sending the packets of size Y belonging to the frame from the streaming server to the subscriber station (video receiver). The timestamps of packets as they leave the server as well as when they get received are recorded. The frame delay is the difference between the timestamp of the last packet of the frame at the receiving side and the timestamp of the first packet of the frame when it left the server. NS-2 uses a universal clock; therefore, it is correct to subtract timestamps of a packet.

5.2.5 Frame jitter

Video streaming applications are also sensitive to jitter. Jitter, like other QoS metrics, such as end-to-end delay and packet loss, arises due to bandwidth dynamics. An ideal video streaming system should have the means of reducing sending rate in order to control for jitter when transmission bandwidth decreases so as to keep jitter within the ranges recommended by the ITU-T [41].

In this study, frame jitter is defined as the variance of the inter-frame times. Frame time is determined by the time at which the last packet of the frame is received. The definition of frame jitter as used in this work is given by following equations [38]:

Inter-frame time, $itF_0 = 0$

$$itF_m = tF_m - tF_{m-1}$$

itF_m: $Timestamp\ of\ the\ last\ segment\ of\ the$

$$\text{frame jitter } jF = \frac{1}{M}\sum_{i=1}^{M}(it_i - i\bar{t}_M)^2$$

Where M: number of frames

$i\bar{t}_M$ *: average of inter-frame times*

5.2.6 Traffic burstiness

Traffic burstiness or variability of traffic volume is an indication of how traffic fluctuates over time. In this study, the traffic burstiness is used to assess how the pipelining-based streaming scheme affects the SVC traffic variability. The hypothesis made is that the pipelining-based video streaming scheme exhibits smoother traffic when compared to the Earliest Deadline First video streaming scheme. This results in improved bandwidth efficiency.

5.2.7 Rate adaptation

The Rate Adaptation Metric is used to determine the adaptability of the video streaming server to the time varying transmission bandwidth during a streaming session. In this study, rate adaptation is defined as the variance between the sending and receiving rate. In an ideal situation, the variance between the sending and receiving rates is zero. Therefore, the smaller variance indicates the adaptability of the video streaming server.

5.2.8 Available bandwidth

The performance of the pipelining-based video streaming scheme depends on the accuracy of the bandwidth estimation mechanism. The available bandwidth is measured based on the packet loss. Available bandwidth is estimated using the following expression and it is measured in Mbps:

$$Available\ bandwidth = \frac{packet\_size*(count+1)}{(t_n - t_1)} \quad (5)$$

Where:

t_n *arrival time of the last received packet*

t_1 arrival time of the first packet

count is the received packet counter

packet_size is the maximum packet size in bytes

To determine accurary of the bandwidth estimation mechanism, to methoths are use to estimated bandwidth. Firstly, available bandwidth is calculated by subtracting the cross-traffic throughput from the known channel capacity. The second value of available bandwidth is obtained by using Equation 5 above.

5.3 Properties of the video sequences

Table 5.1 shown below summarises the properties of the test video sequences used in this study. The video sequences have different bit rates, frames sizes, etc.: as a result, they pose different challenges when streamed over the Mobile WiMAX network.

Table 5.1 Settings and properties of the video test sequences

Test Sequence	Packet size	No. of I-frame layer0 packets	No. of P-frame layer0 packets	No. of B-frame layer0 packets	Minimum bit rate (Mbps)	Maximum bit rate (Mbps)
Paris	750	12	2	1	214.19	1149.582
Foreman	500	13	1	1	201.10	924.80
News	250	16	1	1	107.70	678.01

Packet sizes were determined based on the bit rate of the test sequences and the average size of the B frames. The aim was to have at least two packets for each frame to allow dropping of the enhancement layer packet when bandwidth is insufficient for both packets of the frame. The number of the packets for each frame was selected to allow scheduling of all the packets of layer 0 NALU.

5.4 Performance results

5.4.1 Packet loss

The following three figures show the comparison between the proposed pipelining-based streaming and the Earliest Deadline First streaming schemes in terms of packets lost during each streaming session. The y-axis represents the packet loss ratio while the x-axis represents the bit-rate of the cross-traffic at different streaming sessions. The cross-traffic bit-rate was incremented by 100kbps per streaming session to assess the performance of the two scheduling algorithms as the cross-traffic bit rate increases. Increase in cross-traffic bit rate results in increased congestion level, which means that the congestion level is controlled by the cross-traffic bit rate.

Figure 5.1 shows the packet loss ratio comparison of the two scheduling algorithms when streaming the Paris video sequence at various bit rates.

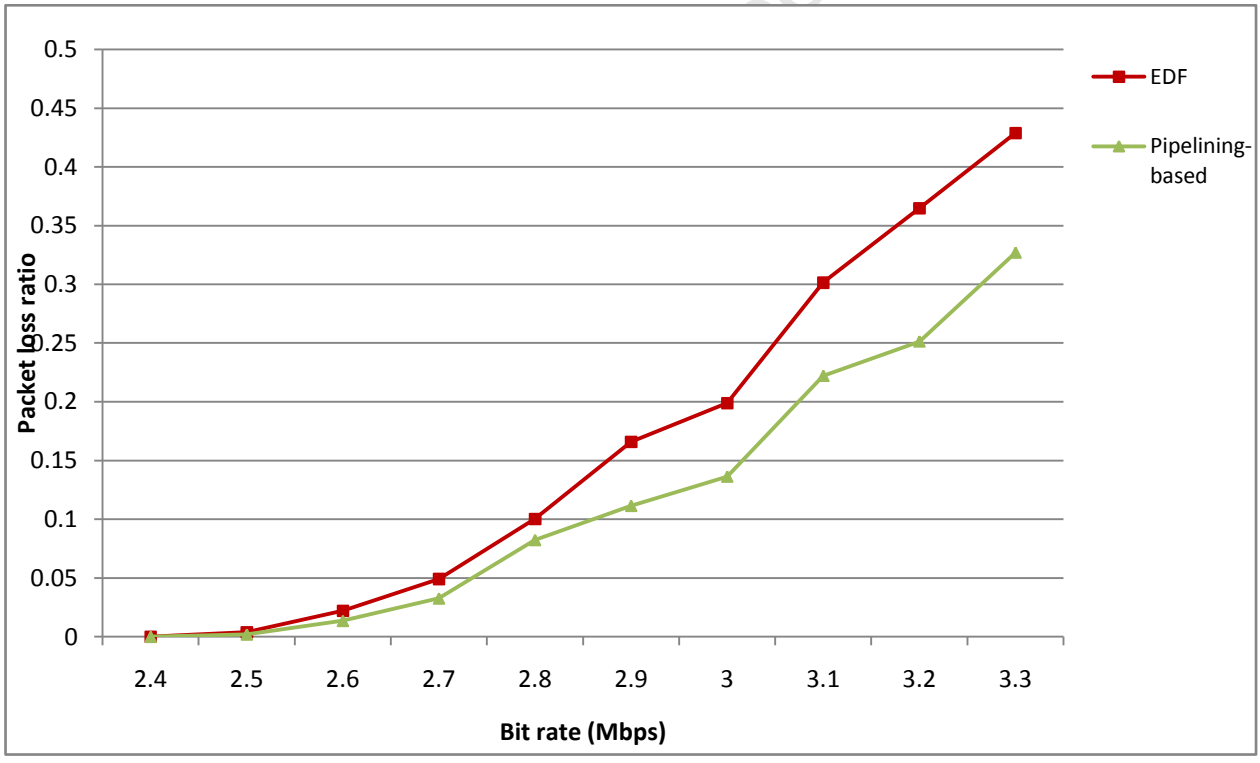

Figure 5.1 Packet loss ratio comparison between pipelining-based scheduling and EDF for the Paris video sequence

Paris has higher bit rate compared to other test sequences. In the first three streaming sessions, when cross-traffic bit rate is below 2.5Mbps, packet loss ratio is below 0.006, with the

pipelining-based scheduler having a slightly smaller ratio. As the cross-traffic bit-rate increases, the packet loss ratio increases as well, with pipelining-based video streaming scheme still exhibiting the lower packet loss. From Figure 5.1, it is evident that the pipelining-based streaming scheme packet loss remains below the Earliest Deadline First streaming scheme in all streaming sessions. The average improvement in packet loss when streaming Paris video was 5.07%.The minimum improvement is 0.2%, obtained when cross-traffic bit-rate was 2.5Mbps, while the maximum improvement is 11.34%, and was achieved when cross-traffic bit rate was 3.2Mbps.

Figure 5.2 depicts the packet loss ratio comparison for the two algorithms when streaming the Foreman video sequence.

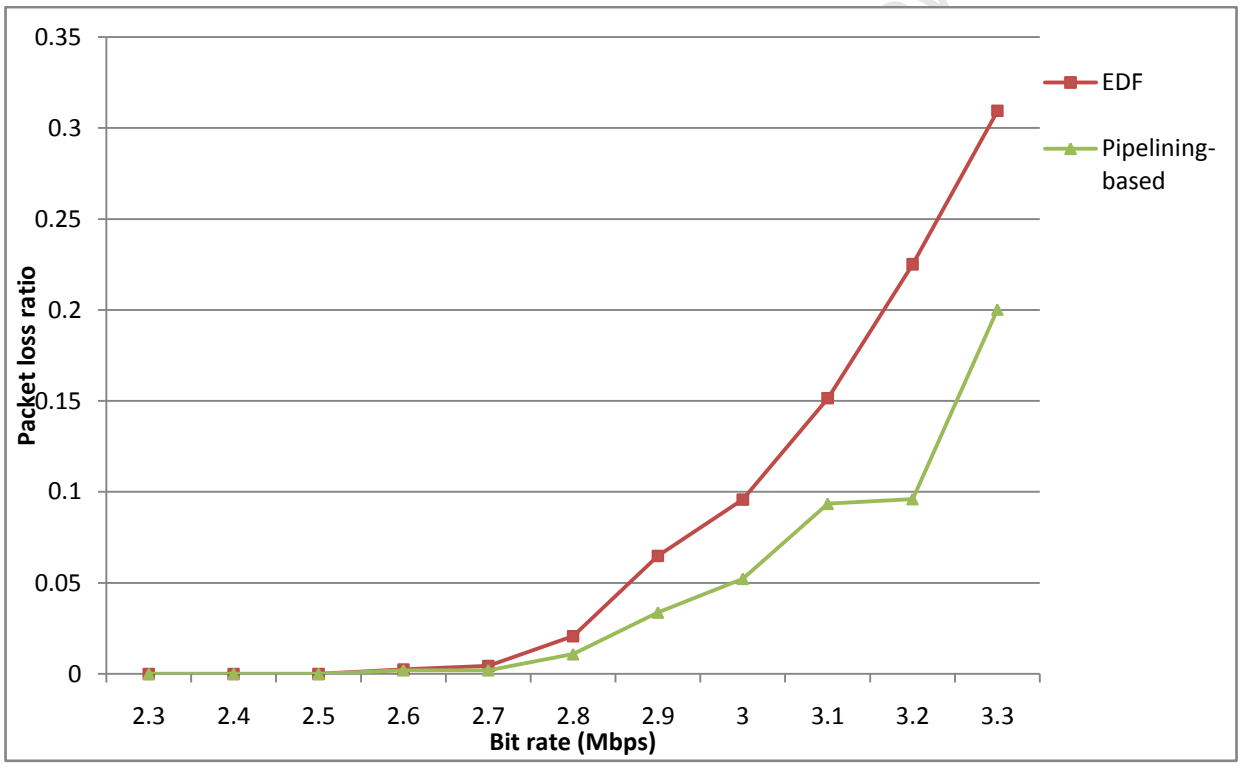

Figure 5.2 Packet loss ratio comparison between pipelining-based scheduling and EDF for the Foreman video sequence

Compared to the Paris video sequence, Foreman has a lower bit rate. The effect of lower bit rate is evident from the lower packet loss at similar cross-traffic bit rate values as the two video sequences are transmitted over similar network conditions. For instance, Foreman started to experience packet losses over 0.006 when the cross-traffic bit rate exceeded 2.7Mbps as

opposed Paris, which started at 2.5Mbps. Pipelining-based scheduling again out-performed the default scheduling by experiencing lower packet loss at all streaming sessions. The average improvement offered by pipelining-based scheduling was 4.79%. The maximum improvement was 12.90% obtained when the cross-bit rate was 3.2Mbps, while the minimum improvement was 0.02%, which was obtained when the cross-traffic bit was 2.7Mbps.

Figure 5.3 shows packet loss ratio comparison for the News Video Sequence.

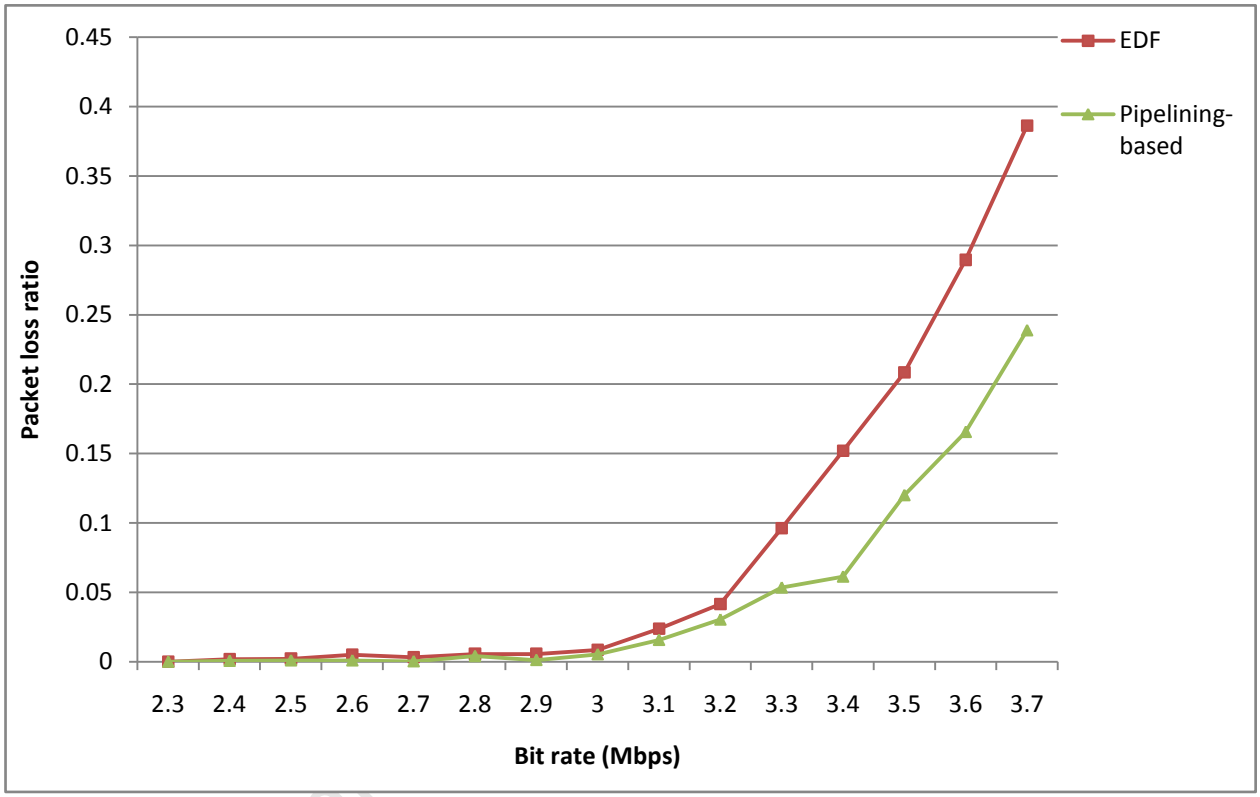

Figure 5.3 Packet loss ratio comparison between pipelining-based scheduling and EDF for the News video sequence

Compared to the other video sequences, this sequence has the lowest bit-rate. Therefore, it experienced the lowest packet loss at corresponding cross-traffic bit rates compared to both Paris and Foreman. For News, both streaming systems experienced packet loss ratio over 0.006 when the cross-traffic bit rate has exceeded 3Mbps. Similar to the previous cases, the pipelining-based scheme experienced lower packet loss. The average improvement in packet loss when streaming News video was 3.79%. The minimum improvement was 0.12%, which was obtained when cross-traffic bit-rate was 2.4Mbps, while the maximum improvement is 14.74%, which

was achieved when cross-traffic bit rate was 3.7Mbps.

Evident from the above three figures, compared to the Earliest Deadline First, the pipelining-based video streaming scheme reduced packet loss when available bandwidth decreased during a video streaming session. Another property of the SVC video utilised by the pipelining-video streaming scheme to minimise packet loss is that incompletely received NALUs are not decodable, which makes transmission of NALUs that will not be fully received a waste on network resources such as bandwidth. However, packet loss could not be eliminated due to factors such as dependence of the pipelining-based video streaming system on the bandwidth monitoring mechanism.

The bandwidth-monitoring algorithm used in this study, uses the video packets to estimate available bandwidth and reports the losses to the streaming server so that the server can adjust the sending rate. The problem arising due to the usage of the video packets is that in some periods, a large number of packets may be lost, hence making the proposed video streaming scheme, which is reactive to packet loss, to still lose some packets. Additionally, there is a delay of at least 80 milliseconds in the feedback path. Therefore, the bandwidth information that the streaming server uses to scale the bit stream is not up-to-date and accurate, also contributing to packet loss.

According to Vidyo[43], packet loss of about 20% is acceptable in SVC video streaming. In fact, all packets of enhancement layers are discardable and discarding them may result in packet loss of over 50%, which is acceptable depending on the encoding configurations. Therefore, in the above figures, packet losses may exceed 20% but still result in usable bit streams.

5.4.2 Peak-signal-to-noise ratio (PSNR) and Frame Loss

Figure 5.4 presents the peak signal to noise ratio for Paris, Foreman and News test video sequences when the two scheduling schemes are used. The streaming networks are set to almost the same conditions in which the packet loss experienced by the three sequences when Earliest Deadline First was around 10%. The y-axis represents the PSNR values obtained for the test sequences with a range of 0 to 45 dB together with their standard deviations. The x-axis represents the reference video, received video when scheduling was carried out using both

pipelining-based and Earliest Deadline First video streaming schemes.

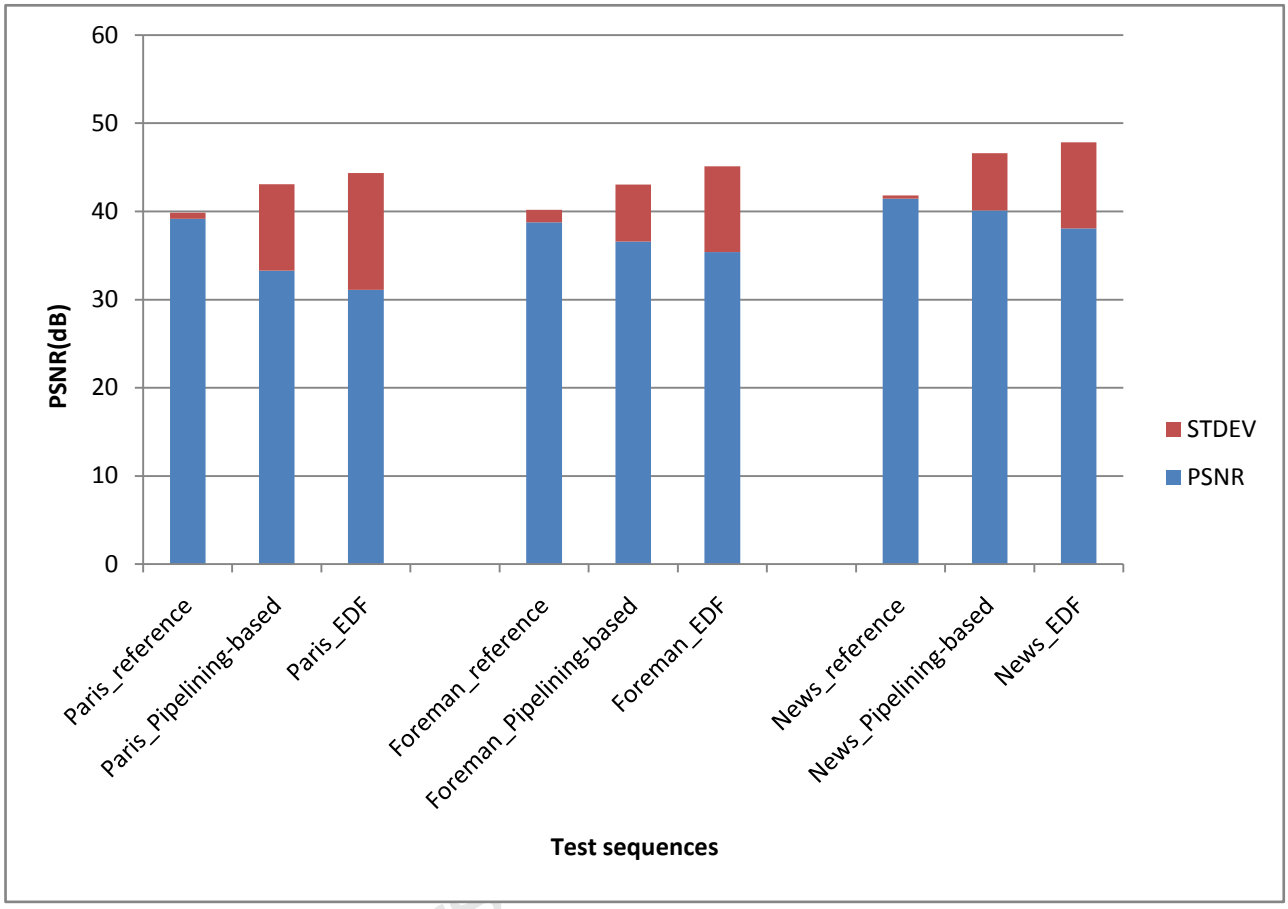

Figure 5.4 PSNR values and STD deviations for Paris, Foreman and News videos.

An interesting observation from the figure is that the PSNR values of the three video sequences when Pipelining based scheduling scheme is used are greater than those obtained when Earliest Deadline First is used. This is attributed to the following criteria :

- Pipelining-based video streaming scheme experiences lower packet losses under the similar network conditions, consequently improved video quality.
- Pipelining-based video streaming scheme protects key frames in the base layer against transmission losses, hence preserves video quality.
- Pipelining-based video streaming scheme prioritises NALUs and drops less priority NALUs when bandwidth becomes insufficient for transmission of all NALUs. This also helps in minimising visual quality degradation when

bandwidth decreases during a streaming session.

When comparing the PSNRs of the video sequences, Paris experienced slightly higher quality degradation, of about 5.88dB in the case of pipelining-based streaming and 8.06 dB with Earliest Deadline First. For the Foreman video, there was 2.21 dB loss when pipelining-based scheme was used and 3.38 dB loss when the Earliest Deadline First was used. For News video, 1.36 dB quality loss was obtained when pipelining-based scheme was used and was 3.37 dB when Earliest Deadline First was used. In all the three video sequences, pipelining-based video scheduling outperformed Earliest Deadline First in terms of preserving video quality.

Another important observation from Figure 5.4 is that the standard deviations obtained when using pipelining-based streaming are smaller than those obtained when using Earliest Deadline First. Low standard deviation values indicate smooth video in terms of variation in quality levels during the video play-out process.

The PSNR metric has indicated visual quality degradation when streaming video using both Earliest Deadline First and pipelining-based scheduling. However, the major challenge of using only PSNR to evaluate video quality is that when used alone, PSNR may be inaccurate in terms of visual quality assessment, as some of the frames might have been lost during the transmission process. For this reason, PSNR is coupled with frame loss to assess the two streaming schemes with respect to visual video quality. Increase in frame loss indicates higher visual quality degradation.

Table 5.2 summaries the frame loss ratio for the video sequences under the same network conditions that produced the PSNR values shown in Figure 5.4. Compared to other video sequences, Paris experienced higher frame losses when using both streaming schemes. The pipelining-based streaming scheme has decreased frame loss by about 7% compared to Earliest Deadline First. For the Foreman and News videos, pipelining-based has decreased frame loss by 4%.

Table 5.1 Frame loss for Paris, Foreman and News video sequences

Vide sequence	Pipelining-based	EDF
Paris	0.076667	0.15
Foreman	0.026667	0.066667
News	0.023333	0.06

Considering both PSNR and Frame loss metrics, it can be seen that the pipelining-based streaming scheme outperforms the Earliest Deadline First. The results show that pipelining-based streaming scheme optimises quality of user experience when streaming SVC video over the Mobile WiMAX network.

5.4.3 Traffic burstiness

The subsection presents a comparison between the traffic profiles of pipelining-based streaming scheme and that of the Earliest Deadline First when traffic leaves the streaming server. The y-axis represents the burst size of traffic scheduling in a period, while the x-axis represents the scheduling period frame index. Since in H.264 video streaming there are as many frames as periods, the period index is used to avoid confusion. Pipelining-based streaming video frames thus schedule in more than one period, which was not the case with the Earliest Deadline First.

Figure 5.5 depicts the traffic profiles of Paris video sequence. From the figure, it can be seen that the peak values of the key frames have been reduced when scheduling was carried out using the pipelining-based scheme. Pipelining-based streaming has reduced the average burst size of the key frames from 15000B to 9000B. This decrease comes at an expense of the non-key frames, that is, the non-key frames of the pipelining-based streaming scheme are much greater in size compared to those of the Earliest Deadline First. The increased size of the non-key frames is addressed through bit stream scaling, by dropping packets based on the priorities when bandwidth is insufficient, thus decreasing burst size of the non-key frames.

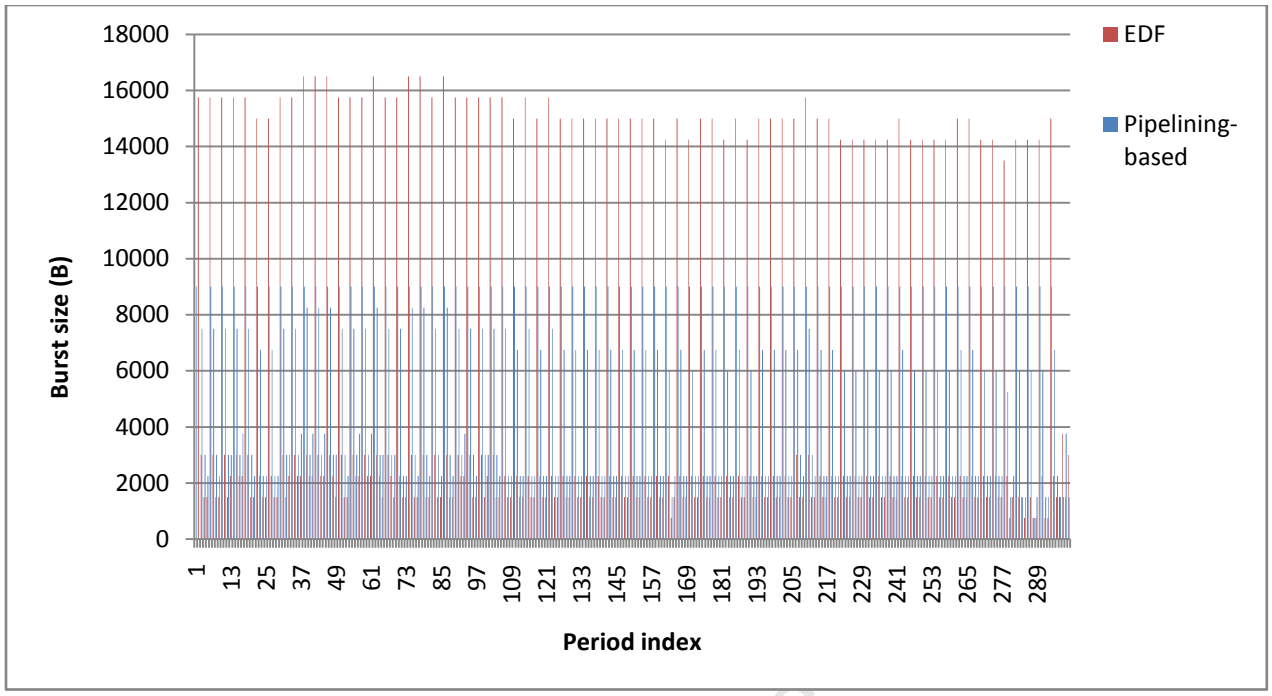

Figure 5.5 Traffic profiles for Paris when scheduling video packets with pipelining-based and EDF algorithms

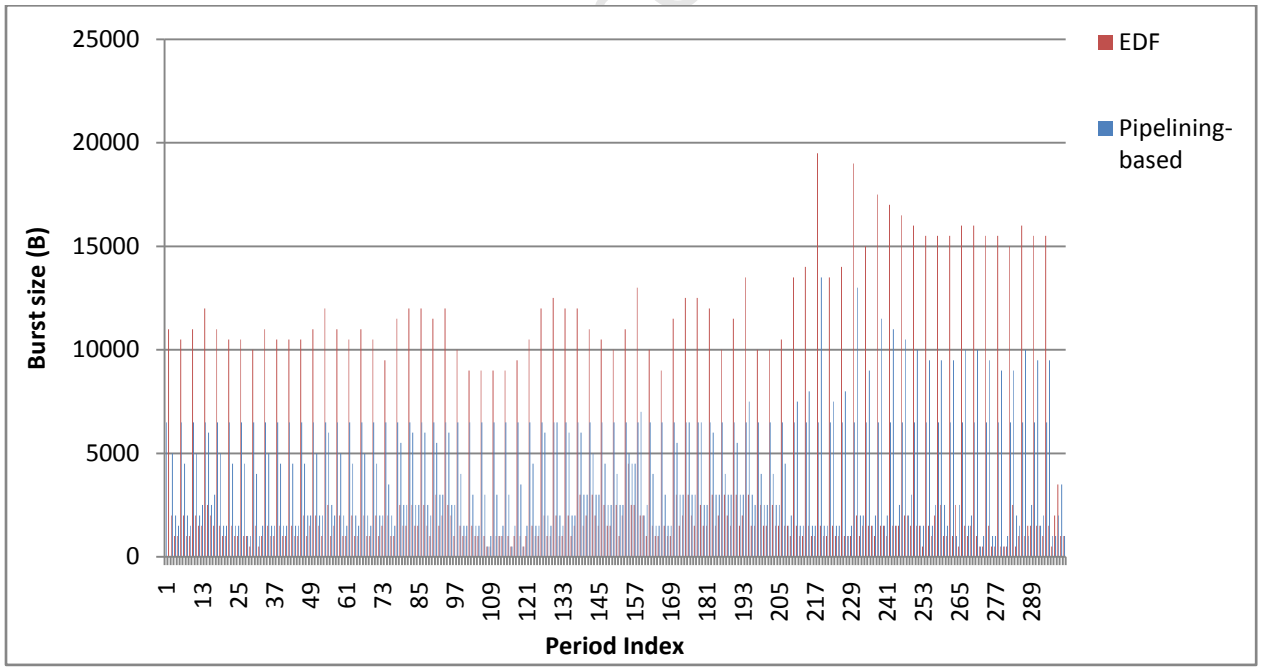

Figure 5.6 Traffic profiles for Foreman when scheduling video packets with both pipelining-based and EDF scheduling algorithms

Figure 5.6 shows the traffic profiles for Foreman video sequence. When packet

scheduling was carried out by the Earliest Deadline First, the maximum burst size was about 10000B for the first 200 periods, and it gets over 15000B for the last hundred periods. The burst sizes for the P frames were below 5000B and most B frames below 2500B. However, when using a pipelining-based scheduling algorithm, the burst sizes of the key frames was reduced in order to decrease the peaks to about 6500B for the first 2 hundred periods and to about 1500B for the remaining hundred periods. The frame size variability among frames of the same type in this sequence is due to its higher motion degree compared to the Paris sequence. Like in the previous case, the peaks are reduced at an expense of P and B frames, but they also do not go beyond 10000B which still show a significant improvement in reducing traffic variability.

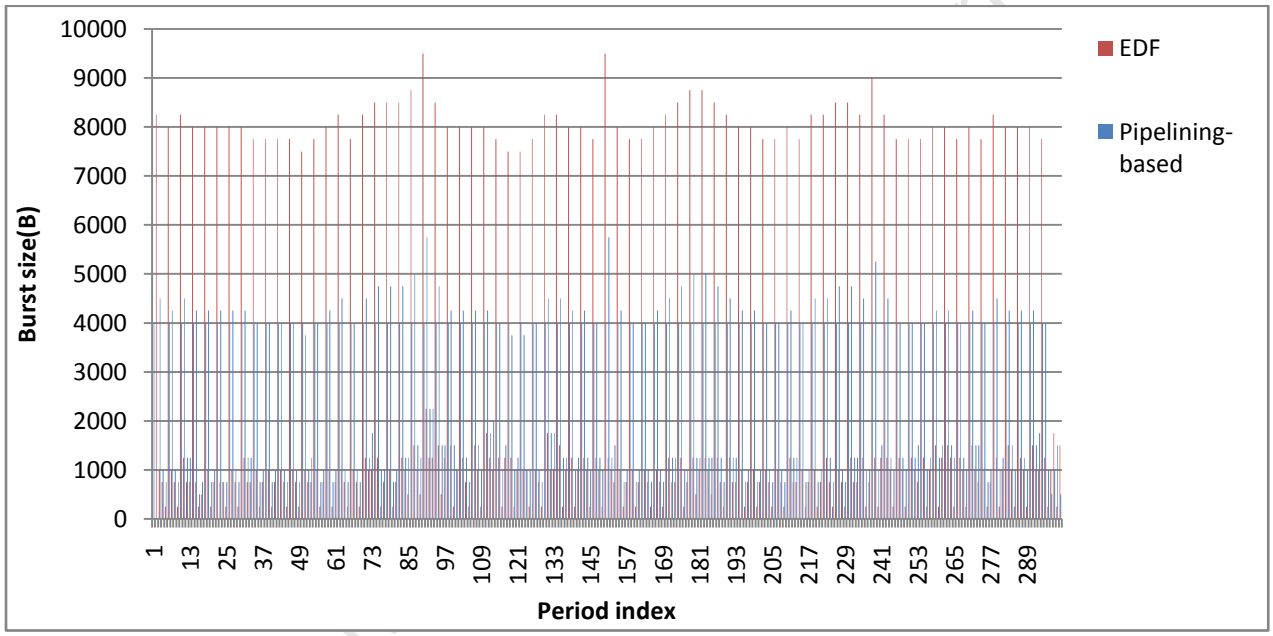

Figure 5.7 Traffic profiles for News when scheduling video packets with pipelining-based and EDF algorithms

Figure 5.7 depicts the traffic profiles for News video sequence. When using Earliest Deadline First scheduling algorithm, the burst size of the key frames is about 8000B, and it was below 2000B for the non-key frames. Again, using pipelining-based scheduling algorithm decreased the burst size of the key frames to about 4000B, but also increased burst sizes of the non-key frames.

Table 5.2 presents the statistics of the SVC-encoded video sequences. From the table, it is evident that the pipelining-based scheduling algorithm reduced maximum frames size as well as

reduced traffic variability, hence offered SVC traffic smoothing. When computing the maximum burst size, the first two periods were excluded as they do not convey the true picture of a GOP, since the key frames follow each other which occurs once in the whole bit stream. Additionally, bandwidth adaptation starts in the third period because of feedback information delay, so analysis of the proposed scheme is done from the third period onwards.

Table 5.2 Statistics of the SVC-encoded video sequences. (MBS: the maximum burst size, STD: standard deviation of the frame size).

Video Sequence	**EDF Scheduling**		**Pipelining-based Scheduling**	
	MBS	STD	MBS	STD
Paris	16500	5904.067	9000	3184.309
Foreman	19500	4930.7	13500	2688.804
News	9500	3251.026	5750	1740.278

The following improvements are achieved through pipelining-based scheduling:

- For the Paris bit stream, the maximum burst size was reduced from 16500B to 9000B. Traffic variability was also reduced. When using the EDF scheduling algorithm, the frame size variability from the mean size was 5905B and using the pipelining-based algorithm reduced the variability from the mean frame size to 3185B. Thus, pipelining-based scheduling reduced SVC traffic variability when streaming Paris video sequence over the Mobile WiMAX network.
- For the Foreman video sequence, the maximum burst size was reduced by about 30%, from 19500B to 13500B. When using the EDF scheduling algorithm, the frame variability from the mean frame size was reduced from 4931B to 2689B, that is, through pipelining-based scheduling.
- For the News bit stream, the maximum burst size was reduced by about 40%, from 9500B to 5750B. When using the EDF scheduling algorithm, the frame size variability from the mean size was reduced from 3252B to 1741B, that is, through pipelining-based scheduling.

The major advantage of traffic smoothing is bandwidth efficiency. The smoothed traffic has lower bandwidth requirements and bit rate variability. Apart from traffic smoothing, bandwidth efficiency in this study is achieved through scaling the video bit stream to the available bandwidth.

5.4.4 Rate adaptation

This subsection presents rate adaptation results of the pipelining-based scheduling algorithm for the three video streams. These results are compared to those of the Earliest Deadline First scheduling algorithm. The x-axis represents a video streaming session in seconds, in which the first period started at 100s and was incremented by 40ms to 111.92. The y-axis represents either the sending rate or the receiving rate in Mbps. While the two graphs are correlated on the same scale for comparison purposes, in reality, the receiving side starts receiving packets a few milliseconds later, due to network delays.

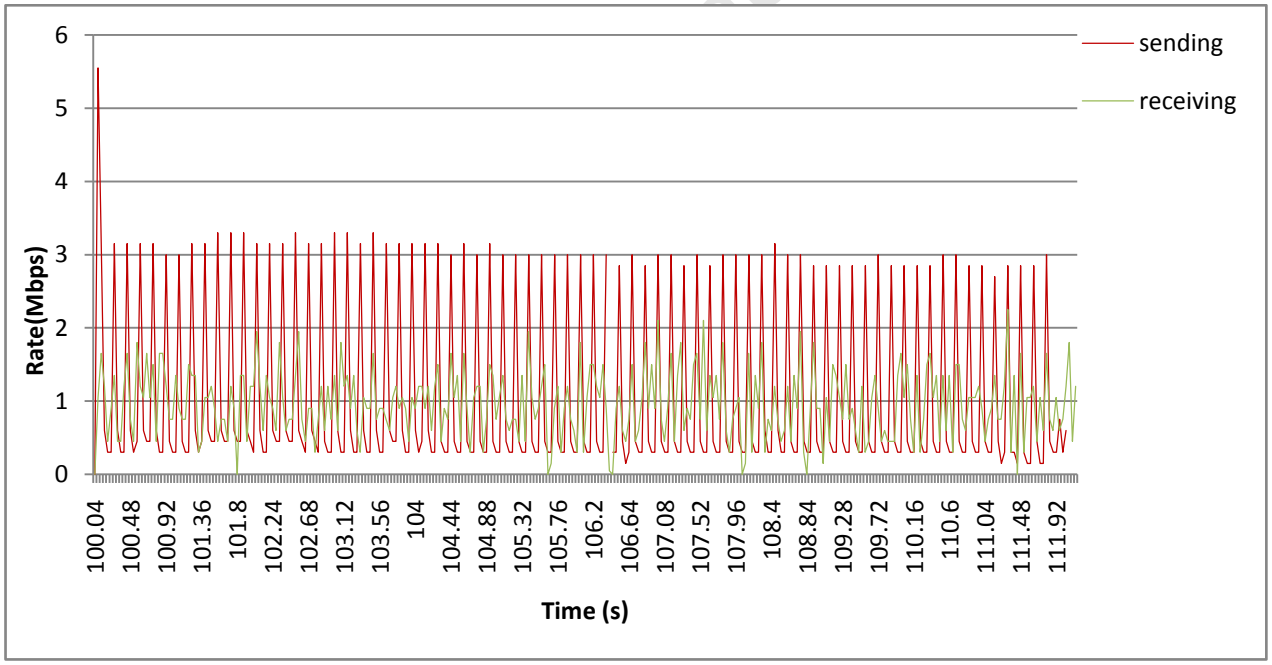

Figure 5.8 Sending rate versus receiving rate of Paris video when using EDF

Figure 5.8 shows the sending rate and receiving rate graphs of the Paris video bit stream when using Earliest Deadline First. Peak rate at the server went as high as 3Mbps, but due to limited network capacity, the high data rates were not sustained. As a result, the receiving peak rate was just above 2Mbps. Since the Earliest Deadline First streaming scheme was not

bandwidth adaptive, there was a difference of about 1Mbps between the sending and receiving peak rates. However, variance between the two graphs is 0.487Mbps.

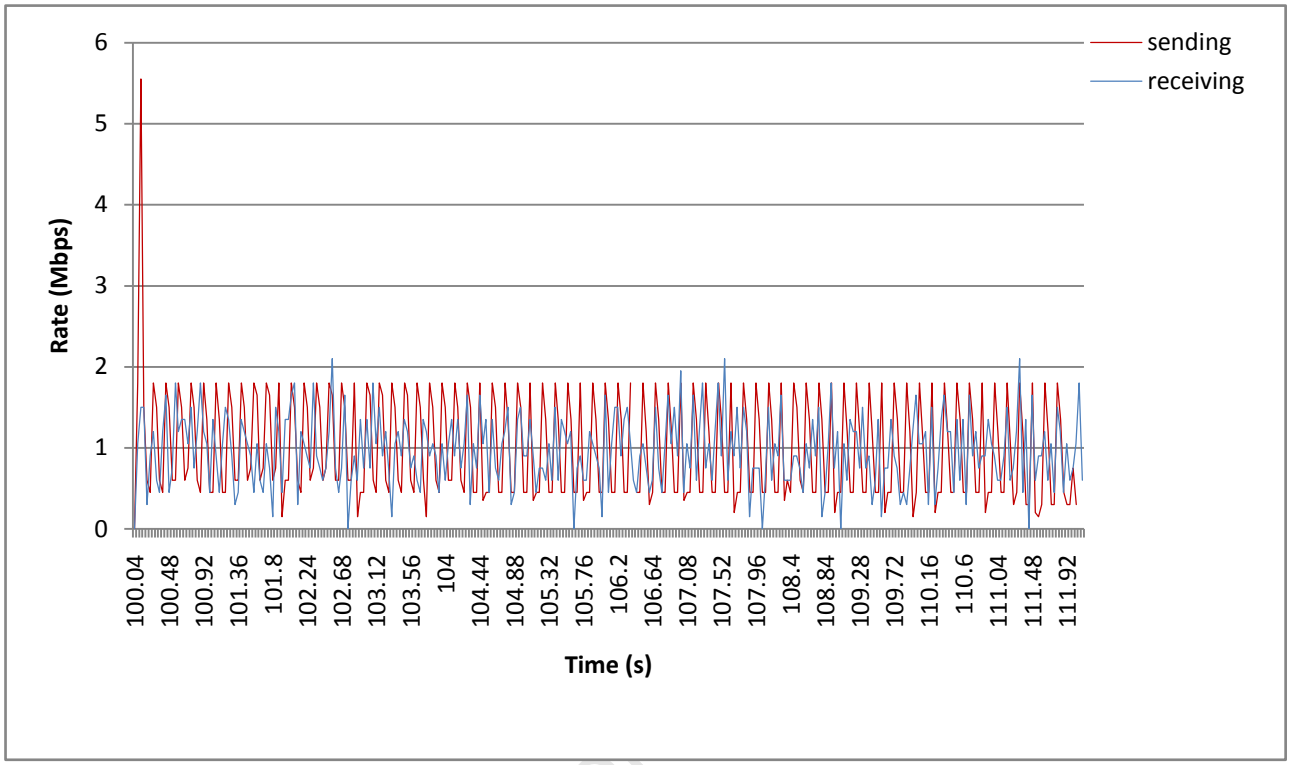

Figure 5.9 Sending rate versus receiving rates of Paris video when using pipelining-based scheduling algorithm

Figure 5.9 depicts the sending and the receiving rate graphs of Paris video bit stream when the pipelining-based scheduling algorithm was used. The peak rate at the server side was 1.9Mbps and this is attributed to the traffic smoothing property of the pipelining-based scheduling algorithm discussed in subsection 5.4.2 above. The network conditions are similar to those of the Earliest Deadline First case above. Thus, the receiving peak rate is the same. The variation between the two graphs is 0.167Mbps, which is smaller than that which is obtained when using Earliest Deadline First.

Figure 5.10 illustrates the sending rate and the receiving rate when scheduling video frames of the Foreman bit stream using Earliest Deadline First. The peak rate at the server side went as high as 3.9 Mbps. In this video the peak rate variation was greater than in Paris. This was due to the higher degree of motion of the Foreman video. Similar to the previous case, the network bottleneck still exists; as a result, the receiving peak rate was decreased to the maximum

of 2.1Mbps. Since, the Earliest Deadline First streaming scheme was not adaptive; there was packet loss experienced, which was due to the network bottleneck. The variation between the sending and receiving rate graphs is 0.357Mbps.

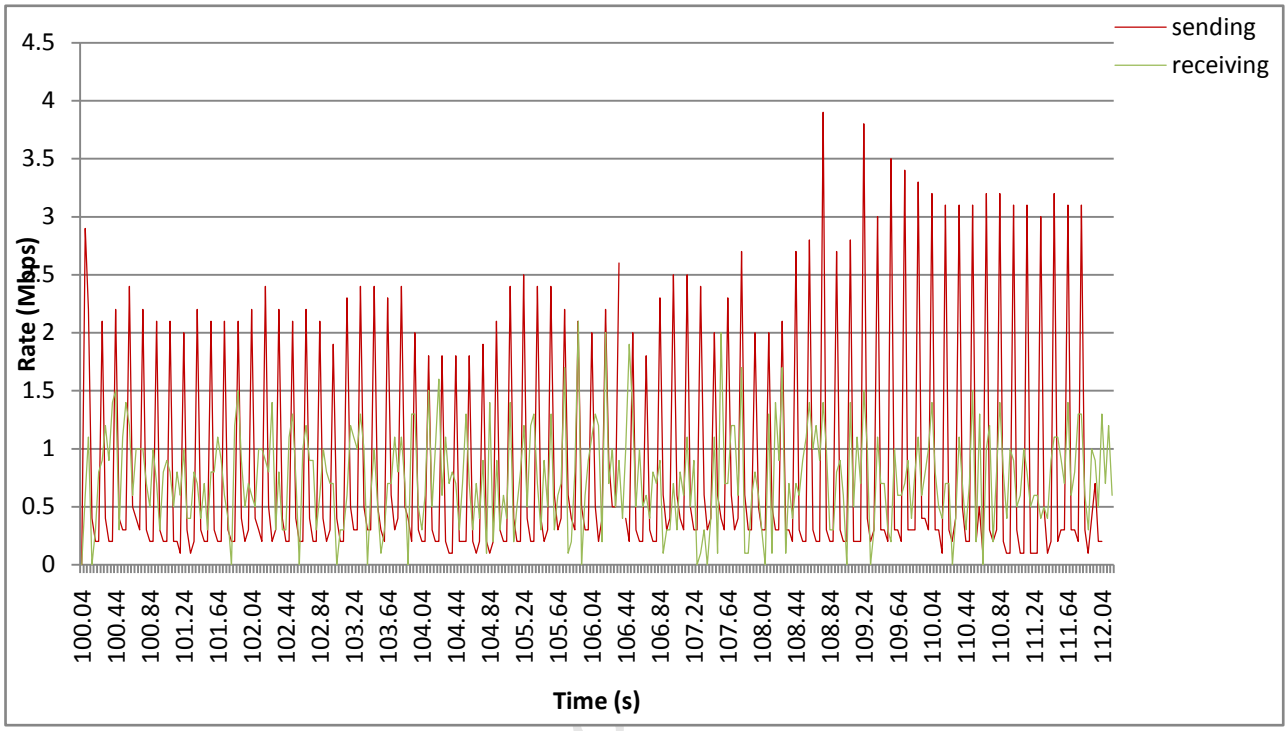

Figure 5.10 Sending rate versus receiving rates of Foreman video when using EDF

Figure 5.11 shows the corresponding rate adaptation graphs for the Foreman video when scheduling is carried out using pipelining-based algorithm. The maximum sending peak rate was 1.3Mbps from 100.00s to 107.56s. It then increased to 2.75Mbps after 107.56s. When comparing graphs in Figure 5.10 and Figure 5.11, from 108.44s onward, it can be seen that the number of peaks with rate over 1.5Mbps at the server has decreased in the Figure 5.10. This was because in Figure 5.11 the server reduced the sending rate using the feedback information from the receiving client, hence adapting to the available bandwidth. The variation between the sending and receiving rates is 0.119Mbps.

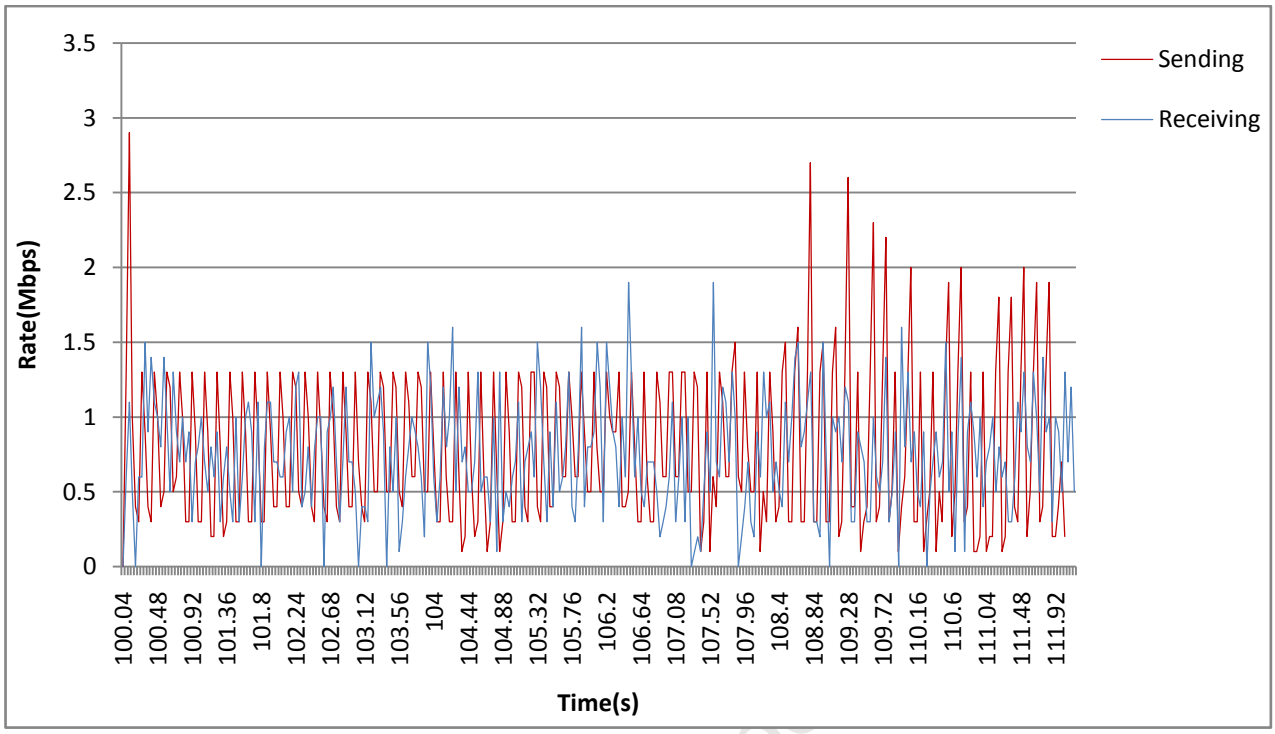

Figure 5.11 Sending rate versus receiving rates of Foreman video when using pipelining-based scheduling algorithm

Figure 5.12 illustrates the sending and the receiving rate for the News video bit stream when scheduling video packets with Earliest Deadline First. Like in the previous cases, the Earliest Deadline First is not adaptive; as a result, packet loss arises due to the network bottleneck. The peak rate at the server is above 1.5Mbps, but this rate is not maintained at the receiving side due to the network bottleneck, as a result the receiving peak rate is below the sending rate at all times. The variation between the two graphs is 0.177Mbps.

Figure 5.13 depicts the sending and receiving rate graphs of the News bit stream while the pipelining-based video scheduling scheme is used. When comparing the sending rate of the two scheduling algorithms, it can be seen that the pipelining-based algorithm reduced the sending rate. For instance, between period 103.20s and 104.00s, the number of peaks is fewer compared to those of Earliest Deadline First. This confirms that the pipelining-based scheduling algorithm adjusts the scheduling rate when bandwidth decreases. The variation between the two graphs is 0.074Mbps.

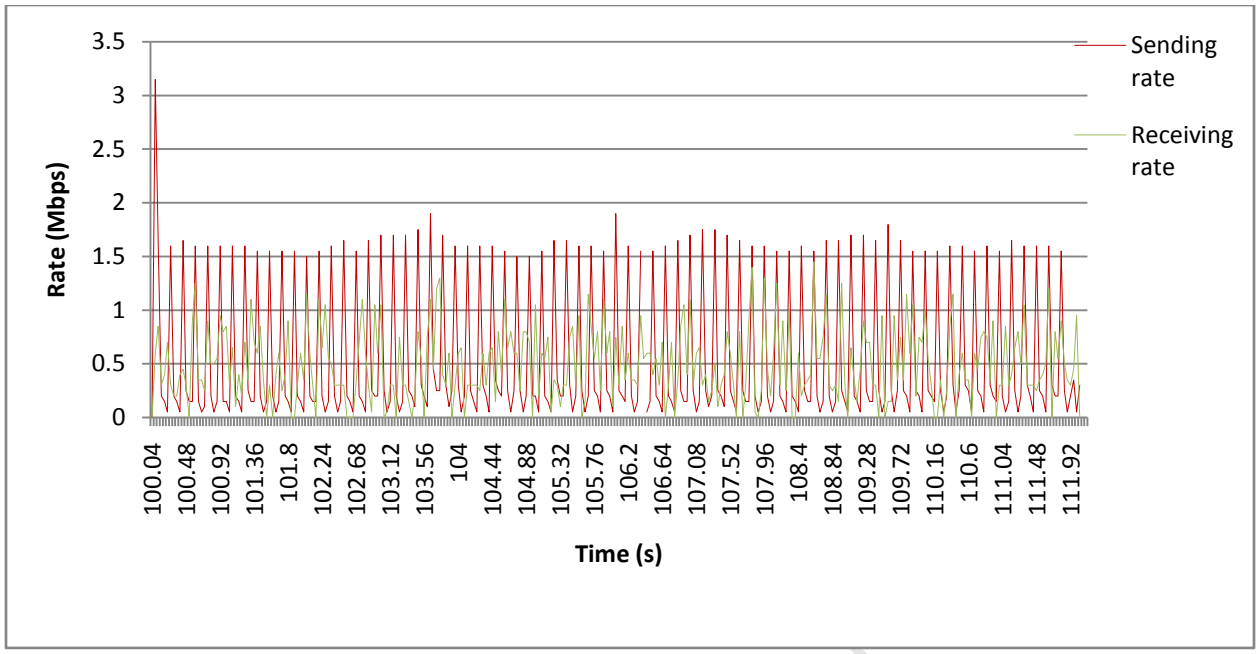

Figure 5.12 Sending rate versus receiving rates of News video when using EDF

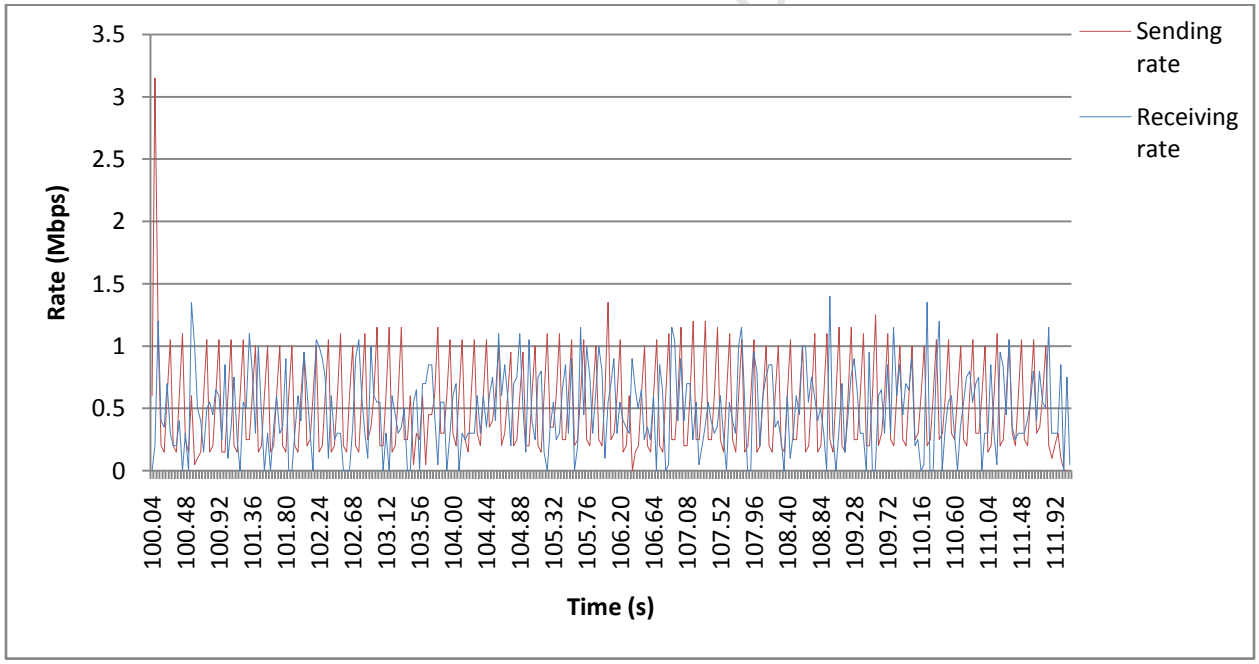

Figure 5.13 Sending rate versus receiving rates of News video when using pipelining-based scheduling algorithm

When comparing the sending and receiving bit rate variances between the pipelining-based streaming scheme and Earliest Deadline First, it is found that variances for pipelining-based streaming scheme are smaller than the corresponding variances for Earliest Deadline First.

Hence, the pipelining based video streaming scheme is bandwidth adaptive.

5.4.5 Frame Latency

The following three figures present frame end-to-end delays obtained when both pipelining-based and Earliest Deadline First were used to stream the three video sequences. The simulation setup used was similar to the one used in the preceding subsection. The x-axis represents the frame index while the y-axis represents the inter frame time in seconds.

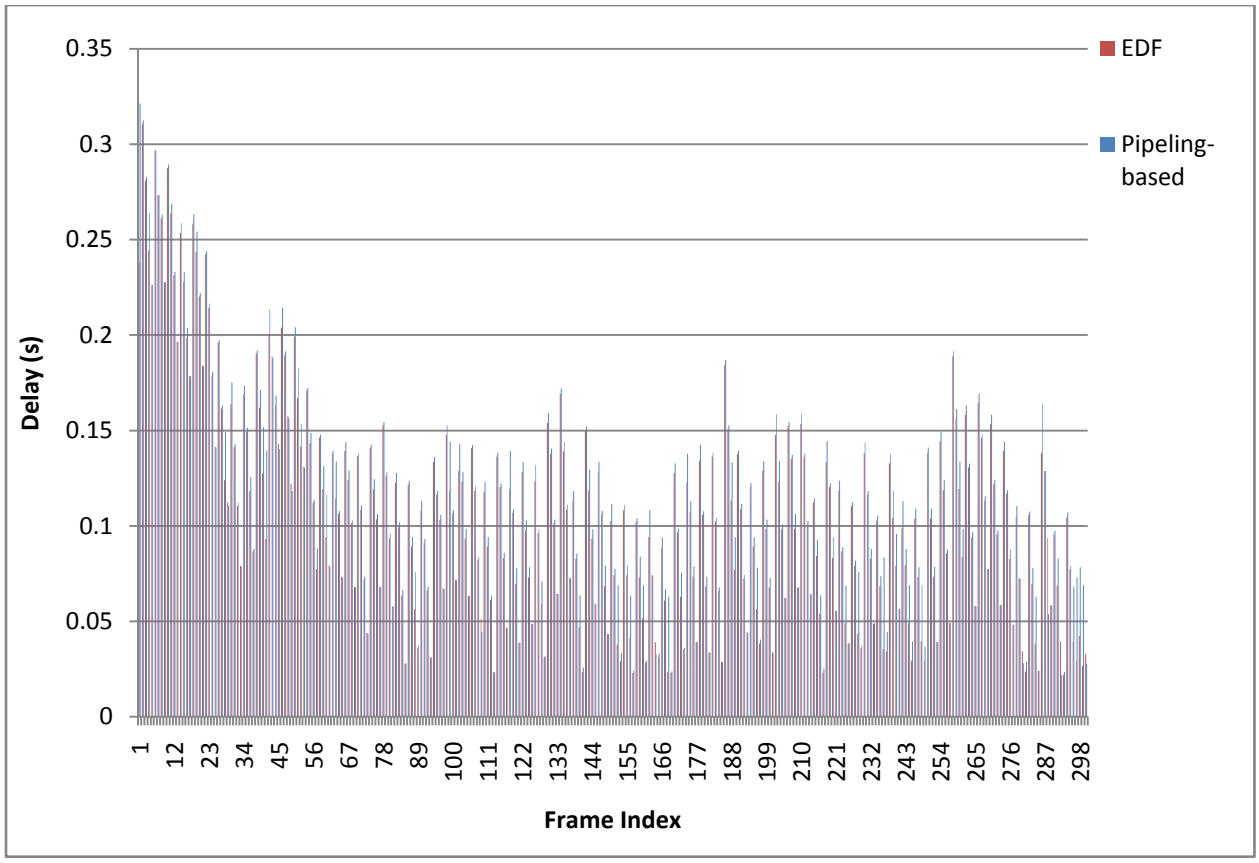

Figure 5.14 Frame delay for Paris video when scheduling video with EDF and pipelining-based scheduling

Figure 5.14 shows the end-to-end frame delay for Paris video when scheduling video with of the two video scheduling schemes. It can be observed that the pipelining-based algorithm increased frame delay compared to the Earliest Deadline First scheduling algorithm. However, the maximum frame delay was increased from 310ms to 320.7ms (about 3%), which when compared to other benefits of pipelined scheduling may be considered as a tradeoff. The most

significant contributing factor to higher frame delays exhibited when streaming Paris was the frame size. Since the test sequence had the highest bit rate, its frames experienced the highest delay.

Figure 5.15 illustrates end-to-end frame delay for the Foreman video bit stream using both pipelining-based and Earliest Deadline First scheduling algorithms.

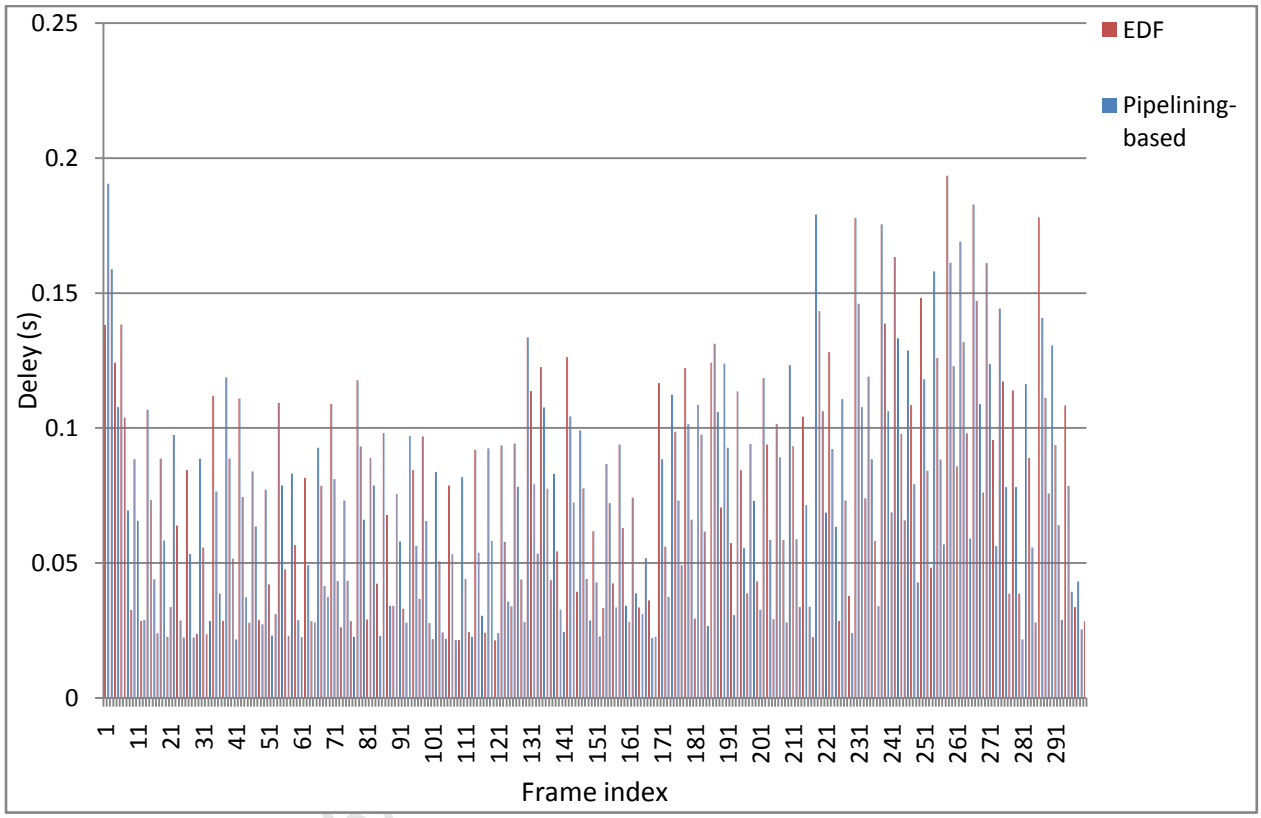

Figure 5.15 Frame delay for News video when scheduling video with EDF and pipelining-based scheduling

As in the previous case, pipelining-based scheduling has increased end-to-end frame delay. The maximum frame delay was increased from about 180ms to 190ms when pipelining-based scheduling was used. Comparing the overall end-to-end delay of Foreman to that of Paris, Foreman experienced a lower end delay because it is lower bit rate video.

Figure 5.15 illustrates end-to-end frame delay for the News video bit stream using both pipelining-based and Earliest Deadline First scheduling algorithms. Like in the previous cases, pipelining-based scheduling has increased the maximum frame delay by about 50ms. News is a lower bit rate and smaller frame sized bit stream compared to both Paris and Foreman video.

Hence, its frames experienced lower frame delays.

When assessing the performance of the pipelining-based video scheduling scheme with respect to end-to-end frame delays, it was found to perform better in higher bit rate video sequences. But, when compared to Earliest Deadline First, pipelining was found to have slightly higher delays, which were attributed to the fact that the pipelining scheduler used more than one period to schedule a frame as opposed to exactly one in the case of Earliest Deadline First. However, frame delays are addressed by the introduction of small delays in the video play-out process.

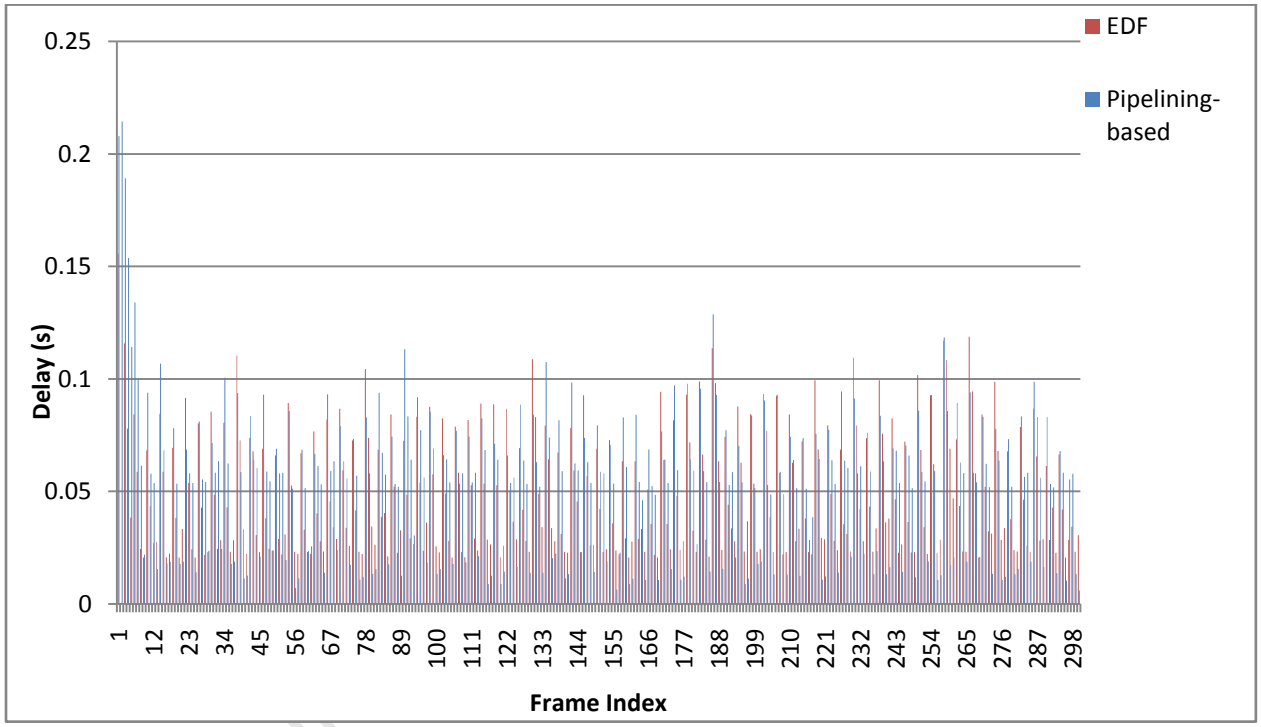

Figure 5.16 Frame delay for Paris video when scheduling video with EDF and pipelining-based scheduling

5.4.6 Frame jitter

Figure 5.17, Figure 5.18 and Figure 5.19 display cumulative jitter results for the three videos: Paris, Foreman and News, respectively. The graphs for each video sequence express the jitter exhibited under the same network conditions but different video scheduling algorithms. The

more erratic the curve, the more likely the frames will arrive at different times resulting in receiving buffer management challenges. The y-axis represents the jitter in seconds while the x-axis represents the video frames.

When comparing the jitter graphs to their corresponding delay graphs best conclusions about jitter can be drawn. The Paris video shown in Figure 5.17 shows the jitter curves for the pipelining-based and earliest deadline first scheduling algorithms. From the figure, it is evident that pipelining-based scheduling exhibits a slightly higher frame jitter. However, the increase is about 0.2ms. The peak jitter for pipelining-based was 4.7ms while it was 4.5 for Earliest Deadline First.

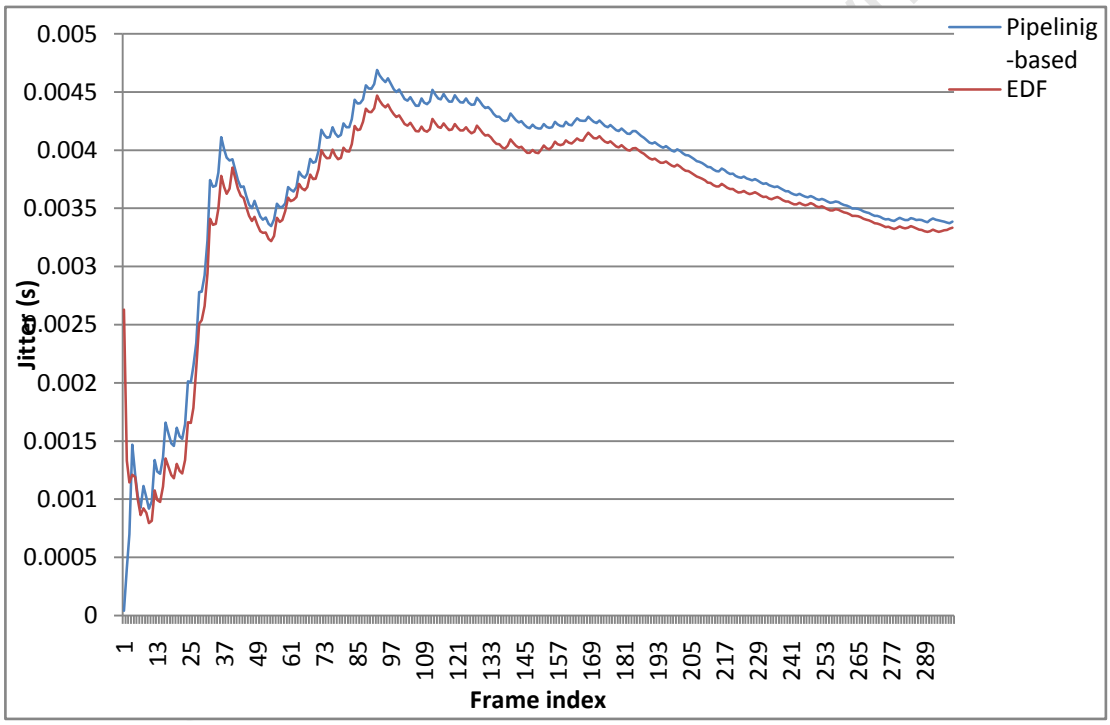

Figure 5.17 Jitter for Paris video

Figure 5.18 shows the jitter graphs for the Foreman video. At the beginning of the streaming session, Earliest Deadline First scheduling algorithm experienced a higher frame jitter but later on went below that of the pipelining-based scheduling algorithm. The peak jitter for Earliest Deadline First scheduling algorithm was 2.83ms while it was 2.68ms for pipelining-based scheduling algorithms.

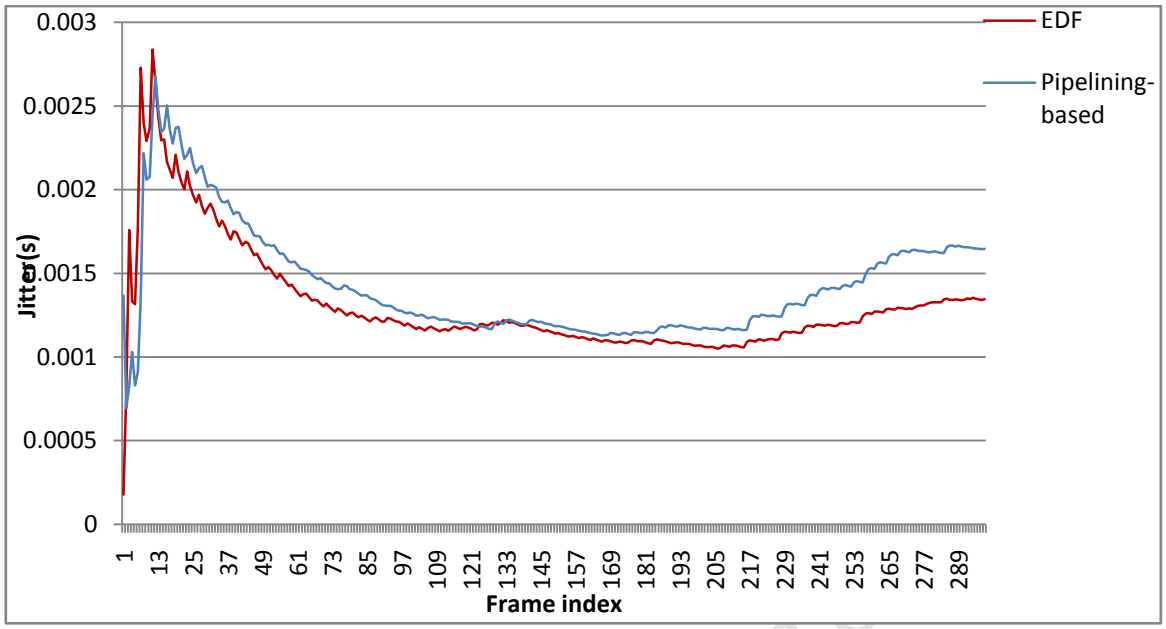

Figure 5.18 Jitter for Foreman video

Figure 5.19 depicts the cumulative jitter graphs for the News video sequence. Compared to the other video stream, News is a low motion video. Pipelining-based scheduled frames experienced higher jitter. The peak jitter for the pipelining-based scheduling algorithm was 4.5ms while it was 2.5ms for Earliest Deadline First scheduling algorithm.

Comparing the graphs for all videos, pipelining-based video streaming scheme proved to be more suitable for high motion (Foreman) and medium motion (Paris) videos. In both degrees of motion, Pipelining-based video streaming increased peak jitter by 0.2ms. The increase in jitter was due to interleaving of frame done during pipelining, resulting in some parts of the frames being scheduled late compared to Earliest Deadline First.

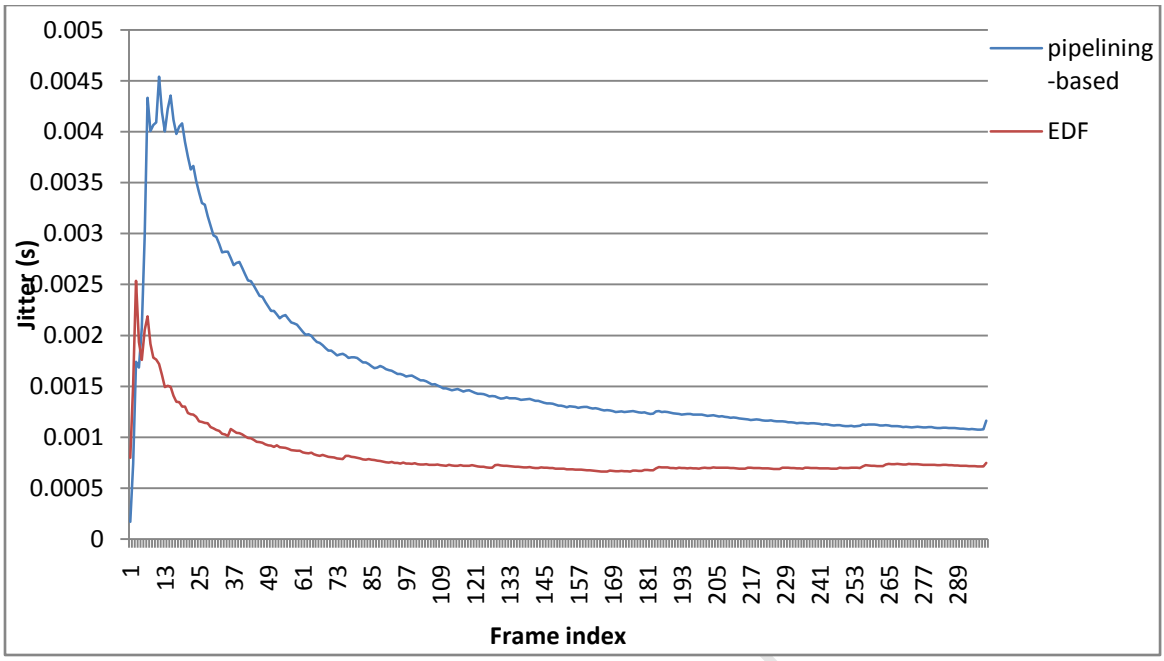

Figure 5.19 Jitter of News video

5.4.7 Available bandwidth

Figure 5.20 displays the bandwidth estimation graphs; one obtained using video traffic scheduled by the pipelining-based scheduler and the other by the cross-traffic. The y-axis represents the available bandwidth measured in bits per seconds (Mbps) while the x-axis represents the simulation time in seconds.

The cross-traffic bandwidth estimate was obtained by subtracting the throughput at the cross-traffic receiver from the know network capacity, 2.8Mbps. The throughput is computed every 40ms and the bandwidth estimate as well. The obtained bandwidth results are presented in Figure 5.20.

The variance between the differences of two graphs was 0.03973, which shows that the graphs are close to each other. Hence, pipelining-based bandwidth estimation method was to a certain extend accurate. The major contributing factor the variance in the two graphs is the packet size, decreasing the packet sizes for both types of traffic would yield more accurate bandwidth estimates.

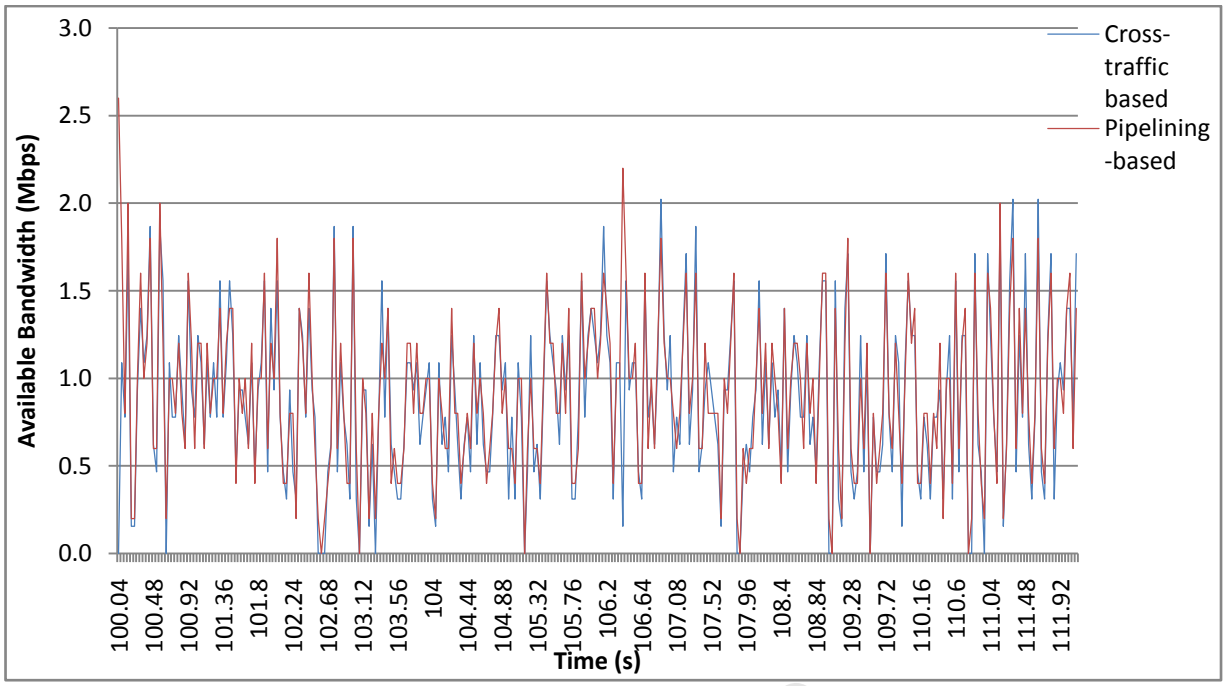

Figure 5.20 Available bandwidth estimated using cross-traffic and pipelining-based traffic

5.5 Summary

This chapter presented the results and analysis of the results carried out using the evaluation framework described in Chapter 4. Three video test sequences - Paris, Foreman and News - were transmitted over the Mobile WiMAX network simulated in NS-2. To investigate the performance of the Pipelining-based video streaming scheme - the most commonly used video streaming scheme, the Earliest Deadline First was used for comparison purposes.

The performance of the two streaming schemes was investigated using the following quality of service QoS) and quality of user experience (QoE) metrics: packet loss, peak-signal-to-noise ratio, frame loss, traffic burstiness, frame latency, and frame jitter. The pipelining-based video streaming scheme was found to reduce packet loss compared to the Earliest Deadline First. In addition to reduced packet loss, pipelining-based streaming scheme offered traffic smoothing and thus reduced SVC traffic variability. This proved the pipelining-based streaming scheme bandwidth efficient. The scheme also offered improved QoE by offering less frame loss and higher PSNR average values with smaller standard deviation values compared to the Earliest Deadline First.

The adaptability of the streaming server to the available bandwidth was investigated using the metric referred to as rate adaptation. The pipelining-based streaming server was found to be bandwidth adaptive compared to the Earliest Deadline First server, which was also confirmed by a decrease in packet loss in comparison with the Earliest Deadline First.

In terms of end-to-end frame delay and frame jitter, the pipelining-based video streaming scheme was found to increase end-to-end frame delay by about 3%, which was not a major problem since the receiver side buffering can offset end-to-end frame delay. The values were within the ranges recommended by the ITU-T. In most of the metrics used, except end-to-end frame delay, the pipelining-based proved to provide the best results for highly variable traffic. Therefore, the results proved the pipelining-based video streaming scheme to be suitable for SVC video streaming over Mobile WiMAX networks.

Chapter 6 Conclusions and Recommendations for Future Work

6.1 Concluding remarks

This thesis was aimed at investigating the challenges of video streaming over Mobile WiMAX networks. The major challenge for video steaming over Mobile WiMAX has been found to be time varying transmission bandwidth. This is due to numerous factors, such as congestion, change in signal strength, and handoff. However, the problem addressed in this study was bandwidth fluctuation in general, which occurs during a streaming session in the Mobile WiMAX network. Normally, this problem results in packet loss, hence visual video quality degradation.

To address this problem, an adaptive video streaming scheme was proposed and implemented. The proposed scheme focused mainly on producing an adaptive video server which scaled video to the available bandwidth in order to minimise packet loss while optimising visual video quality. The video packet scheduling of the proposed adaptive video streaming server has been based on the pipelining design technique and was compared to the default, Earliest Deadline First video streaming scheme. To adapt to the available bandwidth, the streaming server scaled the video bit stream based on the bandwidth signalling information from the video receiver.

The proposed pipelining-based video streaming scheme had the following advantages over the Earliest Deadline First (EDF):

- The pipelining-based streaming scheme significantly reduces packet loss when user transmission bandwidth fluctuates during a streaming session. This is made possible by bandwidth monitoring process, carried out by the server and the receiving client. The streaming server uses the output of the bandwidth monitoring process, that is, the bandwidth feedback information used to increase or decrease the sending rate depending on the available bandwidth.
- The scheme was found to be bandwidth efficient. Bandwidth efficiency is fostered by traffic smoothing, achieved through pipelining of video frames. The use of pipelining

in video scheduling was found to reduce the traffic burstiness by at least 20%. Apart from reducing traffic burstiness, scaling the bit streaming at the streaming server avoids packets being lost along the transmission path. For instance, if packets were lost at the last segment, those packets would have consumed bandwidth to reach the bottleneck where they are lost and eventually fail to contribute to the visual quality.

Another important issue is one of the characteristics of SVC video, which allows only completely received network abstraction layer units (NALUs) to be decoded. Because of this property, the server schedules complete NALUs only when available bandwidth permits scheduling of parts of an NALU. Therefore, bandwidth is saved that could otherwise have been used to send packets that do not contribute to the visual video quality. However, this idea was not fully exploited during the implementation due to the simulation tools limitations.

The pipelining-based streaming scheme offered better visual video quality compared to the EDF streaming scheme. This was attributed to the use of pipelining in video scheduling. Pipelining allowed prioritisation of NALUs based on their impact on video quality. During the streaming session, the streaming server dropped lower priority NALUs when available bandwidth was not sufficient for all NALUs. As a result, even when throughput for a video that was scheduled using two schemes was the same, the visual quality of the received video was different due to pipelining.

One of the setbacks of the proposed scheme was the increase in frame delay and frame jitter. However, the increase was not significant. In summary, pipelining has proved to be an instrumental method to offer bandwidth adaptation when steaming video over the Mobile WiMAX networks.

6.2 Recommendations for future work

During the course of this study, a number of interesting ideas have surfaced but could not be included in this research project due to time limitation and the scope of this project. The following issues can be considered for further work on this study:

- Improve the bandwidth monitoring mechanism. The performance of the pipelining-based streaming scheme, in terms of packet loss, depends on the accuracy of the bandwidth monitoring mechanism. The mechanism used in this study had a number of shortcomings; as a result improving it or using one that is more accurate will further decrease packet loss.
- Investigate optimum pipelining cycle. The size of the pipelining cycle can affect the adaptability of the pipelining-based video streaming scheme. The short pipelining cycle size results in a fast reacting system, which reduces packet losses but may compromise video quality as at times the higher priority NALUs can be dropped. Whereas a longer pipelining cycle yields the opposite in that high priority NALUs can be spared with ease, but might increase packet loss since the system will take time to adjust the sending rate to the available bandwidth.
- Implement the proposed scheme on the test bed. Simulations are an effective tool for the prediction of network performance. They are however limited in that they often do not take into account the real-world factors and do not accurately reflect the behaviour of the physical networks.

References

[1] WiMAX Forum, Mobile WiMAX – Part I: A Technical Overview and Performance Evaluation, 08/2006 [online], Available : http://www.wimaxforum.org/resources/documents/technical/release

[2] WiMAX Forum, "WiMAX a way forward in India," 2010. [Online] Available: http://www.wimaxforum.org/resources/documents/technical/release

[3] A. Kumar, *Mobile Broadcasting with WiMAX : Principles, Technology, and Applications.* Amsterdam; Paris: Elsevier, 2008.

[4] World Cellular Information Services. WiMAX deployments. *2011(02/07/2011),* 2011.

[5] W. Simpson, *Video Over IP.* Burlington, MA: Focal Press, 2008.

[6]CNEWS. Streaming video to outpace P2P traffic growth. *07/2011*[Online] Available: http://news.cnet.com/8301-30686_3-20006530-266.html

[7] J. Casasempere, P. Sanchez, T. Villameriel and J. Del Ser, "Performance evaluation of H.264/MPEG-4 scalable video coding over IEEE 802.16e networks," in *2009 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, BMSB 2009, may 13, 2009 - may 15,* 2009, .

[8] H. Hellwagner, I. Kofler, M. Eberhard, R. Kuschnig, M. Ransburg and M. Sablatschan, "Scalable Video Coding," 2011.

[9] H. Sun, A. Vetro and J. Xin, "An overview of scalable video streaming," *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING,* vol. 7, pp. 159-172, 2007.

[10] O. I. Hillestad, A. Perkis, V. Genc, S. Murphy and J. Murphy, "Adaptive H.264/MPEG-4 SVC video over IEEE 802.16 broadband wireless networks," in *Packet Video 2007,* 2007, pp. 26-35.

[11] H. Juan, H. Huang, C. Huang and T. Chiang, "Cross-layer system designs for scalable video streaming over mobile WiMAX," in *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE,* 2007, pp. 1860-1864.

[12] T. A. Le, H. Nguyen and H. Zhang, "EvalSVC — an evaluation platform for scalable video coding transmission," in *Consumer Electronics (ISCE), 2010 IEEE 14th International Symposium on,* 2010, pp. 1-6.

[13] M. Tran, G. Zaggoulos, A. Nix and A. Doufexi, "Mobile WiMAX: Performance analysis and comparison with experimental results," in *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th,* 2008, pp. 1-5.

[14] L. Lin, W. Jia and W. Lu, "Performance analysis of IEEE 802.16 multicast and broadcast polling based bandwidth request," in *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE,* 2007, pp. 1854-1859.

[15] B. Li, Y. Qin, C. P. Low and C. L. Gwee, "A Survey on Mobile WiMAX [Wireless Broadband Access]," *Communications Magazine, IEEE,* vol. 45, pp. 70-75, 2007.

[16] WiMAX Forum, Recommendations and Requirements for Networks based on WiMAX Forum Certified Products, 02/2006, [Online] Available: http://www.wimaxforum.org/resources/documents/technical/

[17] C. So-In, R. Jain and A. Tamimi, "Scheduling in IEEE 802.16e mobile WiMAX networks: key issues and a survey," *Selected Areas in Communications, IEEE Journal on,* vol. 27, pp. 156-171, 2009.

[18] N. Kamaci and Y. Altunbasak, "Performance comparison of the emerging H.264 video coding standard with the existing standards," in *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on,* 2003, pp. I-345-8 vol.1.

[19] Joint Video Team, "JSVM software manual," 2009.

[20] D. Wu, Y. T. Hou and Y. Zhang, "Scalable video coding and transport over broadband wireless networks," *Proceedings of the IEEE,* vol. 89, pp. 6-20, 2001.

[21] H. Schwarz, D. Marpe and T. Wiegand, "INVITED PAPERS - Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," *IEEE Transactions on Circuits and Systems for Video Technology : A Publication of the Circuits and Systems Society.,* vol. 17, pp. 1103, 2007.

[22] I. E. G. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia* (1st ed.) 2003.

[23] T. C. Thang, J. Kim, J. W. Kang and J. Yoo, "SVC adaptation: Standard tools and supporting methods," *Signal Process Image Commun,* vol. 24, pp. 214-228, 3, 2009.

[24] P. Malumba, "A Common Analysis Framework for Simulated Streaming-Video Networks", 2009., Port Elizabeth: Rhodes University

[25] ISO/IEC JTC1/SC29/WG11, " Overview of Scalable Video Coding," 2008.

[26] D. Wu, Y. T. Hou, W. Zhu, Y. Zhang and J. M. Peha, "Streaming video over the Internet: approaches and directions," *Circuits and Systems for Video Technology, IEEE Transactions on,* vol. 11, pp. 282-300, 2001.

[27] B. Shen, W. Tan and F. Huve, "Dynamic Video Transcoding in Mobile Environments," *Multimedia, IEEE,* vol. 15, pp. 42-51, 2008.

[28] S. Coulombe and G. Grassel, "Multimedia adaptation for the multimedia messaging service," *Communications Magazine, IEEE,* vol. 42, pp. 120-126, 2004.

[29] Pengye Xia, S. -. G. Chan and Xing Jin, "Optimal Bandwidth Assignment for Multiple-Description-Coded Video," *Multimedia, IEEE Transactions on,* vol. 13, pp. 366-375, 2011.

[30] D. Miras, "A survey of network QoS needs of advanced internet applications -working document," in 2002, .

[31] G. Sun, U. Samarawickrama, J. Liang, C. Tian, C. Tu and T. D. Tran, "Multiple Description Coding With Prediction Compensation," *Image Processing, IEEE Transactions on,* vol. 18, pp. 1037-1047, 2009.

[32] J. Kim, T. Um, W. Ryu and B. S. Lee, "IPTV Systems, Standards and Architectures: Part II - Heterogeneous Networks and Terminal-Aware QoS/QoE-Guaranteed Mobile IPTV Service," *Communications Magazine, IEEE,* vol. 46, pp. 110-117, 2008.

[33] W. Stallings, *Computer Organization and Architecture : Designing for Performance.* Upper Saddle River, NJ: Pearson/Prentice Hall, 2006.

[34] J. L. Hennessy, D. A. Patterson, D. Goldberg, *Computer Architecture: A Quantitative Approach.* 2003.

[35] T. Issariyakul , E. Hossain, *Introduction to Network Simulator NS2.* New York: Springer, 2008.

[37] National Institute of Standards and Technology, Simulation Models for NS-2, [Online] Available: http://www.nist.gov/itl/antd/emntg/ssm_tools.cfm

[38] B. Rathke and J. Klaue, "EvalVid - A Framework for Video Transmission and Quality Evaluation," *Computer Performance,* vol. pp, pp. 255-272,.

[39] GPAC, "MP4Box," vol. 2011, .

[40] U. Arizona State, Video traces research group, Arizona State University, 08 2011. [Online]. Available: http://trace.eas.asu.edu/index.html

[41] ITU-T RECOMMENDATION P.910, "Subjective Video Quality Assessment methods for multimedia applications," 9/1999, 2000.

[42] Wikipedia, Peak signal to noise ratio, Wikipedia, the free encyclopaedia, 04 2011. [Online]. Available: http://en.wikipedia.org/wiki/PSNR.

[43] Vidyo, Wikipedia,06/09/11 [Online] Available : http://en.wikipedia.org/wiki/Vidyo

[44] J. Curtis and T. McGregor, “Review of bandwidth estimation techniques,” in Proc. New Zealand Computer Science Research Students’ Conf., vol. 8, New Zealand, Apr. 2001

Appendix A

Accompanying CD-ROM

- **Thesis hard-copy**

  An electronic copy of this thesis can be found in the "Thesis" directory.

- **Software**

  All the software (source code and binary files) that was used to develop the evaluation framework is located in the "Software" directory.

- **Simulation scripts and the video sequences**

  Simulation scripts and the video sequences and located in the "Simulations" directory, which is organised according to the following directory structure:

Paris	Foreman	News
Earliest_deadline_first	Earliest_deadline_first	Earliest_deadline_first
Pipelining_based	Pipelining_based	Pipelining_based

- **Results and evaluation scripts**

  Results and evaluation scripts are located in the "Results" directory. The MS excel files and Virtual Basic for Applications are used for manipulation of the trace files produced by the simulation scripts.

- **Manual**

  A small-scale user manual has been provided and is located in the "Simulations" directory.