Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática,
2009

André Braga Reis

Escalonamento de Pacotes Áudio e Vídeo em Redes WiMAX em Malha com QoS

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática,
2009

André Braga Reis

QoS-based Audio and Video Packet Scheduling over WiMAX Mesh Networks

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e de Telecomunicações, realizada sob a orientação científica de Prof.ª Dr.ª Susana Sargento, Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro; de Prof. Dr. Francisco Fontes, Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro; e de Prof. Dr. Andreas J. Kassler, Computer Science Department, Universidade de Karlstad, Suécia.

o júri / the jury

presidente / president	**Prof. Dr. Paulo Miguel Nepomuceno Pereira Monteiro** Professor Associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro
orientador / adviser	**Profª. Drª. Susana Isabel Barreto de Miranda Sargento** Professora Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro
co-orientador / co-adviser	**Prof. Dr. Francisco Manuel Marques Fontes** Professor Auxiliar Convidado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro
arguente / examiner	**Prof. Dr. Joel José Puga Coelho Rodrigues** Professor Auxiliar do Departamento de Informática da Universidade da Beira Interior

agradecimentos / acknowledgements

Ao concluir mais um importante passo na minha vida, é com enorme satisfação que aqui deixo sinceros agradecimentos a todos os que, directa ou indirectamente, contribuíram para a realização desta dissertação e me apoiaram ao longo do meu percurso académico.

À Professora Dra. Susana Sargento, pelo excelente espírito crítico, pelo apoio incondicional, pela permanente disponibilidade e pelo forte incentivo à realização de um trabalho de excelência.

Ao Professor Dr. Andreas Kassler, da Universidade de Karlstad, por todo o tempo dedicado à orientação de grande parte desta dissertação, e por todo o auxílio prestado durante a minha estadia em Karlstad.

Ao Dr. Jacob Chakareski, do Swiss Federal Institute of Technology, pelo indispensável auxílio nos temas de processamento e estimação da qualidade de conteúdos de vídeo.

A todos os meus amigos e colegas, e em especial ao André Cardote, à Cláudia Sequeira, ao António Costa e ao Rui Botte, pelo excelente espírito de camaradagem e amizade incondicional que demonstraram ao longo de todos estes anos.

E por fim, mas não menos importante, um sincero muito obrigado aos meus pais, sem os quais todo este percurso não teria passado de uma mera utopia.

palavras-chave IEEE 802.16, WiMAX, Malha, QoS, QoE, MOS, Escalonamento, Redes Sem Fios, Redes Metropolitanas, WMAN

Resumo

É evidente na actualidade que os utilizadores procuram cada vez mais aceder a serviços multimédia e aplicações interactivas nos seus terminais móveis. Há, portanto, uma necessidade de implementar arquitecturas de Qualidade-de-Serviço (QoS) e Qualidade-de-Experiência (QoE) robustas, que sejam capazes de fornecer um atraso baixo para as aplicações interactivas, ao mesmo tempo que lidam com outro tipo de aplicações que requerem uma maior largura de banda, mas com mais tolerância a atrasos, desta forma maximizando a utilização dos recursos disponíveis na rede e melhorando a experiência do utilizador final.

A norma IEEE 802.16 representa uma das tecnologias mais avançadas e de maior relevância para o acesso sem fios em banda larga a redes de área metropolitana. O modo de operação Ponto-multiponto (PMP) do IEEE 802.16 foi desenvolvido para suportar requisitos de QoS, controlável pelo operador da rede, e desta forma complementando as soluções móveis de terceira-geração já existentes. Um modo alternativo de operação em malha (MESH) permite a criação de redes flexíveis e auto-configuráveis em que o tráfego é encaminhado através de vários nós.

Esta tese aborda os temas de QoS e QoE quando aplicados a redes sem fios em malha, operando sobre a norma IEEE 802.16. São contribuídos melhoramentos e análises de desempenho a uma nova arquitectura para trazer suporte de QoS ao modo de operação MESH do standard 802.16. Também é apresentado um novo escalonador de pacotes com o objectivo de melhorar a qualidade subjectiva de serviços de áudio, vídeo e transferência de ficheiros que o utilizador final experiencia.

Os resultados provenientes de simulações demonstram tanto a eficiência da arquitectura QoS em termos de medidas objectivas como taxa de transferência e atraso de pacotes, como o bom funcionamento do escalonador de pacotes para QoE, com melhorias visíveis em métricas de qualidade subjectiva.

keywords IEEE 802.16, WiMAX, Mesh, QoS, QoE, MOS, Scheduling, Wireless Networks, Metropolitan Networks, WMAN

Abstract

It is clear nowadays that users are becoming increasingly interested in accessing multimedia and interactive applications on their mobile terminals. Therefore, there is a need to implement robust Quality-of-Service (QoS) and Quality-of-Experience (QoE) architectures capable of providing low delay for such interactive applications, while at the same time dealing with other bandwidth-hungry but more delay-tolerant services, and thereby maximizing the network's available resources and improving the end-user experience.

The IEEE 802.16 standard represents one of the most relevant and advanced technologies for broadband wireless access in metropolitan area networks. The point-to-multipoint (PMP) mode of IEEE 802.16 has been designed to support quality of service (QoS) requirements, controlled by the network operator, thus complementing the existing third-generation mobile solutions. An alternative mesh (MESH) mode of operation allows the creation of flexible, self-configuring networks with traffic routing through various nodes.

This thesis approaches the subjects of QoS and QoE when applied to wireless mesh networks operating under the IEEE 802.16 standard. It provides improvements and performance evaluations of a new architecture to bring QoS support to the 802.16 MESH mode of operation. It also presents a new packet scheduler with the aim to improve the subjective quality of audio, video and file transfer services, as experienced by the end user.

Simulation results demonstrate both the efficiency of the QoS architecture in terms of objective measurements such as throughput and packet delay, and the good functioning of the QoE-aware packet scheduler, with noticeable increases in subjective quality metrics.

Contents

List of Figures

List of Tables

Acronyms

3G	Third Generation (Mobile Telephone Standard)
ASU	Arizona State University
BE	Best Effort
BPSK	Binary Phase Shift Keying
BS	Base Station
BWA	Broadband Wireless Access
CBR	Constant Bit Rate
CID	Connection Identifier
CIF	Common Intermediate Format
CPE	Consumer Premises Equipment
CPS	Common Part Sublayer
CS	Convergence Sublayer
DSL	Digital Subscriber Line
ertPS	extended-real-time Polling Service
ETSI	European Telecommunications Standards Institute
FTP	File Transfer Protocol
FDM	Frequency Division Multiplexing
FTTC	Fiber-To-The-Curb
FTTH	Fiber-To-The-Home
GoP	Group Of Pictures
GPRS	GSM Packet Radio Services
GSM	Global System for Mobile Communications
GSM-AMR	GSM Adaptive Multi-Rate (Voice Coder)
HiperMAN	High Performance Radio Metropolitan Area Network
HSDPA	High-Speed Downlink Packet Access
HSPA+	High-Speed Packet Access Plus
ICMP	Internet Control Message Protocol
IE	Information Element
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ITU	International Telecommunication Union
LOS	Line-Of-Sight
MA	Multiple Access
MAC	Medium Access Control (OSI Layer)
MESH	Mesh Mode
MIB	Management Information Base
MOS	Mean Opinion Score
MPEG	Motion Picture Experts Group
MR-WMN	Multi-Radio Wireless Mesh Network
MSE	Mean Square Error
MSH-CSCH	Mesh Centralized Schedule message
MSH-DSCH	Mesh Distributed Schedule message

NLOS	Non Line-Of-Sight
nrtPS	non-real-time Polling Service
OFDM	Orthogonal Frequency Division Multiplex
OFDMA	Orthogonal Frequency Division Multiple Access
PDU	Protocol Data Unit
PHY	Physical (OSI Layer)
PMP	Point-to-Multipoint
PSNR	Peak Signal-to-Noise Ratio
QAM	Quadrature Amplitude Modulation
QCIF	Quarter Common Intermediate Format
QoE	Quality of Experience
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RSVP	Resource Reservation Protocol
rtPS	real-time Polling Service
SDU	Service Data Unit
SS	Subscriber Station
SSIM	Structural Similarity Index
SOFDMA	Scalable Orthogonal Frequency Division Multiplex
SVC	Scalable Video Coding
TCL	Tool Command Language
TOS	Type Of Service
UDP	User Datagram Protocol
UGS	Unsolicited Grant Service
UMTS	Universal Mobile Telephone System
VBR	Variable Bit Rate
VOD	Video-on-Demand
VoIP	Voice over Internet Protocol
WiFi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access
WMN	Wireless Mesh Network

"I do not think that the wireless waves I have discovered will have any practical application."

– Heinrich Rudolf Hertz

Chapter 1

Introduction

The advent of ubiquitous wireless broadband, driven in a strong way by second- and third-generation mobile standards, has sparked great interest in the research and development of broadband wireless access technologies. Motivated by a strong demand by users and service providers alike, innovation and investment in these wireless technologies has too been strongly pushed forward. An issue faced by these new technologies is the need to cover as large an area as possible with a unique wireless network, in order to reduce the deployment costs as well as to be able to provide last-mile wireless broadband service. Such a technology would allow cheap and rapid coverage of large, sparsely-populated areas, as well as to improve mobility in densely-populated ones.

The predominant broadband access technologies nowadays are DSL (Digital Subscriber Line), which delivers broadband over twisted-pair telephone wires, and Cable technology, which is supported over coaxial cables designed for television broadcasting. Both of these are mostly capable of up to 20 Mbps of downlink bandwidth to each user. In rapid expansion are the fiber-optical technologies, with governmental and corporate initiatives driving the deployment of large-scale Fiber-To-The-Home (FTTH) and Fiber-To-The-Curb (FTTC) networks. These networks are already capable of delivering gigabit bandwidths to the end-user, a 50x to 100x improvement over DSL and Cable, enabling the delivery of not only Internet access but also high-definition television and video-on-demand, digitally.

These technologies, however, are of fixed access. User mobility and a large demand for broadband access in mobile terminals pushed for the creation of broadband wireless access (BWA) technologies to satisfy the need for ubiquitous availability of broadband service. The current technologies are implemented on top of cellular network protocols such as GSM, GPRS, and UMTS, with maximum data rates of 7.2 Mbps in UMTS networks with HSDPA support[1]. One of the technologies that shows much promise for broadband wireless access is the IEEE 802.16 family of standards, commonly dubbed "WiMAX" by the WiMAX Forum industry group. The main features of this standard include data rates of up to 70 Mbps and transmission ranges of up to 20 km [1]. This, coupled with its wireless nature and ease of deployment, make WiMAX a promising new technology for providing BWA.

[1]HSDPA supports a maximum of 14 Mbps of downlink throughput, and further speed increases are available with the emerging HSPA+ standard, which provides downlink speeds of up to 42 Mbps.

1.1 Motivation

It is clear nowadays that users are becoming increasingly interested in accessing multimedia and interactive applications on their mobile terminals. Therefore, there is a need to implement robust QoS architectures capable of providing low delay for such interactive applications, while at the same time dealing with other bandwidth-hungry but more delay-tolerant services, and thereby maximizing the network's available resources and improving the end-user experience.

On the topic of Quality-of-Service, the IEEE 802.16 standard does not clearly define an architecture and the methods required to ensure strict QoS-based packet scheduling for networks operating in the MESH mode. The standard defines no information exchange about the data scheduling service associated with each type of traffic; consequently, this does not allow a node to estimate the specific characteristics and needs of the traffic which it is forwarding.

The QoS mechanisms for the PMP mode of operation, on the other hand, are well established and define a number of service classes with specific methods to ensure and meet strict QoS requirements. This inconsistency between QoS specifications for the IEEE 802.16 PMP and MESH modes also limits the integration of networks operating on different modes, due to a lack of a common set of service classes. We are thus prompted, in this thesis, to investigate, develop and improve architectures which can both bring QoS support to the 802.16's MESH mode and facilitate its integration with the PMP mode, by defining compatible sets of service classes.

Quality-of-Service alone, however, cannot guarantee a good user experience, especially where multimedia applications are concerned. End-user experience for multimedia applications cannot be judged by common metrics of network efficiency, and new techniques have been developed in the past few years that aim to evaluate the real satisfaction as perceived by the user. Along with these new techniques, research has led to the development of packet schedulers that no longer aim on maximizing the network throughput, but instead try to maximize the combined end-user experience. The majority of this research, however, is focused towards a single application target, such as maximizing quality for all video flows or for all VoIP flows. We intent on researching methods to maximize the end-user experience in a combined fashion for audio, video, and file transfer applications, while also keeping in mind the concept of mesh networks and how to optimize these methods to meet the specific characteristics of these networks.

This thesis thus aims to narrow the gap between three major topics of research: wireless mesh networks, quality-of-service, and quality-of-experience.

1.2 Objectives

The main goals of this thesis are to research mechanisms for providing Quality-of-Service and Quality-of-Experience in IEEE 802.16 networks, and in particular their application in wireless mesh networks.

On the topic of Quality-of-Service, we aim to improve an architecture for providing QoS service classes for the IEEE 802.16 MESH mode of operation, and then do a thorough evaluation on the performance of this architecture and its shortcomings.

On the topic of Quality-of-Experience, we propose a new packet scheduler designed with the goal of obtaining the highest possible quality for audio, video and file transfer flows, as experienced by the end user. We also look into how this scheduler can be designed around mesh networks, in order to deal with their particular multi-hop nature.

Combined, these two approaches provide mechanisms for improving both the objective and subjective characteristics of traffic flows in wireless mesh networks.

1.3 Contributions of this Thesis

The research performed in this work resulted in the following contributions:

- An improved QoS architecture for IEEE 802.16's MESH mode of operation, which shares the same service classes as those defined for the PMP mode, with a functional and thoroughly tested simulation environment.
- A QoE-aware packet scheduler for multiple audio, video and data streams, optimized for wireless mesh networks, which can be easily adapted to suit a number of scenarios and specific requirements, plus a software simulator for this scheduler's operation in 802.16 mesh networks.
- Integration of the QoS architecture with the QoE-aware scheduler, and consequent performance evaluation.

1.4 Outline

This thesis is structured as follows. In Chapter 2, we first do a brief introduction on the IEEE 802.16 standard, and then provide a more thorough description on the topics most relevant to this work: the MESH mode of operation in IEEE 802.16, and the mechanisms for Quality-of-Service support which are specific to that mode. Then, we introduce the concept of subjective quality measurement, the Mean Opinion Score metric, and detail the existing mechanisms for subjective measurement of audio and video quality. In Chapter 3, we elaborate on the topic of providing QoS mechanisms for the IEEE 802.16 MESH mode, our improvements to the existing mechanisms and the performance results of such improvements. In Chapter 4 we present a new QoE-based packet scheduler for wireless mesh networks, the models used for computing the MOS of audio, video, and file transfers, our simulator implementation of the scheduler, and the performance evaluation of the implementation, as well as the results of integrating the QoS architecture and the QoE-based scheduler. Finally, in Chapter 5 we conclude this thesis and provide guidelines for future research.

Chapter 2

Background

In this chapter we shall introduce the topics, standards and tools which are the groundwork for this thesis. In Section 2.1 we describe the IEEE 802.16 Standard and, in particular, its modes of operation for wireless mesh networks; Section 2.2 introduces the concept of subjective evaluation of multimedia service quality as perceived by the end user, and the existing methods for such measurements; and in Section 2.3 we talk about the *ns2* simulator and the *ns2mesh* patch for it, which implements support for 802.16's MESH mode, and upon which we implement and simulate all the work in this thesis. Each section also includes references to all of the related work, papers, books and technical reports related to the topics being approached in this thesis.

2.1 The IEEE 802.16 Standard

IEEE 802.16 is a family of IEEE standards for the creation of high data rate wireless metropolitan networks (WMANs). A WMAN is defined as a data network that can cover up to several kilometers, typically spanning an entire city. Figure 2.1 illustrates the existing categories of wireless networks, sorted by their span.

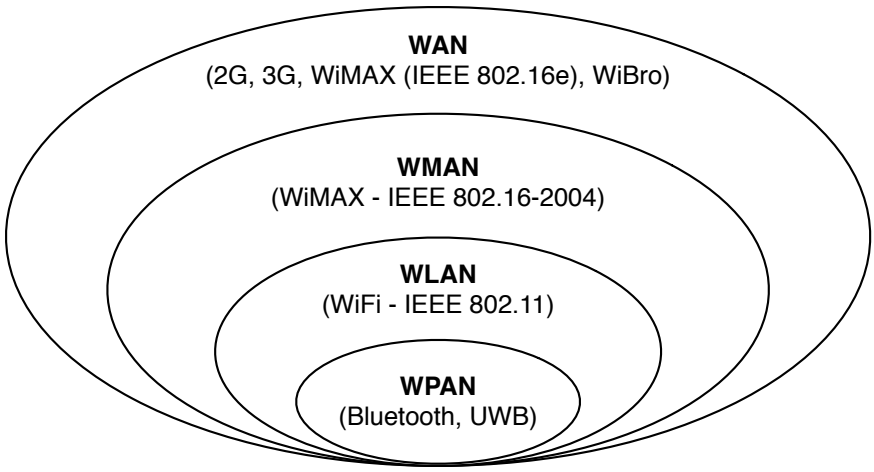


Figure 2.1: Illustration of network nomenclatures according to span

IEEE 802.16 is considered a broadband wireless access (BWA) system, due to its support for high data rates. BWA systems have a number of applications in telecommunication systems. First and foremost, they can be used to provide fixed-position broadband access. Such a high-throughput service can be used for voice communication (effectively doubling as a telephony system), high-speed Internet access, television, and video-on-demand. In this fashion a BWA can take the place of other broadband systems such as DSL and cable, while dispensing the need for the physical deployment of infrastructure such as telephone or coaxial cabling. This is particularly useful for providing broadband access in sparsely-populated regions, where the cost associated with running cables to each building would be prohibitively expensive. It is in these scenarios that the 802.16 family of standards poses itself as a competitor for fixed-line broadband providers in rural and urban environments.

Other BWA applications include WiFi backhauling, where the BWA network services a number of Consumer Premises Equipments (CPE), which is essentially a radio equipment capable of connecting to the network's BS and sharing the access to another equipment or a user terminal. On these CPEs, an 802.11 WiFi router can be attached to provide WLAN-sized service to a location, hence the denomination of WiFi backhauling. BWA can also be used for Telemetering[1] (due to its large reach), nomadic Internet access, and with proper support by the standards, mobile broadband access.

Evolution of the 802.16 standards

The IEEE 802.16 working group has authored a number of documents and major versions of 802.16. Table 2.1 summarizes some of the 802.16 standards released up to the present day [2]. The working group for BWA was created in 1999, and divided into two subgroups

Table 2.1: Description and status of IEEE 802.16 standards

Standard	Description	Status
802.16-2001	Fixed Broadband Wireless Access (10–63 Ghz)	Superseded
802.16a-2003	2-11 GHz PHY, non-Line-Of-Sight	Superseded
802.16-2004	Consolidation of previous standards	Current
802.16e-2005	Mobility, OFDMA and SOFDMA	Superseded
802.16k-2007	Bridging of 802.16 standards	Current
P802.16i	Mobile Management Information Base	Superseded
802.16-2009	Rollup of all standards up to date	Current
802.16j-2009	802.16-2009 amendment with multihop relay support	Current
P802.16m	Data rates of 100 Mbps for mobile, 1 Gbps for fixed	In Progress

for separate works on the 2-11 GHz and 10-66 GHz frequency intervals. In 2001, the first version of the 802.16 standard was released, supporting only line-of-sight (LOS) fixed-access communication, in a point-to-multipoint scheme. The use of frequencies within the licensed spectrum and lack of conformance with European standards for high-performance WMAN networks (ETSI HiperMAN) were some of the issues present in this version of the standard.

[1]Telemetering: a system that allows remote measurement and reporting of information; information types can range from simple water, gas, and electricity consumption values, to more advanced uses such as medical vital sign monitoring and rocket control.

The 802.16 working group subsequently produced revision 802.16a, a version which included non-line-of-sight (NLOS) support in the 2-11 GHz band, and Multiple-Access (MA) on the MAC layer with OFDMA. Further merging and improvements resulted in 802.16-2004, which replaced all of the previous versions and became the basis for WiMAX solutions. In 2005, IEEE 802.16e-2005 was completed and approved, which included the much-sought support for terminal mobility, new quality-of-service classes, and a new multiplexed mechanism, Scalable OFDMA (SOFDMA). More recently, the 802.16j-2009 amendment rolled up all previous versions of the standard, and includes support for a Mobile Management Information Base (Mobile MIB) and multihop relaying. Some of the technical characteristics associated with each revision of the standard can be found in Figure 2.2 [3].

Table 2.2: Basic data on IEEE 802.16 standards

	802.16	**802.16-2004**	**802.16e-2005**
Completion date	December 2001	June 2004	December 2005
Frequency band	10-66 GHz	2-11 GHz	2-11 GHz for fixed 2-6 GHz for mobile
Application	Fixed LOS	Fixed NLOS	Fixed and mobile NLOS
MAC architecture	PMP	PMP, MESH	PMP, MESH
Data rate	32-135 Mbps	1-75 Mbps	1-75 Mbps
Multiplexing	Burst TDM/TDMA	Burst TDM/TDMA/OFDMA	Burst TDM/TDMA / OFDMA/SOFDMA
Duplexing	TDD and FDD	TDD and FDD	TDD and FDD
Channel bandwidths	20 MHz, 25 MHz, 28 MHz	1.75 MHz, 3.5 MHz, 7 MHz, 14 MHz, 1.25 MHz, 5 MHz, 10 MHz, 15 MHz, 8.75 MHz	1.75 MHz, 3.5 MHz, 7 MHz, 14 MHz, 1.25 MHz, 5 MHz, 10 MHz, 15 MHz, 8.75 MHz

2.1.1 Technical specifications of the IEEE 802.16 standard

The main features of the IEEE 802.16 standard are as follows [1]:

- Carrier frequencies inferior to 11 GHz, although support is considered from 2 to 66 GHz. Currently available equipments operate on the frequency bands at 2.5 GHz, 3.5 GHz, and 5.7 GHz.
- Orthogonal Frequency Division Multiplexing (OFDM), a frequency-division multiplexing (FDM) scheme utilized as a digital multi-carrier modulation method; OFDM is particularly known for its high radio resource use efficiency and its ability to cope with severe channel conditions.
- Data rates of up to 70 Mbps with a low-noise radio channel and ideal conditions. A number of studies have shown, however, that a more reasonable figure is of 10-20 Mbps.
- Large distances of access, up to 20 km, thanks to the high robustness of OFDM and the support of a variety of modulation techniques.

The standard supports a number of digital modulation schemes for the transmission of data. The four main modulations supported by 802.16 are BPSK, QPSK, 16QAM and 64QAM. Each modulation is capable of an increasingly higher bit rate at the expense of being less robust and more susceptible to signal errors and path loss. Figure 2.2 illustrates the modulations selected by 802.16 depending on the signal-to-noise ratio (SNR) (it is assumed, for the sake of simplicity, that a greater distance results in a lower SNR).

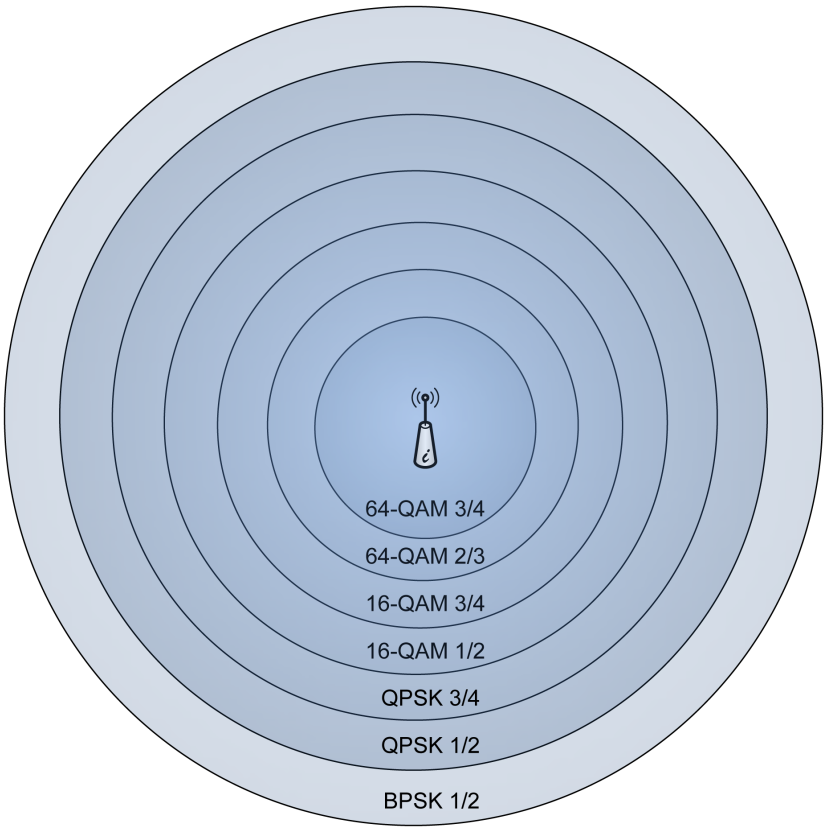


Figure 2.2: Diagram of 802.16's selection of carrier modulations

The standard defines two modes of operation: *Point-to-multipoint* (PMP) mode, and *Mesh* (MESH) mode. The 802.16's MAC layer is primarily designed for PMP network topologies, where all the subscriber stations (SS) are within range of the base station (BS), and all communication is directed to that BS. The MESH mode is an extension to the PMP mode that allows for traffic to be routed through other nodes, and for transmissions to occur directly between nodes without routing through a BS.

2.1.2 Mesh Networks and the IEEE 802.16 MESH mode

Wireless mesh networking is a promising architecture for the next generation of wireless networks. Wireless mesh networks' inherent capacity for self-configuration, self-organization, and self-healing grants them a number of desirable advantages when compared with traditional point-to-multipoint networks.

Unlike cellular networks, where the failure of a single base station directly causes a service

unavailability over a large geographical area, wireless mesh networks (WMNs) are highly fault-tolerant, even in the event that a number of nodes go down. WMNs can support an infinite number of network topologies – Figure 2.3 demonstrates one such topology for providing BWA to a number of locations. In this example, the backhaul of the network is provided

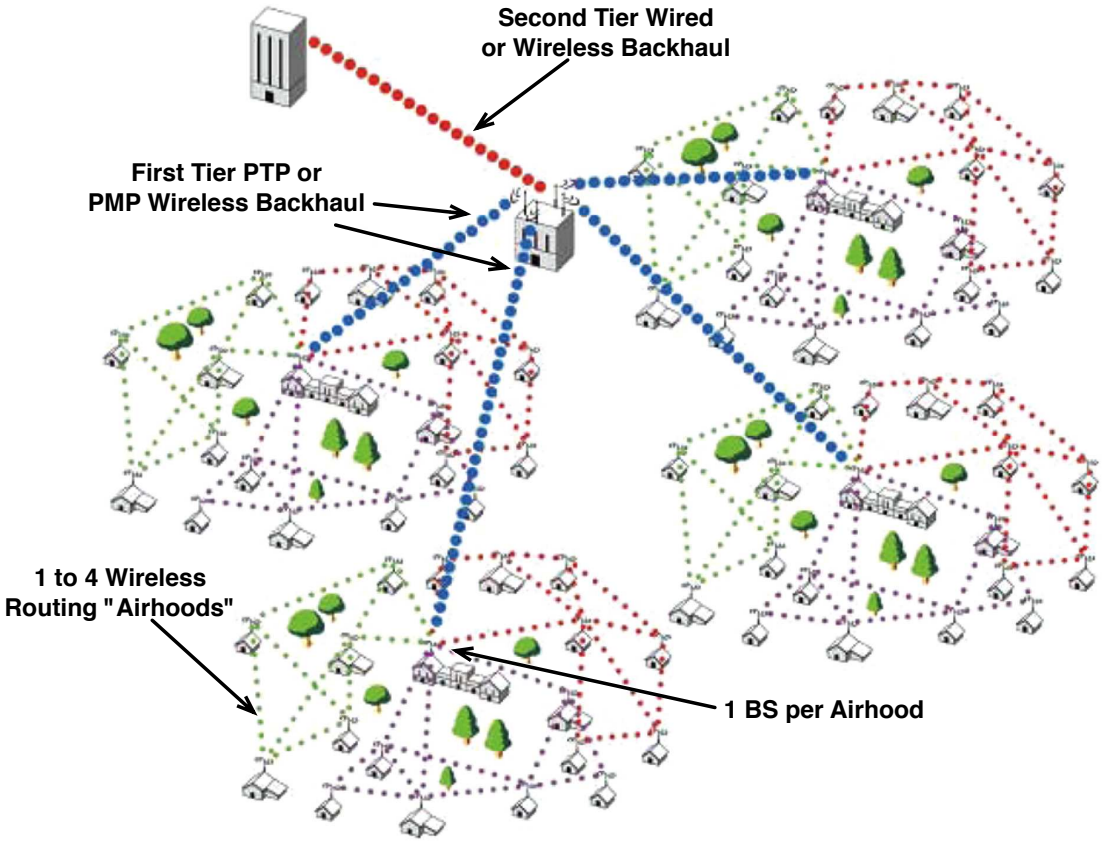


Figure 2.3: Example topology of a mesh-based BWA network

over a fixed PMP wireless network to four central locations, from where the service is then distributed across a number of destinations with a WMN. The mesh topology allows for easy routing around obstacles such as trees, buildings and landscape, covering a large area with efficiency and redundancy.

One issue of providing broadband wireless access is the need to cover large areas as efficiently as possible, both to provide reliable service to the end user and lower deployment costs. Figure 2.4 relates the probability between the existence of a reliable link between any two points in the network and the chance that a new node is under the area of service of the network, for a PMP scenario, and two MESH scenarios: BS ('AirHead') plus 10 nodes, and BS plus 50 nodes. As it can be seen, the coverage and robustness of the mesh network improves exponentially with the addition of new nodes to the network, in stark contrast to the linear increase seen in the PMP mode [4].

Throughput is also one of the areas where BWA mesh networks have the potential to outperform traditional PMP networks. Consider the scenario presented in Figure 2.5, where

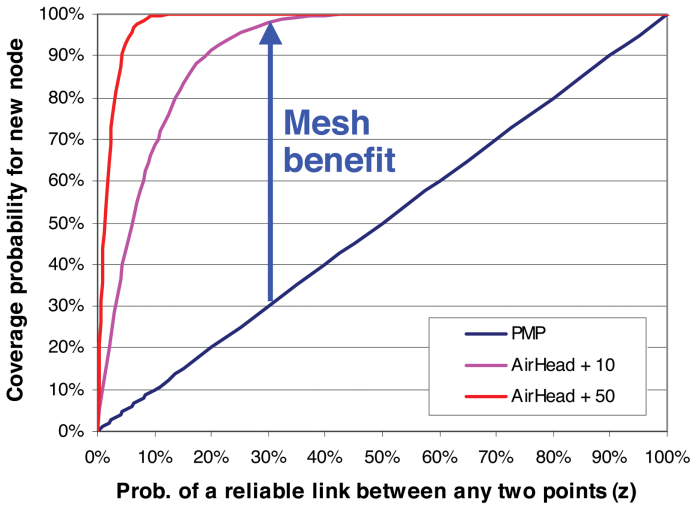


Figure 2.4: Coverage versus probability of a reliable link in PMP and MESH

the mesh BS on the left is trying to communicate with the subscriber station on the right. In a PMP mode, the communication would be made via direct path, while in the MESH mode, a two-hop approach with the intermediate SS would be used. Here, path loss works in favor of the two-hop approach: signal path loss, by definition, increases in a squared relation with the distance between sender and receiver (r^2) [5]. Thus, if channel conditions dictate, e.g., that the direct link between the mesh BS and the last SS supports a QPSK$\frac{1}{2}$ profile (16 Mbps throughput), then the shorter links will each have 9 dB less path loss (due to the r^2 relation), and should be capable of 16QAM$\frac{3}{4}$ (48 Mbps throughput). This means that, operating with a single radio, the two-hop scenario will obtain 24 Mbps of effective throughput, compared to only 16 Mbps for the direct link. With multiple radios, the mesh network can maximize its two-hop communication potential and be capable of the full 48 Mbps that each link provides, a 3x increase from the PMP scenario.

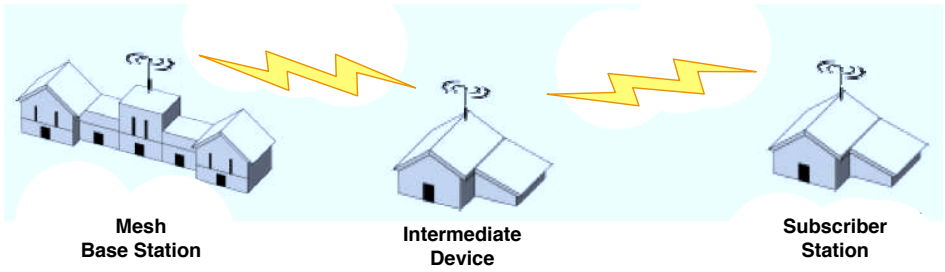
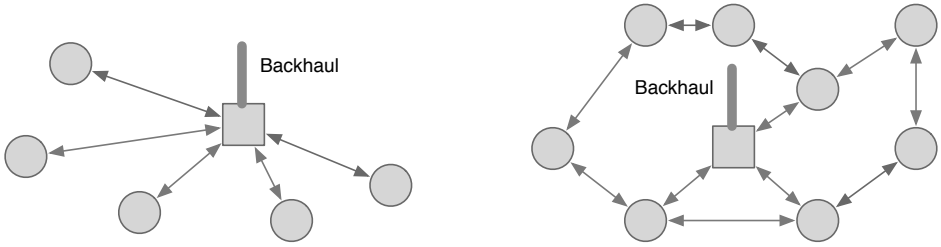


Figure 2.5: Example scenario of a two-hop transmission

The IEEE 802.16 working group standardized, with amendment 802.16a-02/30r1, enhancements to the MAC layer protocols in order to provide support for mesh networks. Version 802.16a of the standard, approved in January of 2003, includes support for Non-Line-of-Sight transmissions, operating in both licensed and unlicensed frequency spectrums ranging from 2 to 11 GHz, supporting data rates of up to 75 Mbps and ranges of up to 20 km [6].

Physical operation in the MESH mode



(a) PMP mode (b) MESH mode

Figure 2.6: Sample topologies for PMP and MESH networks

Figure 2.6 compares the typical topologies of PMP and MESH networks. In a network operating under the PMP mode, a central Base Station (BS) has the main role, and is responsible for coordinating and handling all traffic from the other nodes in the network, called Subscriber Stations (SS). The SSs need to communicate directly with the BS, and no direct communication between SSs can occur. In a mesh network, each SS can form links with neighbor SSs and transmit data directly. This brings numerous advantages: e.g., there is no central point of failure in the network, unlike in a PMP network where disconnecting the BS effectively shuts down the whole network. Another advantage of a mesh network is its capacity to easily cover wide areas, since each node is, in itself, a base station, and thus routing around obstacles and extending access to remote areas is facilitated.

MAC layer: frame structure

IEEE 802.16's MESH mode defines mechanisms for SSs to communicate between one another in the MAC layer. We will now look into the frame structure of the MESH mode, as described in the standard, and the mechanisms for QoS it enables.

The MESH mode in IEEE 802.16 can only operate in Time-Division Duplexing. In this mode, the time axis is divided into frames of a configurable length, and each frame is divided into two subframes: the *control subframe* and the *data subframe*. Transmissions in the control subframe are always made using QPSK-1/2 modulation, while modulation in the data subframe can be negotiated to get the best possible data rate. A control subframe is composed of a series of transmission opportunities, so as to enable multiple nodes to send control messages during that subframe. Likewise, a data subframe is divided into a number of minislots, for the same purpose of permitting shared access by multiple nodes.

The control subframe itself has two types, to cater for different needs:

- Network Control Subframe
- Schedule Control Subframe

Figure 2.7 shows the frame format of the IEEE 802.16 MESH mode, with emphasis on the schedule control subframe. The network control subframe is used to transmit management messages for such functions as joining a new network, synchronizing with an existing network, and establishing links with neighbors. A distinguishing feature of the IEEE 802.16 standard

is the support for Quality-of-Service directly at the MAC layer: the schedule control subframe is used by nodes to transmit MAC management messages for setting up data transmission schedules. For this purpose, the standard defines a set of control messages to enable nodes to set up transmission schedules and synchronize their transmissions in a way that is collision-free. These messages are sent by nodes in the schedule control subframe opportunities.

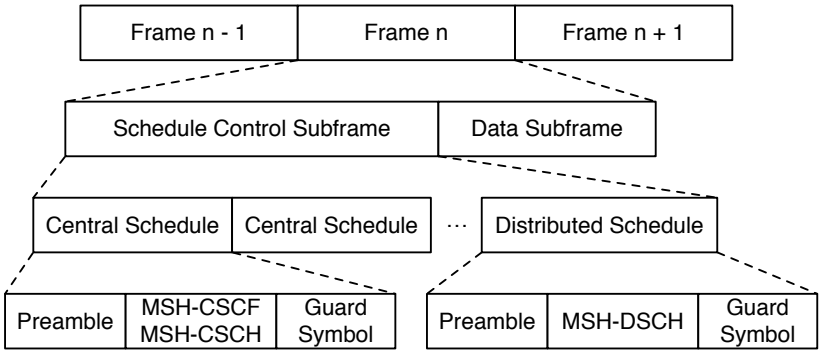


Figure 2.7: Schedule control subframe structure

MAC layer: scheduling mechanisms

The MESH mode defines two types of scheduling: coordinated centralized scheduling, coordinated distributed scheduling, and uncoordinated distributed scheduling.

Centralized Scheduling

In centralized scheduling, a central node in the mesh network (mesh BS) is responsible for the data subframe scheduling coordination, resource allocation and resource granting. In this scheduling mechanism, the network assumes a tree-like structure with the mesh BS at the root of the tree. Each SS determines its traffic requirements and then sends a Mesh Centralized Schedule (MSH-CSCH) Request message to its parent node. Each node aggregates the traffic needs from its children and forwards them to their parent node, until the mesh BS is reached. Once the mesh BS receives a Mesh Distributed Schedule (MSH-DSCH) message, it extracts the traffic requests in it, determines the amount of granted resources per node for each link, and broadcasts an MSH-CSCH Grant message to all its nodes, who then forward it to their own neighbors.

Centralized scheduling simplifies the process of making scheduling decisions, since the mesh BS has knowledge of all the traffic demands in the network. It introduces a performance penalty however, due to the time required to exchange messages back and forth from the mesh BS to the mesh SSs, and due to the overhead bandwidth implied, which can become problematic for large and / or dense networks. It also introduces a single point of failure in the network: if the mesh BS goes down, nodes in the network will be unable to request and be granted bandwidth to communicate.

Distributed Scheduling

Distributed scheduling is used when a node needs to reserve bandwidth for transmission to another node in their two-hop neighborhood. Two methods of distributed scheduling exist: coordinated and uncoordinated distributed scheduling.

When using coordinated distributed scheduling, a distributed algorithm is used by the nodes to compete for the transmission opportunities in the schedule control subframe. This assures that every node in the local neighborhood can listen to the schedule control subframe and be aware of the reservations scheduled by its neighbors. Once a node wins a transmission opportunity in the control subframe, a three-way handshake process to schedule minislot reservations is employed, as shown in Figure 2.8. The algorithm responsible for allocating minislots has not been defined in the standard, and is open to implementation.

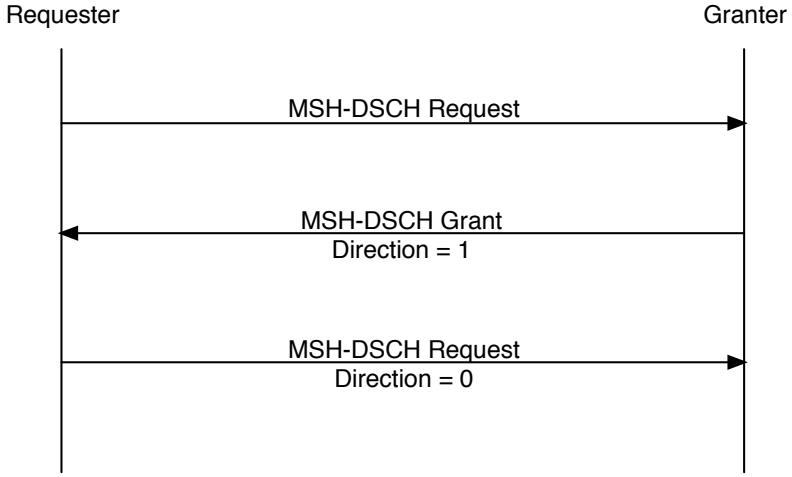


Figure 2.8: Three-way handshake reservation mechanism

In uncoordinated distributed scheduling, however, the nodes send directed requests and grants in the data subframe, thus bypassing the competition for transmission opportunities. This has the advantage of decreasing the time it takes to request new bandwidth, but it can also happen that the control messages sent in this fashion are not received by all the other neighbors (due to other simultaneous transmissions), which may then cause those nodes to schedule conflicting transmissions. Due to this issue, uncoordinated distributed scheduling should be avoided for long-term reservations.

Distributed scheduling makes use of Mesh Distributed Schedule (MSH-DSCH) messages to transmit information. An MSH-DSCH message can carry one or more of the following Information Elements (IE):

Availability IE - to advertise minislot reservation statuses

Scheduling IE - to coordinate the time for the next MSH-DSCH transmission slot

Request IE - to request a number of minislots from a neighbor

Grant IE - to grant minislots previously requested from a Request IE

When a node wishes to request minislots, it sends an MSH-DSCH message with a RequestIE, targeted to the destination of the request. Along with this RequestIE, the requesting node must also send a number of AvailabilityIEs, indicating which minislots it has available for transmission. The destination node then receives this MSH-DSCH, and matches the AvailabilityIEs in it against its own list of minislots available for reception. Then, if a suitable number of minislots is found, the node sends back to the requester node an MSH-DSCH with a GrantIE with $Direction = 1$ indicating which minislots it is granting. Finally, the requester node receives this message and acknowledges the reservation with an MSH-DSCH containing a GrantIE with $Direction = 0$ (known as a Grant Confirmation).

For a more in-depth look into the details of the MESH mode frame structure and bandwidth reservation process, a technical report on the workings of the MESH mode of operation is provided in [7], and is an important anchor point for researching this topic; also of relevance is [8], on the performance of the election procedure in the MESH mode. Further information can also be found in the IEEE 802.16 standard document [9].

2.1.3 QoS architectures for the IEEE 802.16 MESH mode

Quality-of-Service support in 802.16's MESH mode can be split in two topics: the scheduling mechanisms defined in the standard for bandwidth reservations and management messages, and the QoS architectures designed to make use of those mechanisms to support service classes and QoS requirements. This work will focus solely on the research of a QoS architecture, adapting the scheduling mechanisms only when required.

On the topic of scheduling in the MESH mode, there is sizable amount of research on the evaluation and performance improvement of those scheduling mechanisms [10, 11, 12], for both single-radio and multi-radio setups, and for centralized [13, 14] and distributed [15, 16] scheduling schemes.

Regarding QoS architectures, C. Cicconetti *et al* propose, in [17], an End-to-end Bandwidth Reservation Protocol for the MAC layer of 802.16's MESH mode, inspired in RSVP, to support traffic flows with strict QoS requirements. The suggested framework was shown capable of establishing bandwidth reservations with a low latency, and included support for distributed Call Admission Control. Algorithms for efficient and fair centralized scheduling, with means to satisfy aggregate mean traffic requirements and to provide QoS to both real-time and non-real-time traffic flows are provided in [18].

None of these architectures support, however, a set of service classes compatible with the ones already defined in the standard for the PMP mode. A novel QoS architecture for managing bandwidth in the MESH mode is presented in [19], with the aim of supporting the same scheduling services that exist for the PMP mode and thus facilitate interoperability between the two modes. This architecture brings the PMP service classes UGS, rtPS, nrtPS and BE to the MESH mode, and is the groundwork for the QoS research performed in this thesis. Chapter 3 will refer to this architecture in greater detail.

We propose to extend this QoS architecture to include mechanisms for a fair distribution of minislot reservations, which includes methods to cancel active reservations, and also to implement it in an 802.16 MESH mode simulator, dealing with the issues that arise from an actual implementation of the architecture. This includes the definition of service class

behavior, of packet matching and mapping, and of some mechanisms that are undefined such as the process with which minislots for rtPS unscheduled messages are reserved.

2.2 Subjective quality evaluation of user services

In the last years, considerable efforts have been made to predict end user perception of provided services. In particular, services that provide content such as video and audio should be evaluated in terms of the end user experience: the perceived Quality-of-Service, or Quality-of-Experience. This stems from the fact that, to the end user, mathematical measures such as throughput and packet loss are meaningless unless they represent clear modifications to the experienced service.

Audio and video services have a higher relevance in this topic due to the fact that they introduce new requirements and expectations. In terms of audio, speech perceptiveness and the ability to maintain an interactive conversation between two parties is much more relevant to a VoIP call than the typical measure of bit rate used in audio. This ability is mostly impacted by factors such as network delay, packet error rate, and also the burstiness of these errors. In terms of video, the end user is mostly concerned with blurriness, skipping, blocky images and video 'hiccups'. Although some of these can be measured directly from the encoding process at the source, many of these issues can be originated by network problems during the video transmission and, again, a simple measure of throughput and delay is insufficient to clearly identify what the end user is experiencing.

A need for subjectively measuring service quality thus arose, to overcome all of these issues and to try and provide the best user experience when providing this kind of services. Measurements based on panels of human users, albeit expensive and time consuming, provided statistically relevant information and laid the groundwork for further research. Lately, a greater focus has been granted to this topic, and a number of methods to predict the end user perception of quality in real-time and with as much accuracy as possible have been researched.

We now introduce a popular metric to judge subjective quality, the Mean Opinion Score, and discuss the techniques available to predict the perceived quality of audio and video streams.

2.2.1 The Mean Opinion Score metric

One popular metric to capture user satisfaction is the Mean Opinion Score (MOS). MOS was originally proposed for subjectively estimating voice quality, and it provides a numerical measure of the quality of human speech at the destination. To obtain a MOS score, subjective listening tests are performed. A number of subjects listen to a test sentence transmitted over a given communication channel, and then rate the quality as one of the following:

1 - Bad

2 - Poor

3 - Fair

4 - Good

5 - Excellent

The combined ratings by all of the subjects are then mathematically averaged to obtain a quantitative indicator of the system performance. The MOS is thus the arithmetic mean of all the individual scores.

This method of conducting subjective tests is clearly not feasible for real-time monitoring of audio quality. To overcome this issue, and making use of data obtained from tests with panels of human listeners, the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) provided a mathematical model capable of predicting, with a high correlation, the MOS scores that would result from a subjective test. This test methodology is standardized as ITU-T recommendation P.862, *Perceptual Evaluation of Speech Quality* (PESQ) [20]. PESQ itself, however, is computationally expensive and works by comparing the original audio sample with the one at the receiver's end, something which is impractical since it requires one of the end points to have both audio samples for applying the model.

A number of alternative mathematical models were then created with the goal of estimating this subjective quality from limited sets of data. One of these models, also from ITU-T, is the E-model [21], which is capable of predicting MOS from common, objective measures such as packet delays and packet loss. We adopt this model to obtain audio quality scores for our scheduler, and it is thus described in greater detail in Section 4.1.2.

2.2.2 Video quality measurement and estimation

Due to the increased popularity of multimedia services in the Internet, video data is expected to account for a large portion of traffic in the future next-generation networks, be they wired or wireless. To transport video over a network, a video file is normally encoded to reduce its bandwidth requirements. Encoding incurs a quality penalty however, and even then the bandwidth required to transmit a video file is normally in the order of hundreds of *kbps*, or even some *Mbps* for high-quality, high-definition video whose popularity is also growing considerably.

Due to the inherent complexity associated with digital video, the topic of estimating video quality is very broad. There is a vast number of codecs, each with numerous parameters and variations, frame structures, decoding techniques and error recovery techniques. In this section we summarize the most important concepts regarding video processes and video quality estimation.

Principles of non-scalable video encoding

The most common encoding methods for network video nowadays are based on non-scalable video encoding. This means that, to reduce the bandwidth of the video, a transcoding step is required at the video source. Scalable video encoding is a relatively new encoding method which facilitates bandwidth reduction of videos by simply dropping frames, thus skipping the need for a transcoding part. We will now focus on the principles of non-scalable video encoding.

Digital video consists of a number of video frames that are displayed at a fixed frame rate. The frame format specifies the size of each individual frame in terms of pixels: the

ITU-R/CCIR-601 format (the common TV format) has 720x480 pixels (i.e., 720 pixels in the horizontal direction and 480 pixels in the vertical direction); the Common Intermediate Format (CIF) format has 352x288 pixels, and the Quarter CIF (QCIF) format has 176x144 pixels [22]. CIF and QCIF are common frame formats for network-related simulations and codec testing.

MPEG encoding introduced three types of frames, which are also present in the H.26x family of codecs: intra-coded (I-frames), inter-coded (P-frames), and bidirectional coded (B-frames) [22]. These frame types are then organized into groups which are denominated of GoPs (groups of pictures). A typical GoP pattern can be, for example, IBBPBBPBBPBBIBBP..., where the GoP is composed of an I-frame and all the P- and B- frames occurring until the next I-frame. One such GoP pattern, with three P-frames in a GoP and two B-frames between each I/P-frame is illustrated in Figure 2.9. This image also shows the reference-based coding

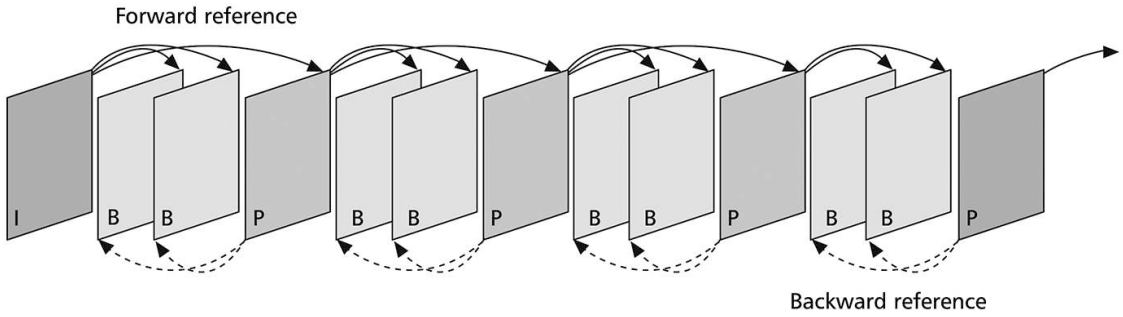


Figure 2.9: Typical MPEG GoP pattern with references for predictive coding

methods of MPEG: a P-frame is normally coded with reference to the preceding I- or P-frame, i.e., the I- or P-frame serves as a forward reference. A B-frame is normally coded with reference to both the preceding and the succeeding I/P-frames. This, however, means that a P-frame cannot be decoded if the previous I/P-frame is not present (or was not received), and that likewise a B-frame cannot be decoded if any one of the preceding and succeeding I/P-frames are not present as well.

Network performance evaluation with video streams

Network simulations of video traffic normally make use of one of three different methods to simulate a video stream. The *video bit stream*, generated from the video encoding applications, contains the complete video structure and bit information. With this method, the video quality can be evaluated with either objective or subjective methods, since it is possible to playback the source and the resulting stream. A video bit stream is however very large and cumbersome, ranging from several gigabytes for a compressed video to many tens of gigabytes for uncompressed video.

An alternative to these video bit streams are *video traces*. A video trace retains only the size, ordering, and decoding time of each frame. This can be used to simulate traffic with the exact same pattern of a true video stream. Side information may also be included in a video trace, such as the distortion information associated with each video frame. A video trace also

has the advantage of not being subject to copyright issues, which can happen with video bit streams.

The third method is called a video traffic model, and is in its essence a mathematical model based on statistical analysis of video stream characteristics which is then capable of indefinitely simulating traffic streams with patterns which resembles the streaming of a video. The quality of these traffic models is measured by its correlation to actual video stream characteristics, and they can be parameter-tuned to easily provide many kinds of encoded video patterns.

For the simulations in this thesis, we make use of video streams to simulate video traffic. Further information on the structure of video traces and how to process and packetize video frames from these traces can be found in [22, 23, 24].

Distortion prediction methods for video streams

There is a fair amount of research done on methods to estimate the quality of video sequences. These methods result in output metrics such as Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and PSNRplus, which measure the similarity between a video stream before and after transmission through a network channel, or before and after encoding.

PSNR is the ratio between the maximum power of a signal and the power of the noise which is affecting that signal. In image compression, the signal is the original data, and the noise is the error introduced by compression. When comparing compression codecs, PSNR is used as an approximation to human perception of reconstruction quality – a higher PSNR would normally indicate that the reconstruction is of higher quality.

SSIM [26] is a framework for accessing quality based on the degradation of structural information. SSIM is shown to compare favorably against typical methods such as PSNR, but is computationally more expensive and may not be suitable for real-time evaluation of video degradation.

PSNRplus [25] is a method to increase the correlation between subjective and predicted video quality by estimating the parameters of the linear regression line for each video sequence. Although this method produces improved results compared to the other three methods listed, it requires every sequence to be coded or compressed 3 times in order to obtain 2 additional instances, hence making this technique unsuitable for real time applications.

In our simulations we make use of a model presented in [27] for evaluating the impact of packet losses in video quality while accounting for application-specific parameters such as the video codec, loss recovery techniques, bit rate, packetization and content characteristics. The model is generally applicable to any motion-compensated video scheme, and specifically designed around real-time requirements.

2.3 The *ns2mesh* IEEE 802.16 MESH mode simulator

Our tool of choice for evaluating the performance of the architectures researched in this thesis is an implementation of the IEEE 802.16 MESH mode for the *ns2* simulator, *ns2mesh80216* [28, 29]. The *ns2* simulator alone does not support 802.16's MESH mode,

but researchers from the University of Pisa and Georgia Institute of Technology developed a patch (*ns2mesh80216*) that allows mesh networks to be simulated. This extension to *ns2* has no support for the PMP mode, only the MESH mode.

The *ns2mesh80216* network simulator implements access to the data sub-frame by means of the three-way handshake specified in the IEEE 802.16 standard [9], plus a Fair End-to-end Bandwidth Access scheduler [30] for handling traffic priorities. No support for uncoordinated distributed scheduling is included, and the scheduler has no support for QoS service class differentiation. One caveat of this patch is that it provides no support for *ns2*'s physical models (i.e., it has no way of specifying the position of the nodes and negotiating data modulations according to a given level of interference). This means that all links between nodes are purely logical, and modulations have to be specified by hand or fed by some other physical simulation software.

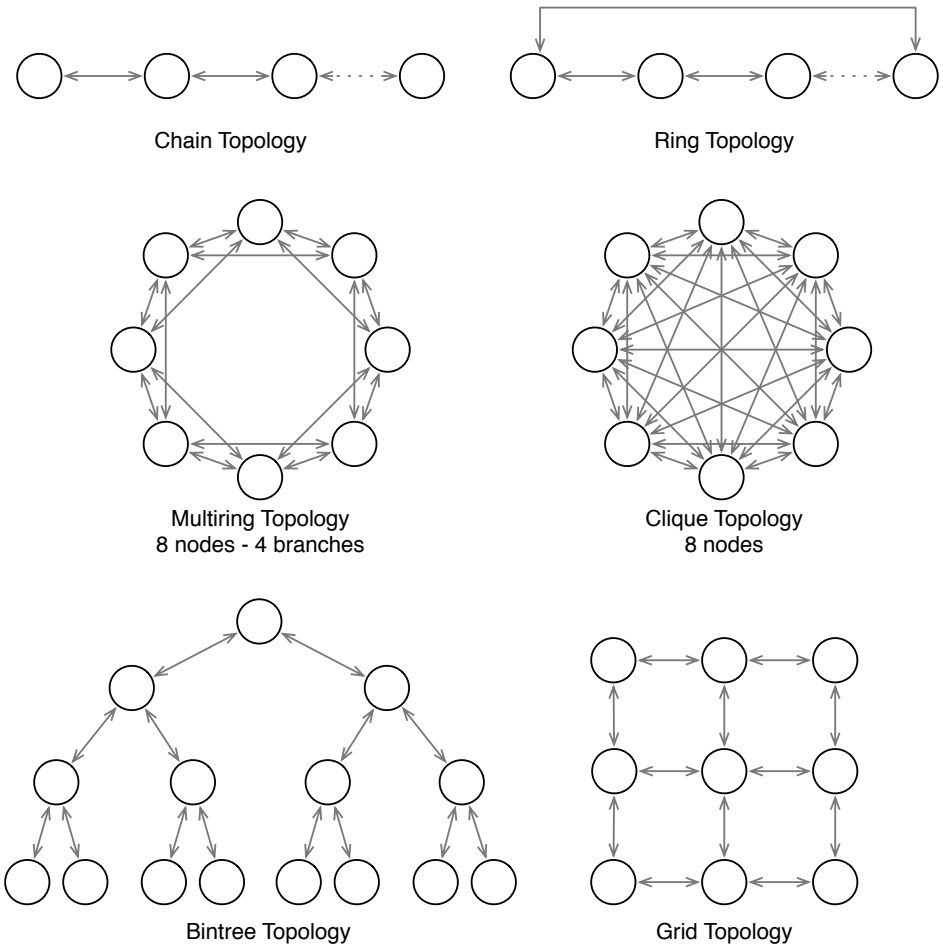


Figure 2.10: Mesh topologies supported by the *ns2mesh80216* simulator

This simulation environment includes a TCL simulation file scripted for ease of configurability, with a number of topologies for mesh networks predefined. Examples of some of these topologies are shown in Figure 2.10. *ns2mesh80216* also bundles the *ns2measure* framework [31], which is a set of C++ modules designed by the same authors to enable flexible and

efficient data collection on large-scale simulations, complemented by a set of tools to perform statistical analysis on the collected measurements. We do extensive use of this framework to collect measurements of delay, throughput and packet loss in our simulations of QoS, and extend the base classes to support collection of subjective quality parameters for the QoE simulations.

2.4 Summary

In this chapter we provided insights and technical information on the IEEE 802.16 standard, its MESH mode of operation and the scheduling mechanisms it supports. We then reviewed some of the research made on these scheduling mechanisms, and listed the existing works which provide architectures for better QoS support in the standard.

We also introduced the concept of subjective evaluation of user experience. We described the Mean Opinion Score measurement, and went through the existing methods to estimate user experience for audio and video services, which included a brief introduction to the concepts surrounding digital video transmission.

Finally, we described the *ns2* network simulator and the *ns2mesh* extension for providing 802.16 MESH mode simulation support.

Chapter 3

Implementing QoS in IEEE 802.16 mesh networks

Good bandwidth management algorithms are crucial to ensure a robust and efficient behavior of IEEE 802.16 mesh networks. Over the past few years, most research in QoS for 802.16 has been directed at this standard's PMP mode of operation, while architectures to support QoS in the MESH mode have been mostly non-existent. As discussed previously in Chapter 2, the MAC layer in IEEE 802.16's PMP mode fully defines mechanisms for QoS. It provides different service classes to meet the demands of all kinds of services, from real-time to high-throughput. Our research will focus on bringing the same level of QoS support to the MESH mode of operation of 802.16 networks.

In this chapter, we will look into a novel architecture for managing bandwidth in the 802.16 MESH mode of operation which aims to implement scheduling classes similar to those currently supported by the PMP mode. We begin with a thorough description of our QoS architecture for the MESH mode, in Section 3.1. Then, we describe how the architecture was implemented in the *ns2mesh* simulator, and what improvements were made to that implementation, in Section 3.2. In Section 3.3, we evaluate the performance of the architecture by performing a number of simulations with various scenarios, and then finally we summarize our research in Section 3.4.

3.1 A QoS architecture for 802.16 mesh networks

In this section we describe our architecture of choice to implement QoS in the IEEE 802.16 MESH mode, which can be found in [19]. The most relevant characteristic of this architecture is that it aims to implement the same QoS service classes defined in the 802.16 standard for the PMP mode, and thus facilitate interoperation between networks operating under the two modes, MESH and PMP. The architecture brings the UGS, rtPS, nrtPS and BE service classes to the MESH mode, and mechanisms for their correct functioning in multihop networks.

3.1.1 QoS support in IEEE 802.16's PMP mode

Since the purpose of this QoS architecture is to provide seamless coexistence of PMP and MESH network, we will first begin by describing the QoS classes already defined for

802.16's PMP mode. QoS under the PMP mode is provisioned on a per-connection basis. All transmissions must occur within the context of a connection, which means that a bandwidth reservation process must occur previous to any data being sent. The MESH mode must operate on the same fashion, and require explicit bandwidth reservations.

The point-to-multipoint mode of operation defines the following data scheduling services: Unsolicited Grant Service (UGS), real-time Polling Service (rtPS), non-real-time Polling Service (nrtPS) and Best Effort (BE). Their purposes are:

UGS - meant to support real-time streams of data consisting of periodically-issued fixed-size packets (i.e., traffic with a constant bit-rate pattern).

rtPS - designed to support variable-size packet streams with low delay requirements.

nrtPS - meant for high throughput data streams with non-strict delay requirements.

BE - traffic in this service class is serviced on a slot-available basis.

The next section will describe in greater detail the methods for implementing these four service classes in the MESH mode.

3.1.2 Architecture

This architecture's main components belong to the scope of the MAC Common Part Sublayer (CPS), as defined in the standard. It comprises four modules:

- Data Management Module
- MAC Management Module
- Bandwidth Management Module
- Packet Classifier

Figure 3.1 illustrates the architecture's modules and functioning. We will now describe these four modules and their role in the architecture.

The *Packet Classifier* module exists in the Service-specific Convergence Sublayer (CS), and is responsible for mapping outgoing packets coming from the Network Layer and assigning mesh CIDs to them, which are later used to assign each packet to a service class. This mapping is made between the IP Type of Service (TOS) field[1] and the fields Priority, Drop Precedence and Reliability of the mesh CID. Table 3.1 shows the mappings chosen for this implementation. Packets with the TOS fields marked with the Throughput flag only (values 2 and 3) are assigned to nrtPS; the ones marked with the Delay flag only (4 and 5) are assigned to rtPS; those marked with both Throughput and Delay flags (6 and 7) are placed in the UGS class; and finally, packets with no Throughput nor Delay flags (0 and 1) are given to BE.

The *Data Management Module* is responsible for enqueueing packets arriving from the Convergence Sublayer into their respective service class queues, according to the mesh CID

[1]For the purpose of this work we assume the Network Layer protocol to be the Internet Protocol. Other mappings similar to the one described can be implemented to support any Network Layer protocol also supported by the 802.16 standard itself.

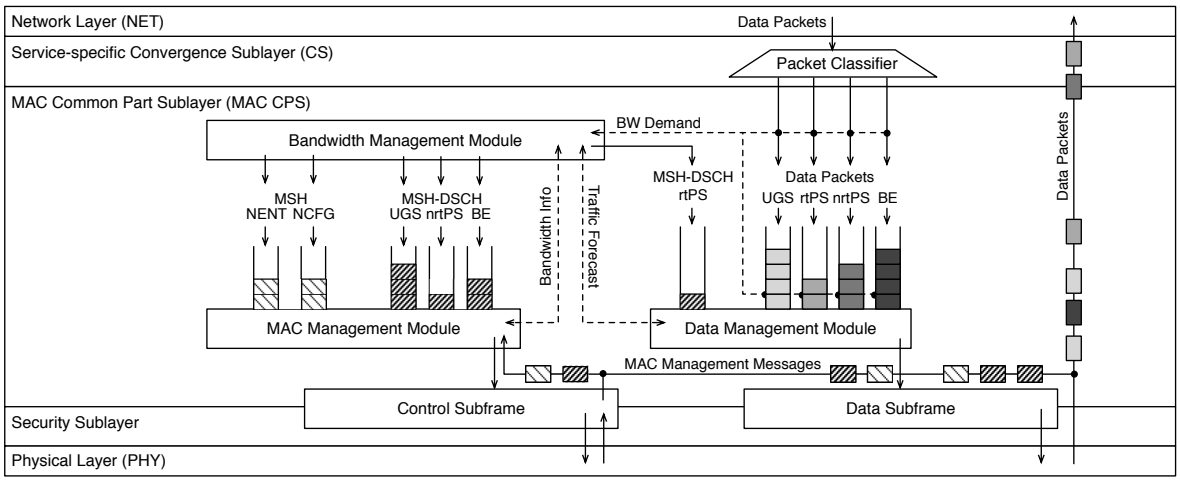


Figure 3.1: Diagram of the QoS architecture

Table 3.1: Mappings from IP TOS to Mesh CID fields

IP TOS	802.16 Service Class	MSH CID Priority	MSH CID Drop Precedence	MSH CID Reliability
0	BE	0	3	0
1		1	3	1
2	nrtPS	2	2	0
3		3	2	1
4	rtPS	4	1	0
5		5	1	1
6	UGS	6	0	0
7		7	0	1

fields set by the *Packet Classifier* module. Since this module is directly responsible for accessing the data subframe, it also handles Mesh Distributed Schedule (MSH-DSCH) messages from uncoordinated distributed scheduling, which are sent in this subframe. The *Data Management Module* keeps track of the minislot reservations who were granted by other nodes, and sends queued packets in the minislots reserved for transmission of data of a specific service class. Specific schedulers can be implemented in this module in order to try to meet strict QoS requirements.

The *Data Management Module* also performs a traffic forecast, by computing the amount of data that is arriving from the CS layer in intervals of 32 frames, and then informing the *Bandwidth Management Module* of the new bandwidth requirements, so that it can request more bandwidth or cancel existing grants. In our architecture we opt to inform the *Bandwidth Management Module* only when the traffic profile has changed by more than 20% or reduced to zero, as to avoid overloading the module and the network with bandwidth requests.

The *MAC Management Module* has the role of handling and processing the MAC management messages, both of network control and of scheduling control. We will focus on the handling of scheduling control messages only. This module receives messages from other nodes, such as bandwidth requests, grants or grant-confirmations, and updates the respective

internal tables who keep track of minislot needs and grants. If necessary, the information contained in these messages is forwarded to the *Bandwidth Management Module*, which is responsible for handling the bandwidth needs of the node. The *MAC Management Module* is also responsible for the election algorithm, which is required for the node to compete for transmission opportunities in the control subframe. The schedule control messages in our implementation include information regarding the service class of the minislot request, and thus the module can grant bandwidth based on the expected traffic pattern of that class. The structure of these MSH-DSCH schedule control messages in our implementation is shown in Figure 3.2.

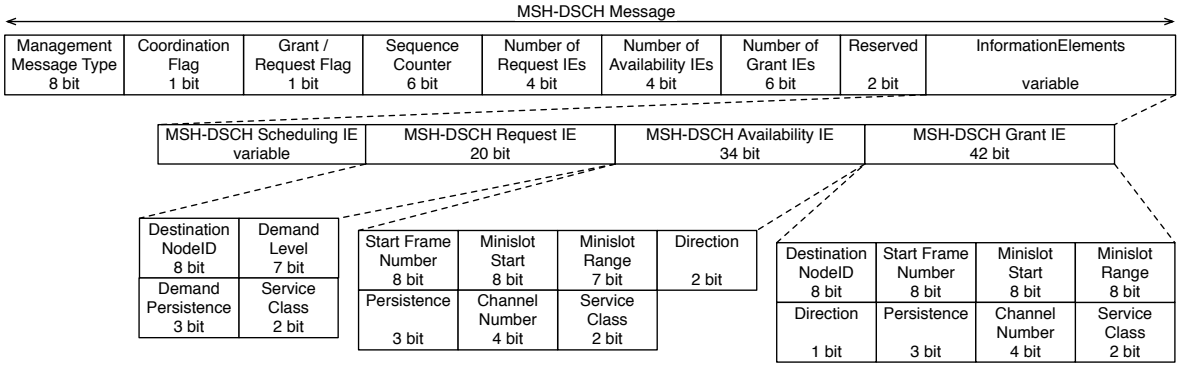


Figure 3.2: Structure of an MSH-DSCH message and the Information Elements within it

The last module of this architecture is the *Bandwidth Management Module*. This module is responsible for receiving traffic forecast from the *Data Management Module* and generating bandwidth requests when more bandwidth is required. Likewise, it must generate bandwidth cancel messages when a given service class is no longer receiving the same amount of data as before. This module must also process bandwidth requests from neighboring nodes, by inspecting the internal minislot allocation tables and distributing them accordingly.

We now briefly describe the workings of each service class.

Unsolicited Grant Service

The UGS service class is ideal for constant bit rate traffic. Since the bandwidth requirements of this kind of traffic are supposed to remain unaltered for long periods of time, the overhead in bandwidth requests can be considerably diminished by requesting minislots for an undefined amount of frames. In our implementation, the transmitting node performs an estimation of the required bandwidth of all UGS flows, and then sends a unique MSH-DSCH Request IE with a Demand Persistence of 7, which represents a demand for an unspecified number of frames (i.e., infinite), and thus that must be explicitly cancelled when the bandwidth is no longer required. The UGS service class has the highest priority of all the four classes when distributing bandwidth.

Real Time Polling Service

The rtPS service class is designed for variable bit rate traffic which requires very low delay. Most interactive and real-time multimedia applications have a variable bit-rate (VBR)

traffic pattern, and are best handled in the rtPS class. Due to randomness of bandwidth requirements of such an application, there is a need to periodically request and cancel minislot reservations. To be able to do that while keeping a low overhead, we must bypass the competition for the control channel transmission opportunities. The rtPS service class makes use of uncoordinated distributed scheduling to request bandwidth, thus achieving this purpose. By sending bandwidth requests in the data subframe, it can effectively decrease the time required to request bandwidth, especially for high density networks where transmission opportunities are scarce. In the event of minislot exhaustion due to high throughput demands, rtPS is allowed to borrow small amounts of bandwidth from other service classes (typically, 5 minislots) for its MSH-DSCH messages. No specific traffic pattern detection is performed by the traffic estimator, and all rtPS requests are made with a Demand Persistence of 5, spanning the request over 32 frames.

Non-Real Time Polling Service and Best Effort

For these two services, bandwidth requests are made with a persistence of 32 frames (Demand Persistence 5), and a Demand Level appropriate for the required bandwidth. The difference between the two lies in the priorities for scheduling their control subframe messages. The BE requests are only granted bandwidth once all other classes have been served.

For the nrtPS service, a minimum transfer rate can be assured by setting a minimum number of slots per frame to it. This prevents starvation when nrtPS needs to contend with rtPS and UGS for minislots.

3.1.3 Bandwidth balancing and minislot distribution

One of the shortcomings of the QoS implementation in *ns2mesh* is regarding multi-hop traffic flows. The issue stems from the fact that, when a node receives a bandwidth request, it has no idea on how much of the traffic coming from that reservation is directed to itself, and how much it will need to forward (and thus make a reservation) to neighboring nodes. This issue is aggravated when a single channel model is being used, since each mesh data subframe must then be allocated for both transmitting and receiving data (while in a multi-channel scenario multiple frequencies can be chosen to transmit and receive data, attenuating this issue to a certain degree). Add to the fact that each link between nodes will have its own burst profile, and thus that each minislot will carry a different amount of bits, and the complexity increases considerably.

Consider the following scenario: a three-node network topology has node A attempting to send data to node C via node B, which is in between A and B. A schematic can be found in Figure 3.3. Node A then proceeds to send a bandwidth reservation to B according to its

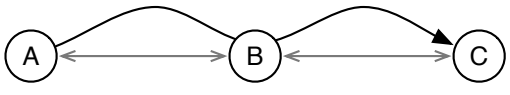


Figure 3.3: 2-hop example scenario

traffic needs. If, hypothetically, the traffic needs at A require it to ask for more than half of the minislots in the data subframe, it will do so. Node B, when receiving the reservation request, has no idea that the incoming packets are destined to C until it actually receives

them, and thus accepts and grants all of the requested minislots to node A. This would be perfectly okay if the destination of the packets was node B itself (node B would not require any minislots for traffic forwarding), but when B then tries to forward the data to C, there aren't enough minislots to forward data at the rate which it is received (due to the single-channel mode of operation). This results in traffic being dropped in node B. In an extreme situation, if node A requests all of node B's minislots, all traffic will be dropped in B since there won't be any slots left for B to forward data to C.

In the given scenario, one could argue that node A knows that B is not the final destination of the packets, and can thus downscale its reservation accordingly. However, node A has no idea of which burst profile is active between B and C; if both burst profiles were the same, a 50/50 minislot distribution would be ideal, but with different burst profiles the minislot ratios would differ.

To overcome this issue, we propose a method of bandwidth balancing where each node is responsible for keeping track of its traffic needs, and of informing neighboring nodes which are sending data of the subsequent limitations. This process is aided by implementing a process to partially cancel bandwidth reservations. On each node, an agent estimates the amount of traffic flowing through the node, and by inspecting the source and destination fields in the PDU headers, determines how much of that traffic will need to be forwarded to another node and the corresponding amount of minislots. By then looking at its own minislot reservations, the agent determines if there is a need to send a bandwidth cancellation message back to the source of the traffic. The amount of minislots in the cancellation request is found with expression 3.1:

$$slots_{cancel} = slots_{source} - \frac{slots_{source} + slots_{available}}{2} \tag{3.1}$$

which is only evaluated if $slots_{available}$ is smaller than the number of slots required for forwarding traffic to the next node. As an example, if the source node is using 40 slots to send data to our intermediate node, and this node only has 8 slots available, then the number of slots to be cancelled at the source will be equal to $40 - (40 + 8)/2 = 16$. By then canceling 16 slots, the source node would be using 24 slots for transmission, and the intermediate node would use the 16 canceled slots plus its 8 available slots, thus also totaling 24 slots.

Figure 3.4 plots the throughput of a bandwidth-demanding traffic flow traversing an increasing number of hops, with the original QoS implementation and with an implementation with our proposed mechanisms for bandwidth balancing. It can be seen that with the original architecture, the effective flow throughput is very low if the flow is traversing more than one hop. With the proposed modifications, the flow can attain a throughput in accord with what is expected for single-radio multi-hop transmissions, being reduced by 1/2 for 2 hops and by 1/3 for three hops.

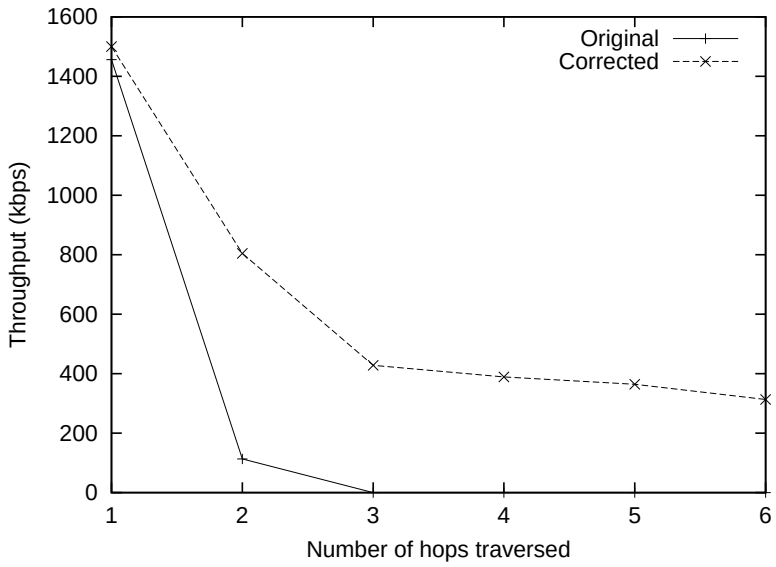


Figure 3.4: Throughput of multihop flows with and without balancing of bandwidth

3.2 Implementation and improvements

In this section we describe the major improvements which were performed on the implementation of the QoS architecture in *ns2mesh*.

rtPS minislot reservation mechanisms

A number of improvements were made to the workings of the rtPS service class. In particular, the algorithm that locates minislots for rtPS to transmit its MSH-DSCH messages in uncoordinated distributed scheduling was fully rewritten. Mechanisms were put in place to avoid conflicting situations where two nodes trying to reserve slots for rtPS DSCH messages to each other would try to allocate slots in the same fashion, resulting in the same slots being reserved on both sides for the same purpose of transmitting scheduling messages, and thereby being unable to communicate these messages. The selection of slots for MSH-DSCH messages is now a three-tiered process: the bandwidth manager first attempts to locate free slots for transmission; failing that, it attempts to borrow slots already reserved for transmission of rtPS packets; and failing that also, it borrows minislots from the other service classes, trying the lowest-priority classes first: BE, then nrtPS, and finally UGS.

Also, instead of always trying to locate 5 minislots for transmitting MSH-DSCH messages (the equivalent of 96 bytes, the typical size of a MSH-DSCH message when using the slowest burst profile), the routine now correctly determines the number of minislots required according to the destination link's burst profile.

Multi-hop minislot balancing

Methods were implemented to cope with minislot reservation issues in multi-hop situations, as was previously described in Section 3.1.3. To implement the cancellation messages without modifying the architecture itself, we adapted GrantIE MSH-DSCH messages to our

purpose. A cancel message is a GrantIE with persistence level set to 'CANCEL'; the fields 'minislot start' and 'minislot range' indicate which block of slots is affected, and the 'service' field indicates which service is to be canceled. The GrantIE message was chosen due to the fact that it can be a directed message (i.e., it has a field for 'Destination NodeID'), and all bandwidth cancellations are directed to a single node.

Support for video trace files

The default tracefile traffic generators in *ns2* are only capable of reading a tracefile with $\{time, size\}$ pairs of data. To support our simulations of video traffic, new classes were required.

A packet in *ns2* is capable of carrying a pointer to an object of class *AppData*, to transport side information with that packet. This class *AppData* can be derived and extended for specific needs. For our requirements, a subclass *VideoData* was created, which, when included in a packet, marks the packet as being of type *VOD_DATA*, and enables the transport of side information such as frame type, frame distortion, and delivery deadline.

To generate these video packets from video trace files, a new traffic generator *VODTrafficTrace*, derived from the *TrafficGenerator* class, was implemented. It works similarly to the original tracefile traffic generator, but is capable of reading extra information from the input tracefile, and creating packets with an attached *VideoData* object containing relevant side information for the video packets. These methods are generic and can be used by any *ns2*-based simulator, by invoking the *Application/Traffic/VODTrace* class in TCL simulation files.

Video trace file packetization tool

To prepare video traces for inputting in *ns2*, we developed 'packetize', a small utility with the following features:

- Accepts terse and verbose H.264 video trace files from ASU (see Section 3.3.4), and also trace files with side distortion information.
- Can packetize the frames in the video trace, by splitting each frame into multiple packets of a user-definable size.
- User-definable packet output rate, in packets-per-second.
- Outputs the results in an *ns2*-compatible format, suitable for using with class VODTraceFile, with side information such as frame type, distortion information, frame delivery deadline.

Figure 3.5 shows the syntax for invoking 'packetize', as given by its help command.

TCL simulation code

The standard TCL files for simulation control provided in *ns2mesh* were also the target of improvements, aimed to improve their versatility and usefulness. Specifically, they now provide:

```
$ packetize -h
Usage: packetize tracefile [-p size] [-f pps] [-o outputfile] [-s|-u] [-a|-b]

Options:
  -p, --packet-size    Size, in bytes, of output network packets
  -f, --pps            Output frame spacing, in packets per second
  -o, --out            File in which to write results
  -s, --strace         Input trace file is of {time,size,mse} format
  -u, --asutrace       Input trace file is an ASU video trace
  -t, --terse          Input trace is in terse format (implies -u)
  -v, --verbose        Input trace is in verbose format (implies -u)
  -a, --ascii          Output in plain-text mode
  -b, --binary         Output in binary mode, suitable for ns2
```

Figure 3.5: Syntax of the *packetize* tool

- Finer tuning of traffic parameters, e.g. per-flow CBR rate and packet size selection.
- Support for randomization of a flow's source node, destination node, or both at once.
- Support for the creation of an arbitrary number of flows with random characteristics with a single command.
- Increased levels of debug, with a controllable debug verboseness parameter.
- Separation between traffic generators from traffic trace files and video trace files.

3.3 Performance evaluation

In order to evaluate the performance of the QoS architecture under study, we ran a series of simulations with various types of traffic and network topologies. We first analyze the performance of the two main service classes of this architecture: UGS and rtPS (the nrtPS and BE classes, as described in Section 3.1.2, are of simple implementation and are analyzed in conjunction with other classes). Then, we study the throughput distribution between the three service classes meant for high data rates: UGS, nrtPS and BE. Finally, we evaluate two mesh topologies with random placement of traffic flows.

For all the simulations in this section, the following parameters were defined: profile *profP3_10*, according to the IEEE 802.16 standard [9], which specifies a channel bandwidth of 10 MHz and a frame duration of 4 ms; burst profile *16 QAM 1/2*, *XmtHoldoffExponent* set to 0, and four control slots per frame. These settings yield a physical data rate of about 13 Mbps in our simulator, although the standard indicates a reference value of only 10,080 kbps.

All simulations presented in this section are run for 30 seconds, and the first 5 seconds of data are discarded. For simulations where the flows have fixed source and destination nodes, each data point is the average result of 10 repetitions. For simulations where the flow's endpoints are randomly distributed, each data point is the average result of 50 repetitions.

3.3.1 UGS service class performance

The first simulation evaluates the performance of the UGS service class. It is expected that the indefinite reservation of bandwidth made for traffic in an UGS service class will lower the average delay of packets in that class, when opposed to the original *ns2mesh80216* simulator

which has no QoS service classes, and where bandwidth requests need to be periodically repeated even if the traffic is of a constant bit rate (CBR) nature. For this simulation, a 5-node *clique* topology was used, as depicted in Figure 3.6. In this topology each node is in the neighborhood of every other node, and thus all communications are made over only one hop.

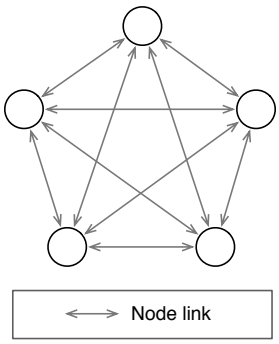


Figure 3.6: 5-node clique topology (5-node, 4-branch multiring)

A 1 Mbps constant bit rate (CBR) flow was introduced in the network. Due to the 1-hop characteristics of the clique topology, the source and destination nodes of the flow are not relevant. We scaled the size of the packets from 100 bytes up to 1800 bytes, in 100-byte increments. The average delay per packet can be seen in Figure 3.7, and is plotted against the results of the same scenario with the original simulator.

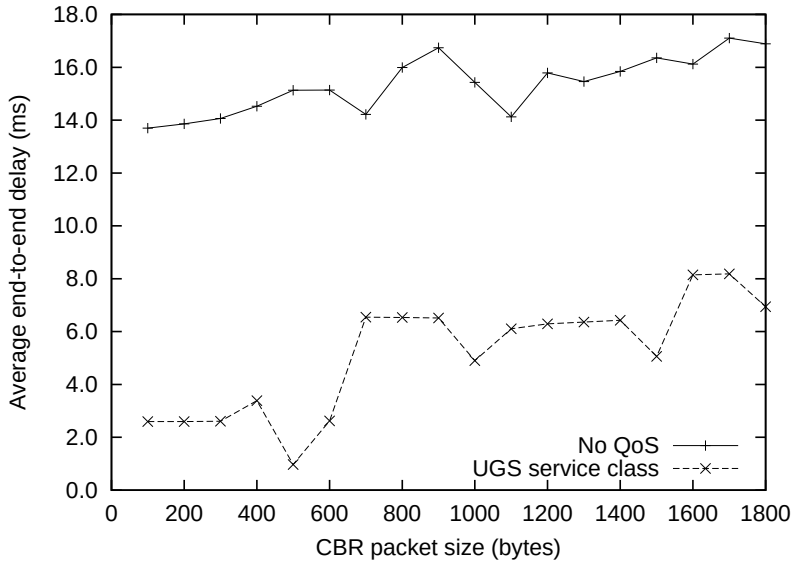


Figure 3.7: Average delay of a CBR flow with increasing packet size

Figure 3.7 evidences that the delay of CBR traffic can be substantially decreased by predicting the transmission requirements and making a reservation for an appropriate number of minislots to handle such traffic. In the original model, the need to periodically re-send minislot requests to handle the traffic has a noticeable impact on the observed delay.

We observe that, in our QoS implementation, packet sizes which are multiple of 500 bytes return the best performance with UGS flows. Let us consider a flow with 500-byte packets now. A 500-byte packet represents a reservation of 21 minislots per frame[2]; inspection of the simulator's internal tables has shown that such a reservation was indeed created for an indefinite period of time.

3.3.2 rtPS service class performance

We now focus on the performance of the rtPS service class. We aim to analyze the performance of this service class with variable bit rate traffic, and to see the effect of bypassing the competition for transmission opportunities in the control channel, by sending MSH-DSCH messages directly in the data subframe. To evaluate the performance of the rtPS service class, we first begin with a *chain* topology. With this topology, we analyze the relation between the average delay of a flow and the number of hops that flow has to traverse in the network, as depicted in Figure 3.8.

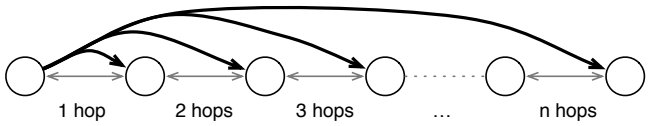


Figure 3.8: N-node chain topology

We simulated flows of VoIP calls traversing an increasing number of hops. The simulation creates the traffic pattern of a VoIP call encoded with the GSM-AMR audio codec, averaging 8 kbps. We then compared the results with the original simulator, with no service classes.

We can see in Figure 3.9 that the use of uncoordinated distributed scheduling considerably decreases the delay of the bandwidth reservation process, and subsequently the delay the packets experience when traversing the network. The delay reduction is particularly noticeable when flows need to traverse long paths, e.g., for up to 7 hops the delay can be kept as low as 50 ms, while the delays the packets experience in the original model can go as high as 300 ms.

Next, we analyze the behavior of the rtPS service class while increasing the network density, i.e., increasing the number of neighbors per node. We chose the *multiring* topology in order to successively increase the number of neighboring nodes, as well as to simulate flows traversing two hops, as shown in Figure 3.10. The increase in network density has a direct impact on the time it takes for a node to win an election for an opportunity to send an MSH-DSCH message in the control frame, so we expect to see a significant difference between coordinated and uncoordinated scheduling for higher network densities.

For this simulation, we again introduced a single VoIP flow in the network, in such a way that it traverses two links. In Figure 3.11 we can see a strong correlation between the delay of a packet and the time needed for the establishment of a three-way handshake agreement. The average delay suffered by the packets is increasingly higher for denser topologies, due to

[2]The conversion from bytes to minislots is given by the equation $minislots = (bytes/\alpha)+1$, with α given by $\alpha = NR\_BITS\_PER\_SYM * NR\_SYM\_PER\_SLOT/8$ (in bytes). In this example, $NR\_BITS\_PER\_SYM = 200$; $NR\_SYM\_PER\_SLOT = 1$; $\alpha = 25$.

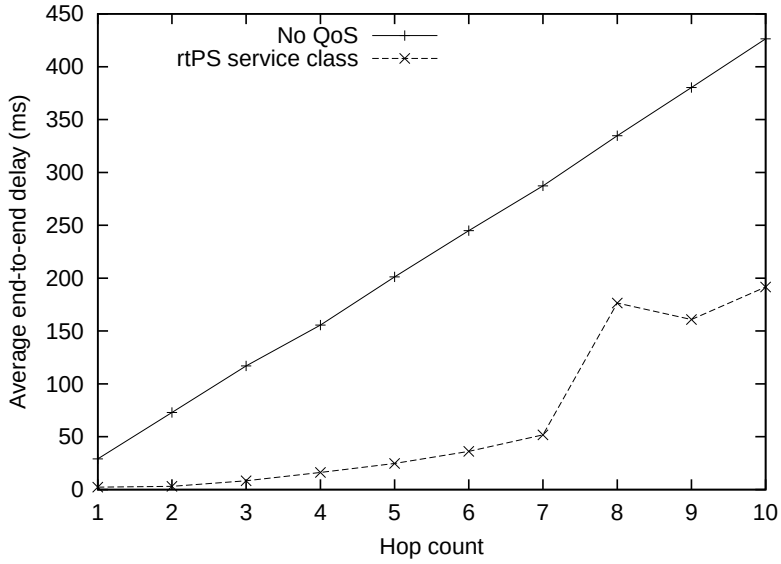
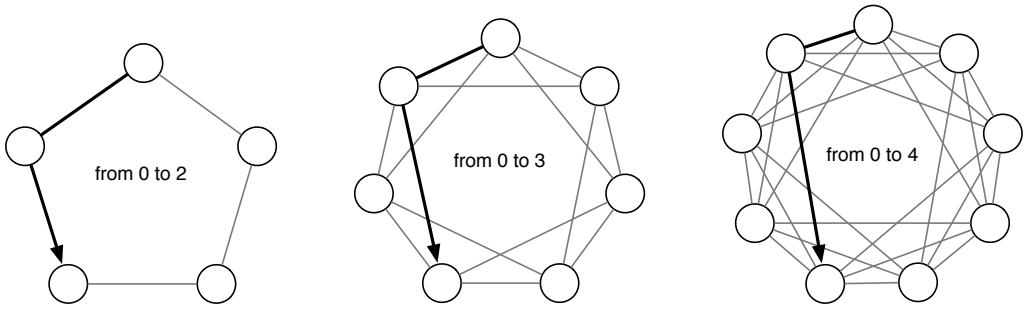


Figure 3.9: Average delay of a VoIP flow with increasing number of hops



(a) 5 nodes – 2 branches (b) 7 nodes – 4 branches (c) 9 nodes – 6 branches

Figure 3.10: Two-hop flows within multiring topologies

increased competition for accessing the control channel. However, that is not seen for our rtPS service class, which by using the data subframe to send bandwidth request message is capable of skipping the need to compete for a transmission opportunity and subsequently decreasing the overall delay for requesting bandwidth. In this simulation, traffic assigned to the rtPS class experienced a constant delay of about 3 ms, regardless of the network density.

3.3.3 Mixed service class performance

In the following simulations we observe the behavior of the three service classes which are designed to support higher data rates: UGS, nrtPS and BE. Let us see how they react when the transmission rates are increased and the available bandwidth is exhausted. rtPS is not meant for high data rate services, due to issues stemming from the uncoordinated reservation of minislots, and thus is not considered here.

For the first simulation, a 5-node clique topology was considered, identical to the one depicted in Figure 3.6. Figure 3.12 shows the throughput of three CBR flows, each assigned

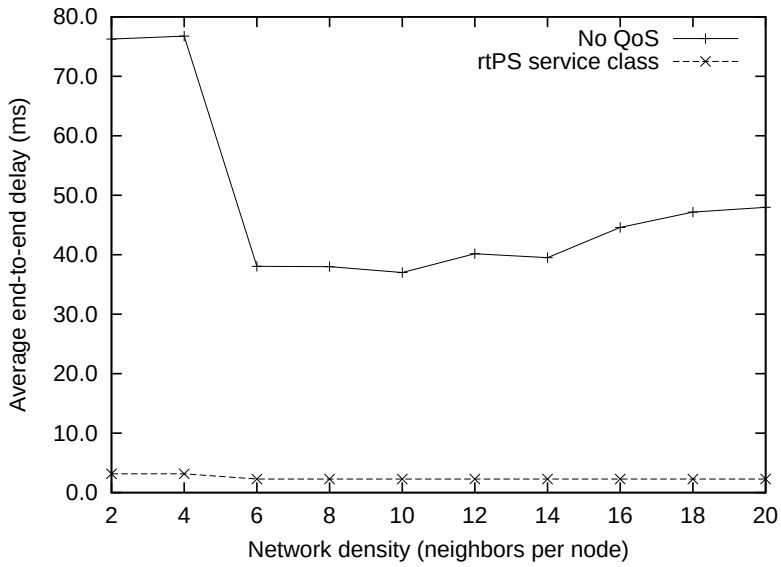


Figure 3.11: Average delay of a VoIP flow with increasing network density

a different service class, with the bit rates of each flow ranging from 1 to 14 Mbps, in 1 Mbps increments. A packet size of 500 bytes was selected, and perfect CBR generation was disabled, for a more realistic simulation. A minimum transfer rate of 1 Mbps was set for the nrtPS service class.

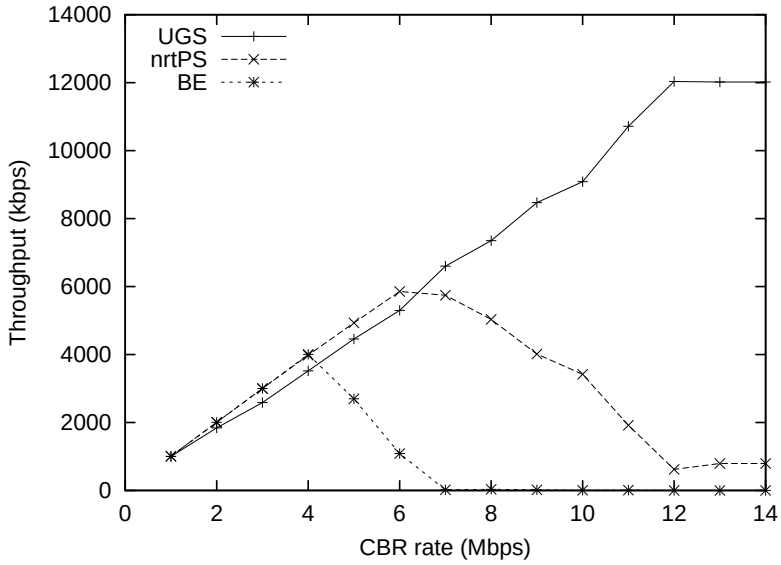


Figure 3.12: Throughput distribution of UGS, nrtPS and BE classes

When bandwidth is insufficient to support all three flows, the UGS class has the highest priority, consuming bandwidth from nrtPS and BE. Notice that after approximately 7 Mbps per flow, the BE service class goes into full starvation and is unable to successfully request any minislots for transmission. This is expected, since each link's maximum theoretical bandwidth

is of about 13 Mbps with profile *profP3_10* and burst profile *16 QAM 1/2*, which is consumed by the UGS and nrtPS flows. We can also verify that the parameter defining the minimum number of nrtPS slots is correctly enforced – the nrtPS flow stabilizes around 1 Mbps under link exhaustion.

Also for the same scenario, in Figure 3.13 we can see the average end-to-end delay of packets in each service class. The flow in the UGS service class is able to keep a very low

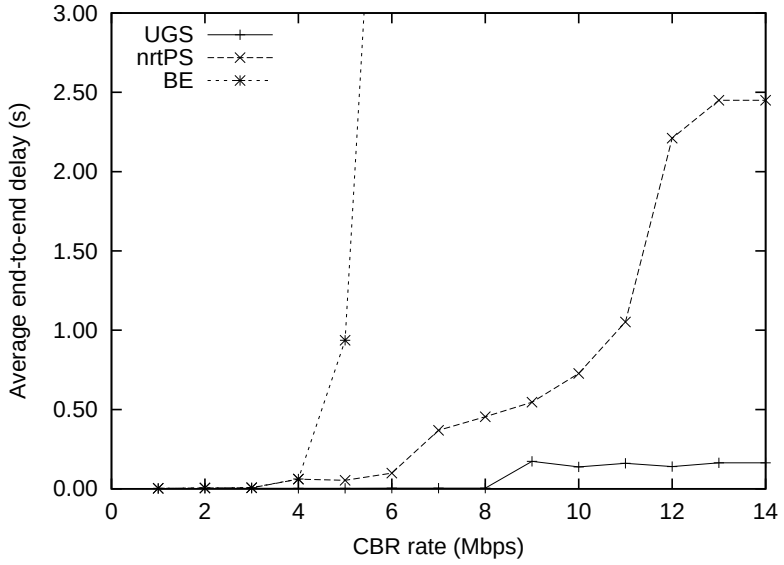


Figure 3.13: Average delay of UGS, nrtPS and BE classes

delay (less than 4 ms) up until 8 Mbps, and from thereon it increases slightly and stabilizes near 17 ms. The delay in BE increases by a very large factor after 4 Mbps which, as it can be seen in Figure 3.12, is the point where its bandwidth starts being restricted. The delay of packets in the nrtPS service class stays below 500 ms up until the 7 Mbps point, where its bandwidth starts being granted to UGS instead. From that point on the delay quickly begins to rise, going up to 2.5 seconds and then stabilizing.

3.3.4 Scenarios with grid and ring topologies

One of the most relevant characteristics of mesh networks is the possibility to deploy networks with versatile topologies. We will now evaluate the performance of our QoS architecture in ring and grid network topologies, with typical user services such as VoIP, video-on-demand, and file transfers.

The video-on-demand is generated from video trace files provided by Arizona State University (ASU) [32]. ASU provides a publicly-available library of trace files generated from encodings of several videos, with a variety of video codecs. In the following simulations, we generate VOD traffic with the 'NBCnews' video trace, a 30fps, 352x288 pixels CIF sample obtained from a TV capture. Our sample is encoded with Main Profile H.264 (non-SVC), a GoP of 16 frames, with 3 B-frames in between I/P keyframes. With the help of the packetization tool developed for this simulator, frames are split into 300-byte network packets, which results in an average bit rate of 700 kbps for this sample.

For the following simulations, traffic types are defined as follows:

- Constant bit-rate (CBR) traffic of 1 Mbps, with 500-byte packets of random payload transported over UDP, assigned to the UGS service class.
- Variable bit-rate (VBR) VoIP traffic, G.711 audio encoding, averaging 90 kbps with 78-byte packets, assigned to the rtPS service class.
- VBR Video-on-Demand (VOD) traffic, from the 'NBCnews' sample trace file, assigned to the nrtPS service class.
- VBR file transfer traffic, FTP protocol over TCP, assigned to the BE service class.

9-node grid network

We simulate traffic in a grid topology with 9 nodes arranged in a square, 3x3 pattern, which can be seen in Figure 3.14. We then successively increase the number of flows assigned to each

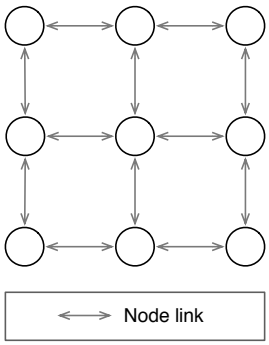


Figure 3.14: 9-node grid topology

service class, and evaluate the network performance. Each flow is placed in the network with a random source and destination node, and due to the random nature of these simulations, traffic may be subject to bandwidth competition, starvation, multi-hop forwarding, and all of these issues may be observed at the same time.

The average one-way packet delay is plotted in Figure 3.15, including the confidence intervals at 95%, as error bars. Despite the unpredictability of this random simulation, it can be observed that voice traffic in the rtPS class exhibits consistently low delays (less than 200 ms, and below 100 ms for no more than 9 flows). CBR traffic in the UGS class obtains the second lowest delays, primarily due to the high throughput requirement set for it (1 Mbps, an 11x markup from VoIP traffic in rtPS). This factor impacts bandwidth reservations, especially when the random positioning of UGS flows causes them to traverse two and three hops, which consumes a large amount of minislots for traffic forwarding and triggers bandwidth cancellation mechanisms described in 3.1.3. Delay associated to the nrtPS service class is relative to the overall network utilization, increasing noticeably when more than 16 flows are present, and traffic in the BE class experiences the highest delay overall.

The average throughput for each service class can be seen in Figure 3.16. Here we can see an approximately linear decrease in effective throughput that follows the saturation of the network with traffic. FTP traffic, which uses the TCP protocol and attempts to consume

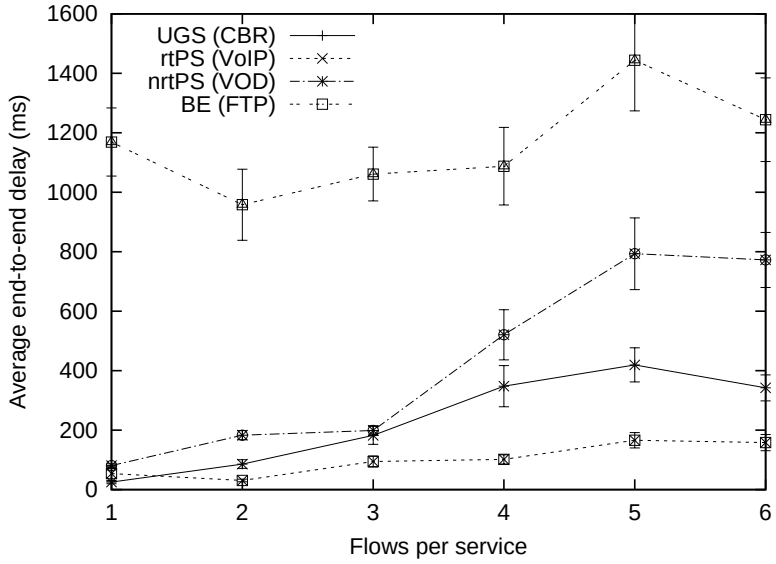


Figure 3.15: One-way delay of traffic in a 9-node grid topology

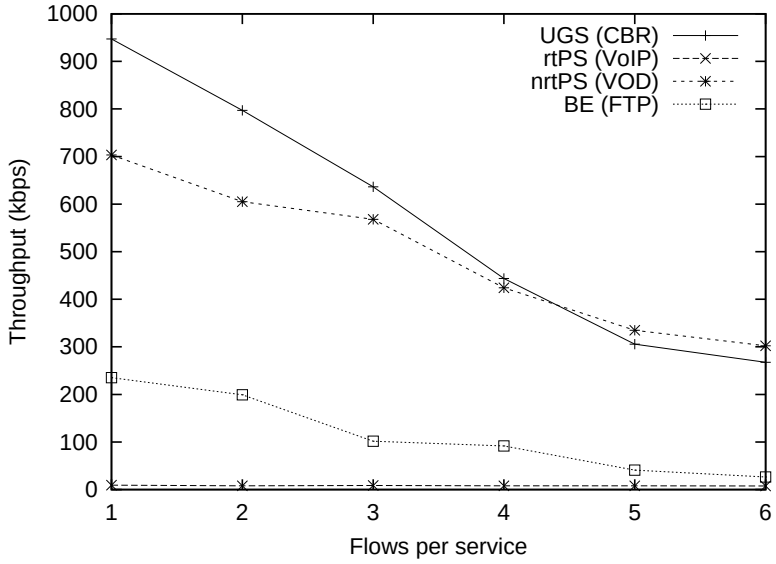


Figure 3.16: Average throughput of traffic in a 9-node grid topology

as much bandwidth available as it can, is placed in the BE service class and is only granted minislots leftover from other reservations. The two service classes aimed to support large bandwidth demands, CBR and nrtPS, are both granted the largest portions of bandwidth in the network as expected. VoIP traffic bandwidth is negligible when compared to the three other classes, and so remains approximately stable. It should be noted that a clean separation in throughput, such as the one previously observed in Figure 3.12, is not observed this time, again due to the randomness associated with flow positioning in this simulation. Flows randomly share and withhold links, leading to the (average) linear decrease observed.

5-node ring network

For the final test case we repeat the previous grid network simulation, but this time with a 5-node network in a ring topology, as depicted in Figure 3.17. In this topology, random placement of traffic flows result in higher probabilities that flows have to share node links (when compared to the 9-node grid topology).

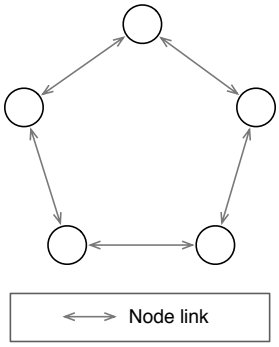
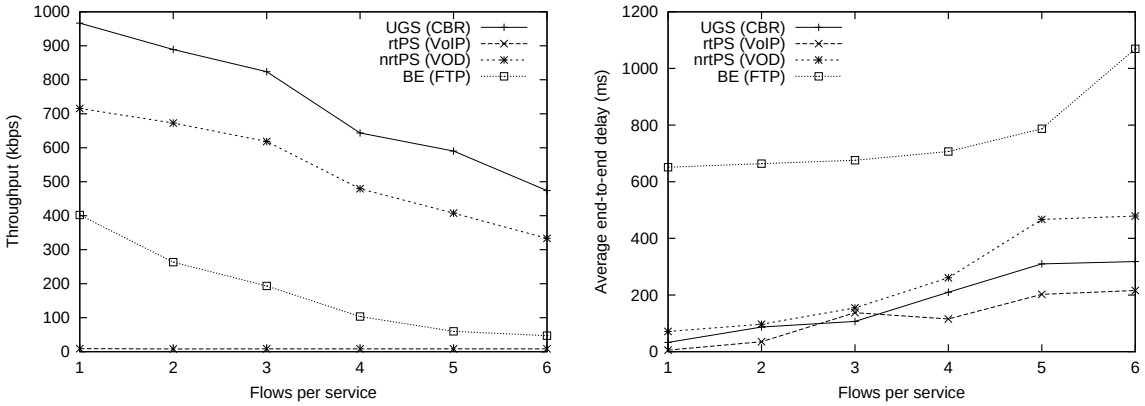


Figure 3.17: 5-node ring topology

The simulation results are in all very similar to what was obtained in the previous simulation, and can be seen in Figure 3.18. Throughput is evenly distributed as the network becomes saturated, as Figure 3.18a demonstrates, with a cleaner separation between the UGS and nrtPS classes than the one seen in the grid scenario. In terms of packet delay, we can observe in Figure 3.18b the same pattern: rtPS is capable of the lowest delays, followed closely by the UGS class. Packets in the nrtPS and BE classes experience much higher delays, comparatively.



(a) Throughput

(b) One-way delay

Figure 3.18: Throughput and delay measurements in a 5-node ring network with QoS

Finally, in Figure 3.19 we have plotted the accumulated throughput of all flows in the network. As one can see in the plotted 'Total Network' line, as more and more flows are added to the network, the total effective throughput reaches a bottleneck near the 5 Mbps mark. This essentially means that random placement of traffic flows results in a network

utilization which is much lower than what the network is capable of supporting. The root of this problem is the number of minislots that are wasted in traffic forwarding, which reinforces the notion that in 802.16 mesh networks a multi-channel approach is important to avoid this waste and maximize effective throughput.

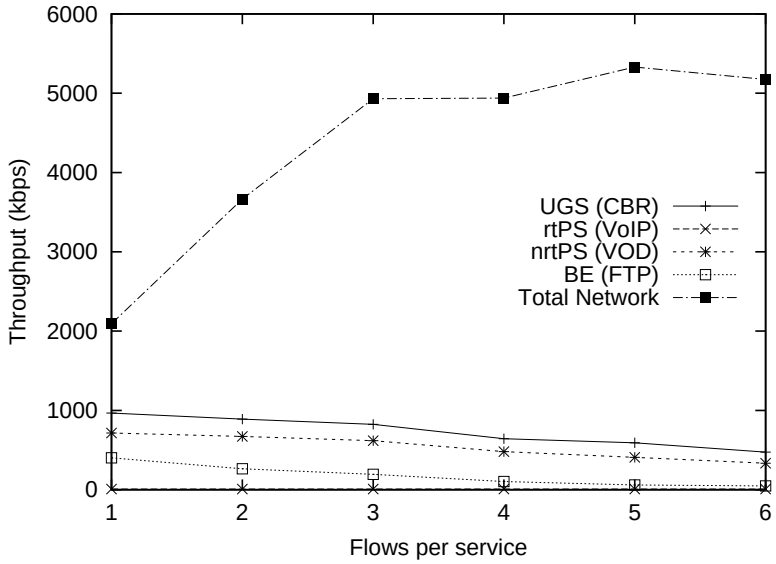


Figure 3.19: Total throughput in a 5-node ring topology

3.4 Summary

In this chapter we have presented an implementation of a QoS architecture for the IEEE 802.16 MESH mode of operation, which is capable of coexisting with the PMP mode of operation by adopting the same QoS mechanisms and service classes as those outlined for the PMP mode, i.e., UGS, rtPS, nrtPS and BE.

Throughout our simulations we were able to observe a number of advantages of the architecture. The bandwidth distribution is optimized between service classes, and QoS requirements of throughput and delay can be met adequately. To handle constant bit rate data flows, the UGS service class can make minislot reservations that do not expire (persistence 7, good until cancelled), thus decreasing the overhead in bandwidth request messages and noticeably decreasing the packet delay of flows with CBR characteristics.

For variable bit rate traffic, the rtPS and nrtPS classes are designed not to make reservations for long periods, since variations in the data rate of these flows is expected. This way, although there is an increased overhead in the number of bandwidth request messages, the probability of having wasted bandwidth (or lack of bandwidth) is reduced, since bandwidth requests can quickly adapt to new traffic requirements. In addition, the QoS architecture uses uncoordinated distributed scheduling to request bandwidth for rtPS service classes, which is important since it bypasses the competition between nodes for access to the control channel, and helps to significantly reduce delay in the rtPS class for networks with a high density of neighbors. This does not apply for nrtPS traffic, where bandwidth is more important than delay, and so it requests bandwidth in the control channel.

Finally, for BE traffic, the QoS architecture attempts to use the remaining unused bandwidth after the reservations for the three other scheduling services. It has no guarantees, and in our simulations we observed that this led to bandwidth starvation when competing with high-throughput flows in the other service classes.

Chapter 4

MOS-based Packet Scheduling in Wireless Mesh Networks

Research in network performance has generally aimed to provide improvements in objective, clearly-measurable traffic parameters. Maximizing effective throughput, link utilization, and minimizing delay and packet loss, with recourse to intelligent packet routing, scheduling, deep packet inspection and QoS provisioning, is typically the main goal.

As was discussed on the introduction to this thesis however, this does not necessarily translate in optimal satisfaction for the end-user. Specific characteristics and traffic patterns of services, such as the streaming audio and video content, show that common network metrics are hardly indicative of the end-users' perception of the quality of the service being provided. Likewise, network mechanisms developed with a purpose of obtaining the best objective-quality parameters (i.e., throughput, delay) from a network may be unsuitable, or less desirable, to the advent of our next-generation content-centric, multimedia-oriented networks, where a focus on the end-user's subjective perception of quality can be expected to yield better results.

In this chapter, we approach the topic of providing QoE in wireless mesh networks, and present methods for a packet scheduler designed to maximize the subjective quality of audio, video and data flows. We also evaluate this scheduler in the *ns2mesh* IEEE 802.16 MESH-mode simulator. We begin with a comprehensive description of the models implemented by this scheduler for the estimation of the subjective quality of video, audio, and file transfer services, in Section 4.1. Then, we describe the scheduler's algorithms and mathematical models for MOS-based scheduling, in Section 4.2. Section 4.3 delves into the importance of meeting specific deadlines in the delivery of audio and video packets, and how this scheduler can be designed so as to make use of this information. The details on the implementation of the scheduler in *ns2mesh*, and its performance evaluation, are given in Section 4.4; Section 4.5 discusses the integration of the MOS scheduler with the QoS models presented in Chapter 3, presenting simulation results; and finally we summarize the outcome of this research in Section 4.6.

4.1 MOS models

MOS models are a critical part of the proposed MOS scheduler, as their accuracy directly reflects on how well the scheduler is capable of keeping distortion across all flows balanced and to a minimum. In this section we present three mathematical models whose purpose is to gather and process various information on video, voice and file transfer protocols, and then provide a subjective measure of the perceived quality in terms of MOS points. The scheduler then uses the values returned by these models to calculate the impact of packet losses and keep overall distortion to a minimum.

4.1.1 Distortion model for video

There is a wide number of literature items on the subject of video quality assessment, as previously discussed in Section 2.2.2. In this thesis we refer to [27], which presents an accurate model for evaluating the impact of packet losses in video quality and accounts for application-specific parameters such as the video codec used, loss recovery techniques, bit rate, packetization and content characteristics. The model is generally applicable to any motion-compensated video scheme, in particular H.264 which is used broadly for the simulations in this thesis.

This model obtains video distortion by first estimating the impact caused by the loss of a single slice (a block of independently coded pixels) of a frame from the video stream. Then, given the video frame structure and the probability of the occurrence of such a loss, a global distortion value is obtained.

Assuming that the expected distortion caused by the loss of a slice is σ_S^2, and that location of that slice is uniformly distributed in $[0, T-1]$, with $T-1$ being the number of predictively coded (P-frames) in the video's group of pictures, then the total distortion D_1 caused by the loss of that slice is given by:

$$D_1 = \frac{\gamma^{T+1} - (T+1)\cdot\gamma + T}{T\cdot(1-\gamma)^2} \tag{4.1}$$

$$= \alpha\cdot\sigma_S^2 \tag{4.2}$$

where γ is an attenuation factor ($\gamma < 1$) to account for the effects of spatial filtering, and varies according to the video characteristics and spatial filtering applied by the decoder. To extend this model to encompass losses from many different lost frames, it is only necessary to average σ_S^2 from all the lost frames and then obtain a value for D_1 which represents the mean distortion.

After obtaining the distortion caused by a single slice D_1, the overall distortion $\overline{D}$ caused to a video sequence is, for the H.264 codec,

$$\overline{D} = s\cdot\overline{n}\cdot P_e\cdot L\cdot D_1 \tag{4.3}$$

In equation 4.3, s is the number of slices per each video frame (typically 2 slices for a QCIF frame, 8 slices for a CIF frame), $\overline{n}$ represents the loss burstiness ($\overline{n}\approx 1$ for Bernoulli losses, $\overline{n} > 1$ for Bursty losses, depending on the aggressiveness of the burst errors), P_e is the packet error rate and L is the number of frames per packet, which depends on the packetization scheme. This model thus includes the effects of the packet loss patterns as expressed by

$\overline{n}$ and P_e, the transmission bit rate expressed in slices per frame with the ratio $s \cdot L$, the packetization strategy as represented by L, and the distortion caused by lost slices from D_1.

This distortion $\overline{D}$ can be mapped to the conventional Peak Signal-to-Noise Ratio (PSNR) with:

$$PSNR = 10 \cdot log_{10} \frac{255^2}{\overline{D}} \tag{4.4}$$

Finally, we can relate PSNR to a Mean Opinion Score value with the following non-linear relation, as suggested in [33], and here adapted to fit the MOS value range:

$$MOS = -\frac{3.5}{1 + exp(b_1 \cdot (PSNR - b_2))} + 4.5 \tag{4.5}$$

where b_1 determines the steepness of the mapping curve, and b_2 represents its central point. For the purpose of our work, we assume that the highest video quality is obtained at a PSNR of approximately 40 dB, and that the lowest video quality is obtained for values below 20 dB. One sample pair of values is then $b_1 = 0.5$ and $b_2 = 30$, which results in the curve depicted in Figure 4.1.

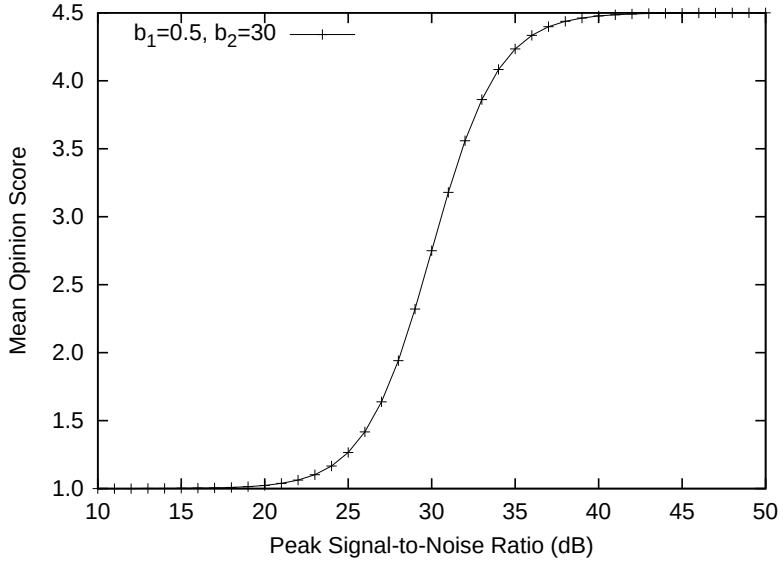


Figure 4.1: Mapping between video PSNR and MOS

It can be seen, in Fig. 4.1, that the mapping is mostly linear for the central values, whereas for the edge values near 20 and 40 dB the relationship essentially saturates. This means that when video quality is at an extreme value, small variations in PSNR will not result in a perceivable variation to the human eye.

4.1.2 ITU-T E-Model for audio

The quality of VoIP calls is adversely affected by loss and delay of its audio packets. The ITU-T E-Model [21] provides parametric estimation and defines an R-factor that accounts for various aspects which impair voice quality. This rating factor R is given by:

$$R = 100 - I_s - I_e - I_d + A \tag{4.6}$$

I_s is the signal-to-noise impairment factor, a combination of all impairments which occur on par with the voice signal. I_d represents the impairments caused by delay, I_e is associated to the impairments caused by losses due to the codecs and network, and A is a compensation factor to account for various user conditions. In VoIP applications, however, only I_d and I_e are typically considered [34, 35]. After choosing the default values for all the other parameters, equation (4.6) can be reduced to:

$$R = 94.2 - I_e - I_d \tag{4.7}$$

This R-factor ranges from 0 to 100 and a score of more than 70 translates to a high MOS, and thus an audio stream of good quality. From the E-Model, we can then translate the R-factor to the MOS scale by:

$$MOS = 1 + 0.035 \cdot R + 7 \cdot 10^{-6} \cdot R \cdot (R - 60) \cdot (100 - R) \tag{4.8}$$

This mapping applies only for $0 \leq R \leq 100$. For $R > 100$, MOS is capped at 4.5, and for $R < 0$, MOS takes the value of 1.

Delay impairments

The total mouth-to-ear delay in a VoIP call is composed by codec delay (d_{codec}), playout delay ($d_{de-jitter\_buffer}$) and network delay ($d_{network}$). d_{codec} is associated with the algorithmic and packetization delays caused by the audio codec; $d_{de-jitter\_buffer}$ (also seen as $d_{playout}$) is the delay associated with the buffer required to negate jitter effects at the receiver; and $d_{network}$ is the one-way delay for the flow's packets in the network. The total delay is the combined sum of the three:

$$d = d_{codec} + d_{de-jitter\_buffer} + d_{playout} \tag{4.9}$$

Once this delay is obtained, its impact I_d on the R-factor is given by:

$$I_d = 0.024 \cdot d + 0.11 \cdot (d - 177.3) \cdot \mathbf{H}(d - 177.3) \tag{4.10}$$

where $\mathbf{H}(x)$ is an indicator function, assuming the value 0 for $d < 177.3$ and 1 if otherwise. The mouth-to-ear delay determines the interactivity of a voice conversation, and has a critical value at 177.3 ms. It is considered that if the total delay exceeds this value, then the interactivity of the conversation is significantly compromised. Thus, equation (4.10) has an additional factor which is only considered if the delay exceeds this critical value.

Loss impairments

Different codecs react in different ways to packet losses in a VoIP call. The impact I_e of packet losses in the R-factor is given by:

$$I_e = \gamma_1 + \gamma_2 \cdot ln(1 + \gamma_3 \cdot e) \tag{4.11}$$

In equation (4.11), e is the total loss probability (in percentage), and the γ_i's are fitting parameters. γ_1 determines the impairment caused by coding, and γ_2, γ_3 represent the impact due to a packet loss for a given codec. Table 4.1 shows the standard values for the three γ parameters for codecs G.729a and G.711.

The loss probability e includes both network losses and playout buffer losses at the receiver, thus being modeled as:

$$e = e_{network} + (1 - e_{network}) \cdot e_{playout} \tag{4.12}$$

Table 4.1: Loss impairment parameters

Codec	γ_1	γ_2	γ_3
G.729a	11	40	10
G.711	0	30	15

Mapping

Figure 4.2 shows the relation between MOS and voice packet delay for packet loss probabilities of 0%, 2%, 5% and 10%. The loss impairment parameters were chosen for the G.711 encoding. In the plot we can clearly see the increased slope after 177.3 ms, the critical delay point required to maintain conversation interactivity. Also, ITU-T suggests the mapping shown in Table 4.2 to relate MOS values given by the R-score model to subjective measures.

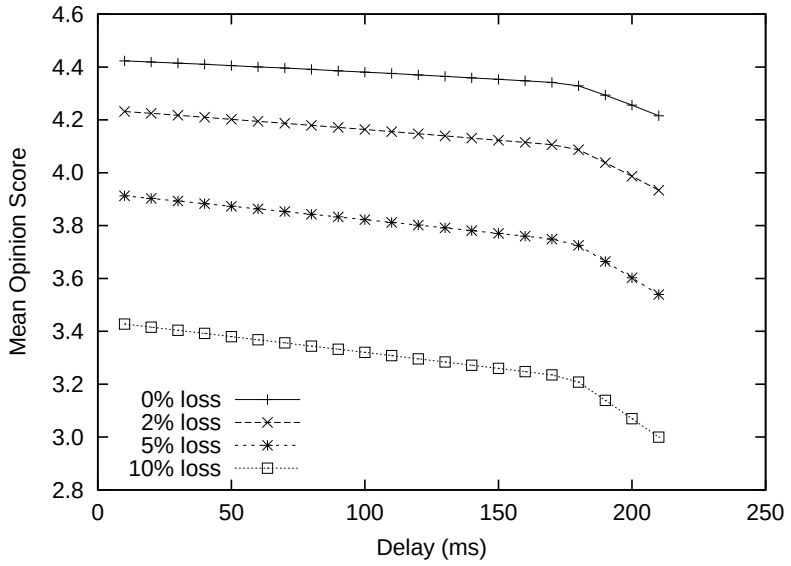


Figure 4.2: Mapping between voice packet delay, loss and MOS for G.711

Table 4.2: Relation between E-Model MOS and user satisfaction

MOS (lower limit)	**User satisfaction**
4.34	Very satisfied
4.03	Satisfied
3.60	Some users dissatisfied
3.10	Many users dissatisfied
2.58	Nearly all users dissatisfied

4.1.3 General model for file transfer servicess

The user satisfaction for a file transfer service is solely dependent on the data rate obtained by that same service. To that end, we consider that each user has a given rate expectation which corresponds to the best user satisfaction. The MOS can then be estimated by the following model:

$$MOS = a \cdot log_{10}[b \cdot R \cdot (1 - Pe)] \tag{4.13}$$

Where:

R - data rate the service is being given

Pe - packet loss percentage the service is experiencing

This model is presented in [36], which considers that the utility of an elastic traffic (such as a file transfer) is an increasing, strictly concave, and continuously differentiable function of throughput. Given that, the logarithmic MOS-throughput relationship in (4.13) is suggested as a way to measure user satisfaction.

The parameters a and b are obtained from the upper and lower rate expectations for the service. If the user is being given a data rate equal to or higher than its expectation, and if there is no packet loss, then the user satisfaction metric should return *4.5*. If, however, the data rate is equal to or below a certain threshold considered to be a minimum expectation for the service being provided, then it is assigned a MOS value of *1*.

In Figure 4.3 the output of this model is shown for varying transmission rates and packet loss percentages, with a and b selected so as to fit the ideal logarithm curve to return a MOS of 4.5 for a data rate of 400 kbps, and a MOS of 1 for 10 kbps ($a = 2.1$, $b = 0.3$).

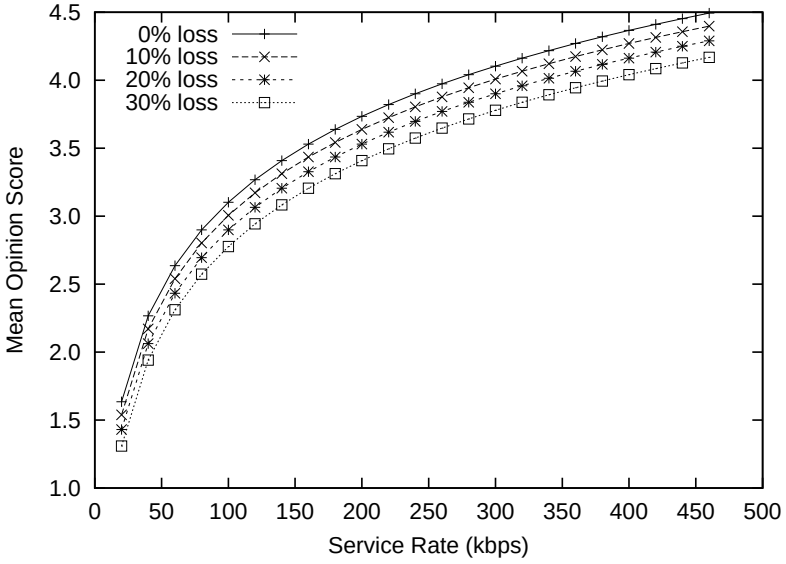


Figure 4.3: Mapping between file transfer data rate and MOS

4.2 MOS balancing

The main goal of this scheduler is to obtain the best possible MOS for all flows in the network. In order to do so, an active queue management algorithm is triggered once the buffer utilization reaches a defined threshold. The scheduler then determines, based on the outgoing data rate and buffer fill rate, how much data it needs to drop from the buffer so as to keep it from overflowing. The principle is the same as the one of a Random Early Detection algorithm.

The scheduler then proceeds to locate sets of packet combinations that, if dropped from the buffer, would satisfy the required reduction in incoming traffic. For each of these packet drop combinations matched, a prediction of the resulting decrease in MOS for each flow is obtained. Each node keeps local statistics of packet drop rate and experienced delay in order to obtain these MOS estimations. The scheduler then chooses the combination of packets that will result in the smallest average MOS decrease for all flows, and drops it from the buffer.

Also considered is the fairness of the drop decision. We aim to obtain as little decrease in MOS as possible while at the same time trying to keep the impact on each flow as small as possible. We thus introduce a penalty into each packet drop combination based on the standard deviation of the MOS decrease values, which will favor the combinations that keep the overall impact on each flow to a minimum.

Finally, we propose that each node should keep track of how much penalty it has inflicted to a given flow, and forward that information to the subsequent nodes if required. For audio and data flows, the penalty can be easily inferred by checking the sequence numbers of packets and registering the loss rate; however, for video flows, when a packet is dropped the penalty on the flow's quality is dependent on the distortion information contained in that packet. Thus we propose that, for video flows, each node attaches a header to the packets it is forwarding with the distortion information of the packets it has dropped from that flow. The information then travels the flow's path and allows each node to estimate the global overall distortion on that flow.

4.2.1 Mathematical model

For an n number of flows, and a set of p packet drop combinations, the optimal packet drop combination can be found by minimizing:

$$Q(p) = \sum_{i=1}^{n} k_i \cdot \Delta MOS_p^i - \lambda \cdot \sum_{i=1}^{n} \Delta R_p^i + \mu \cdot n \cdot \sigma(\Delta MOS) \tag{4.14}$$

where $Q(p)$ is the cost function of combination p. In (4.14), ΔMOS_p^i is the reduction in the MOS of flow i due to the packet drop combination p. k_i is a coefficient value for each flow, which can be used to favor or disfavor packet combinations which drop packets from certain flows. This may also be used to increase the likelihood that the scheduler will drop packets from certain classes of traffic, i.e. data streams instead of video and audio streams. ΔR_p^i is the rate reduction in flow i due to the drop combination p, which is subject to a multiplier λ which can control the aggressiveness of the dropping decision. The choice of the value for λ will determine the weight of combinations which drop the highest amounts of traffic. Finally, $\sigma(\Delta MOS)$ is the standard deviation of the MOS reduction in all of the flows. This is

multiplied by n, so as to make it independent from the number of flows being analyzed, and by a coefficient μ, which is used set the relevance of the standard deviation on the evaluation of the drop combination.

This generic model is fit to be applied directly on top of a list of all packets present in the buffer. In our approach however, as mentioned in Section 4.2, a preselection of packet combinations is done based on a desired reduction in buffer size. The rate reduction component $\lambda \cdot \sum_{i=1}^{n} \Delta R_p^i$ in Eq. (4.14) is thus determined to be irrelevant and is dropped from the model, resulting in the following minimization function:

$$Q(p) = \sum_{i=1}^{n} k_i \cdot \Delta MOS_p^i + \mu \cdot n \cdot \sigma(\Delta MOS) \tag{4.15}$$

To obtain a set of packet combinations suitable in size to drop from the buffer, we first determine an amount L of bits to be dropped from the queue based on the buffer fill rate R_{buff}, the outgoing data rate R_{tx}, and the buffer fullness λ_{buff}:

$$L = (R_{buff} - R_{tx}) \cdot \lambda_{buff} \tag{4.16}$$

Then, a range of $[C_{min} \cdot L, C_{max} \cdot L]$ is chosen, and a set S of n_S packets is selected so that:

$$C_{min} \cdot L < \sum_{i \in S} size(S_i) < C_{max} \cdot L \tag{4.17}$$

I.e., the total size of all packets on the set must be within a predefined range around L, defined by C_{min} and C_{max}. In the event that this restriction fails to match a set of packets, the range should be extended until at least one suitable set of packets is found.

4.2.2 Model limitations and caveats

Due to restrictions on our simulation environment, we consider that all links are logical and thus no physical information is directly available (i.e.: path loss, bit error rate). Link burst profiles and theoretical transmission speeds are available, however. We also consider our model independent of the routing component, and thus no alternate routing decisions for packets are made. Initially, for evaluating the performance of the scheduler, we use the *ns2mesh* simulation unmodified – the MAC layer does not implement QoS classes or differentiated buffers, so the scheduler acts directly on a single buffer. The MOS scheduler is then integrated with the QoS architecture described in Chapter 3, and this limitation is overcome.

4.3 Delay-awareness

Due to the nature of a mesh network, packets needing to traverse multiple hops may be subject to excessive delays. This is of particular importance for audio and video flows, where packets have a given delivery time deadline which, if exceeded, will render the packet useless or significantly decrease the perceived quality of the stream. We propose to analyze the delay requirements of each packet arriving and waiting in the transmission buffer.

4.3.1 Scheduling with delay constraints

In a mesh network, packets are frequently expected to be traveling multiple nodes until they reach their intended target. Intermediate nodes forwarding audio or video packets which possess an associated delivery deadline value can, in possession of an estimate of link delays until the final destination of the packets, perform a number of actions according to this deadline. For video flows, this deadline is typically included in the video frame trace files, but the size of the receiver's buffer should also be taken into consideration since it can compensate for higher delays and jitter. For audio flows, one possible deadline is, e.g., the 177.3 ms mark. The importance of this delay value has been detailed in Section 4.1.2.

The first course of action is to drop the packets which will not be able to meet their delivery deadlines – in order to perform this, each node is required to keep track of the delays of every link in the network, and then obtain the expected delay to the packet's final destination.

The second course of action is to reorganize packets in the buffer so as to enable more packets to meet their delivery deadlines. With information on the experienced, expected, and threshold delay of each packet, the scheduler can reorganize packets in the buffer, moving forward those which are nearing their threshold (taking into account the expected delay to destination) at the expense of those who still have a delay margin to work with.

A third course of action, if a tight integration between the scheduler module and the routing module is possible, is to perform intelligent routing decisions based on these delay estimates, by redirecting delay-sensitive packets across paths with shorter delay budgets, while sending the bulk data from services such as file transfers through the paths with higher delay budgets. The proposed scheduler model does not consider this last methodology.

4.3.2 Obtaining and disseminating link delay estimates

For very dense and large mesh networks, it becomes unwieldy to maintain a list of expected delays for each link. We propose two solutions to overcome this issue.

The first solution consists in using existing IP-based methods, specifically *traceroute*. This approach is advantageous due to the fact that it requires no modifications to the lower-layer protocols. Each node can send UDP datagrams along a flow's path, varying the Time-to-Live value of the UDP packet (or make use of ICMP echo requests), to obtain the delays associated with each link. The downside of this approach is the large number of packets which are generated for dense networks and flows with numerous hops.

For the second solution, we propose the creation of Path-Delay Discovery messages at the MAC layer to keep track of delays. These messages are sent by the source node of a given flow to the destination node. On each intermediary node, that node adds relevant delay information to the message before forwarding it. Finally, the destination node sends the message back to the source node once it receives it, so that all nodes can register the delay values. This way we keep overhead to a minimum, by tracking delays solely for links which are being used in the network, all with a single message.

To carry the information, we propose each delay to be stored in a maximum of 10 bits (up to 1024 values of milliseconds), where the highest value represents a delay equal to or larger that 1 second. In order to keep the messages within a reasonable size, we propose a limit of 8 delay values per message (and thus tracking up to 8 links' worth of information), which

results in a 10-byte message, a reasonable value for a MAC management message. The rate at which these messages are retransmitted should ideally be dynamically adjusted at each node, in order not to saturate the MAC control subframe. It should be noted that these messages can also be transmitted in the data subframe, since the delay information does not need to be passed to all neighbors of a node, but only to the next one on the flow's path.

4.4 Performance evaluation

In order to evaluate the performance of the proposed scheduler architecture, we implemented the scheduler, MOS prediction routines and flow distortion tracking agents in the *ns2mesh* simulator, and carried out a number of simulations with varying topologies and traffic types. In this section we first describe in greater detail how the implementation was achieved, then we list the values chosen for each parameter of the scheduler and MOS routines and explain how they were determined, and finally we present each simulation scenario and its respective results.

The *ns2mesh* scheduler has been discussed previously in Section 2.3.

4.4.1 Implementation

The implemented architecture acts on top of the existing packet scheduler present in *ns2mesh*, which is responsible for pushing packets from the buffers into the minislot reservations done by management messages. We set the queue management algorithm to act on the node's buffer on two events:

(a) - the buffer size exceeds a user-defined threshold, in bytes or percentage

(b) - a packet is about to be dropped due to insufficient space in the buffer

Typically, only the first condition is triggered. Condition (b) only triggers for very large packets and very small buffer sizes, where a packet might be too large for the buffer even before a given buffer fullness threshold set for (a) is triggered.

Once the queue management algorithm is called, it first begins by profiling the packets in the buffer, to determine their size, distortion information (for video packets), and to make a list of which Flow IDs are going through the node. Then, equation (4.16) is applied to determine the amount of bytes that are to be dropped from the buffer (from the equation, a range of bytes). The scheduler then proceeds to evaluate all packet combinations for their cumulative size, to satisfy (4.17). The number of combinations, in the full-lookup method, is equal to $2^{npackets}$. Each combination ID is a base-10 integer, and its base-2 equivalent represents an array of boolean values, each value indicating whether the packet in that position belongs to the combination ID. This way, it is straightforward to map a combination ID to a list of packets in the buffer. An example of this mapping, where the combination ID 466 is applied to a 12-packet buffer, can be seen in Figure 4.4.

Once a list of fitting packet combinations is obtained, the MOS impact calculations are performed. For each combination ID, the list of packets which it defines (found via the combinationID's base-2 representation) is fetched; then, the algorithm goes through each Flow ID, gathers all packets that belong to that flow, and computes the new flow parameters that would result from dropping those packets (e.g., the new packet loss rate, or the new total

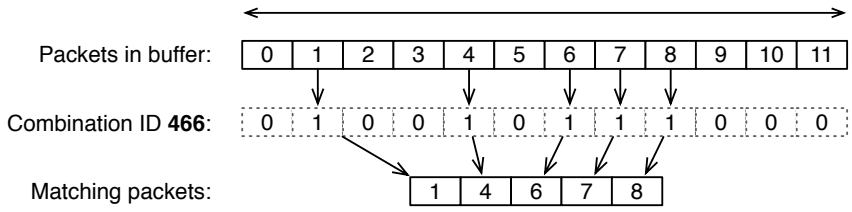


Figure 4.4: Mapping a base-10 combination ID to a list of packets

distortion). With these new parameters, the MOS prediction routine for that flow's traffic type is invoked, and the difference between the old and new MOS (ΔMOS) is determined. The overall total and mean reduction in MOS is calculated from the resulting ΔMOS values of each flow affected by the packet combination, as well as the standard deviation (σ^2) of those values.

Finally, given the list of packet combinations and their respective total, mean and σ^2 MOS impact values, the mathematical minimization function in equation (4.15) is applied to find the best combination of packets to be dropped. The scheduler then drops those packets, and updates the respective flow statistics.

A distortion estimation agent, required for the scheduler to operate, is also attached to each node in the network. This agent is responsible for keeping track of all the packets that are flowing through the node, as well as estimates of the data required to compute MOS values. For video flows, a list of lost packets, their distortion information, and a packet loss rate are required. For audio flows, the average packet delay and packet loss rate is sufficient to obtain the MOS of the flow. For file transfer flows, a throughput measure and the flow's packet error rate are required.

The mechanisms for scheduling based on packet delay deadlines were not implemented in this work, and thus are not considered in simulation.

Scheduler parameters

The scheduler implementation has a number of definable settings that control the scheduler's behavior. Table 4.3 lists these parameters and their meaning.

Table 4.3: Description of MOS scheduler parameters

Parameter	**Description**
enable	Enables and disables the MOS scheduler. Useful for quickly switching between the *ns2mesh* scheduler and the MOS model.
triggerinterval	If set, the scheduler will trigger periodically with an interval defined in milliseconds.
global	If true, delay and throughput statistics are kept globally with the *ns2measure* utility. If false, statistics are kept locally on each node.
buffertrigger	If set, the scheduler will automatically trigger once the buffer reaches a certain threshold. The scheduler assumes a percentage factor if $buffertrigger < 1$, and a number of bytes if $buffertrigger > 1$.
buffertarget	The target set for the final buffer size once the algorithm is triggered. In percentage or bytes.
reductionmargin	Acceptable range around *buffertarget* (C_{min} and C_{max} from Equation 4.17).
ncombs	The maximum number of combinations the scheduler will evaluate for size-based matches. Set as a power of two factor (2^{ncombs}).
maxcombs	The maximum number of size-matching packet combinations the scheduler will try to obtain. Infinite if set to zero.
mosweight	Weight of each flow's current MOS in the scheduler decision process (negative values give less impact to flows with lower MOS).
w_vod, w_voip, w_ftp	Weight coefficients specific to VOD, VoIP and FTP flows.
stddev	Weight of the standard deviation coefficient.
	Video Model
v_gamma	Video model attenuation factor.
v_pfpgop	Number of video P-frames per GoP.
v_spframe	Number of slices per video frame.
v_fppacket	Number of video frames per packet.
v_ltype	Video packet loss factor (packets affected per error).
v_b1, v_b2	PSNR-to-Quality curve fitting parameters.
	Audio Model
gamma1, gamma2, gamma3	Codec parameters for the audio E-model.
	FTP Model
data_a, data_b	Curve fitting parameters for the FTP MOS model.

Scheduler operation example

The following excerpts are examples of the scheduler's operation, obtained from a simulator debug of a scenario with video, voice and data flows. When the scheduler's queue management algorithm is triggered, a number of steps are performed. We present these steps illustrated with pseudocode excerpts of the scheduler's algorithms, when opportune.

1. Triggering and listing packets in the buffers

```
1.883994278 WMOS::trigger    [1] MOS Scheduler timer fired
    Packets in the buffers:
        FTP_DATA  fid 16 ndx 0 id 1 size 54
        FTP_DATA  fid 16 ndx 0 id 2 size 54
        VOD_DATA  fid 2 ndx 0 id 17 size 2678 distortion 28600.000000
        VOIP_DATA fid 11 ndx 0 id 18 size 176
        VOD_DATA  fid 2 ndx 0 id 18 size 1013 distortion 68.000000
        VOIP_DATA fid 11 ndx 0 id 19 size 176
        VOD_DATA  fid 2 ndx 0 id 37 size 1138 distortion 243.000000
        VOD_DATA  fid 2 ndx 0 id 38 size 2678 distortion 4460.000000
        FTP_DATA  fid 19 ndx 0 id 8 size 1078
        FTP_DATA  fid 19 ndx 0 id 9 size 1078
        (...)
    Flow IDs in the buffer: 16 2 11 19 1 9 0
        fid 19 traffic DATA mos 3.375655
        fid 10 traffic VOIP mos 4.362723
        fid 1 traffic VOD  mos 4.013306
        fid 9 traffic VOIP mos 4.362723
        fid 11 traffic VOIP mos 4.362723
        fid 2 traffic VOD  mos 1.000246
        fid 0 traffic VOD  mos 4.500000
        fid 16 traffic DATA mos 1.761583
    Buffer usage 69881/70000, 99.83%
    79 packets in the buffers
```

The scheduler begins by extracting packets from the simulator's buffer. This buffer is stored in a C++ Standard Template Library (STL) data structure that only allows insertion and removal of items in it (random access to items is limited). In order to facilitate work by the algorithms, we first begin by extracting all of the packets from the buffer and placing them in a *vector*-type data structure, named 'pduList', which allows random access, insertion, and removal of items. The status of the buffer and the list of packets in it is then processed and presented to the user.

2. Obtaining a byte range and locating suitable packet combinations

```
    evaluating 262144 combinations for size match
        combination matches for [9681,11081]:
        ID: 5463 5468 5469 5470 5471 5492 5493 5494 5495 5500 5501 5502 5503 5588 5589
5590 5591 5596 5597 5598 5599 5620 5621 5622 5623 5628 5629 5630 5631 5972 5973 5974 (...)
```

To compute the byte range for reduction, the scheduler first finds *targetpoint*, which is the desired final size for the buffer. This is controlled by the parameter *buffertarget*. Then, the amount of bytes required to be removed from the buffer (*bufferreduction*) is obtained, and finally the left and right bounds of the byte range (*lbound*, *rbound*) are determined. The following code illustrates this process:

```
1 targetpoint = maxBufSize * buffertarget
2 bufferreduction = bufSize - targetpoint
3
```

```
lbound = bufferreduction - maxBufSize * reductionmargin
rbound = bufferreduction + maxBufSize * reductionmargin
```

Where *bufSize* and *maxBufSize* are the current and maximum sizes of the buffer, respectively. The next block of code will then evaluate each packet combination for the combined size of the packets contained within it, and if it matches the byte range. It also illustrates how the mapping between a combination ID and its base-2 equivalent is used to select packets:

```
vector validCombs; // to store matching combinations

// evaluate each combination until the limits set by ncombs and maxCombs
for combID from 0 to ncombs and validCombs.size < maxCombs
{
   binComb = dec2bin(combID); // convert the combID into a binary value

   // run through all packets in the buffer
   for packetID from 0 to pduList.size
   {
      // if this packet belongs to the combination, increase the combination size
      if binComb[packetID] == TRUE
         combsize += packetID.size; // fetch the packet size and store
   }

   // if the size of this combination is within the byte range, store its ID
   if  rbound > combsize > lbound
      push combID into validCombs;
}
```

At the end of this routine, the vector *validCombs* contains all combination IDs whose size is within the bounds of the byte range determined earlier.

3. Evaluating the MOS impact of each matched combination

```
5000 combinations matched, processing...
     combID 31117:
          FTP_DATA  fid 16 ndx 0 id 1 size 54
          VOD_DATA  fid 2 ndx 0 id 17 size 2678 distortion 28600.000000
          VOIP_DATA fid 11 ndx 0 id 18 size 176
          VOIP_DATA fid 11 ndx 0 id 20 size 176
          VOD_DATA  fid 2 ndx 0 id 20 size 3128 distortion 27100.000000
          VOIP_DATA fid 11 ndx 0 id 22 size 176
          VOD_DATA  fid 2 ndx 0 id 22 size 932  distortion 76.000000
          VOIP_DATA fid 11 ndx 0 id 23 size 176
          VOD_DATA  fid 2 ndx 0 id 23 size 2928 distortion 26600.000000
          (-) FTP  fid 16 drop 1 delta -0.350450 oldMOS 1.761583 newMOS 1.499212
          (-) VOD  fid 2 drop 4 delta -3.549635
          (-) VOIP fid 11 drop 4 delta -1.326908 oldMOS 4.362723 newMOS 3.253951
          (>) combID 31117 flows impacted 3 MOSimpact total -5.226994 avg -1.742331
               std 1.338688
```

Here, the scheduler extracts the packets belonging to each valid combination ID in *validCombs*. Then, for each combination, the packets associated to it are fetched and grouped by flow ID. These groups of packets are passed to the MOS calculation routine respective to the type of the flow traffic they belong to (i.e., video, audio and data flows

each have its own routine). Each routine then fetches the current MOS of the flow, simulates the resulting MOS caused by dropping the group of packets passed to it, and returns the MOS decrease. With these values, the scheduler is capable of computing the total and average MOS decrease of all flows, and the standard deviation of the decrease.

4. Applying the algorithm and listing the cost function result of each combination

```
synopsis of combinations:
    combID 340:    total -3.549386 avg -3.549386 std 0.000000 impact -3.549386
    combID 341:    total -3.899836 avg -1.949918 std 1.599468 impact -5.499304
    combID 342:    total -3.899836 avg -1.949918 std 1.599468 impact -5.499304
    (...)
```

With the information obtained in the previous step, the scheduler's mathematical model is applied, resulting in an *impact* value, or cost function.

5. Determining the best packet combination and dropping it from the buffer

```
Selecting combID 17492 for drop, impact -3.549362
    Dropping packet fid 2 ndx 0 id 17 size 2678 type 15
    Dropping packet fid 2 ndx 0 id 18 size 1013 type 15
    Dropping packet fid 2 ndx 0 id 19 size 883 type 15
    Dropping packet fid 2 ndx 0 id 21 size 948 type 15
    Dropping packet fid 2 ndx 0 id 23 size 2928 type 15
Resulting buffer:
    Buffer usage 59128/70000, 84.47%
```

Finally, the combination with the lowest *impact* is selected, and its corresponding packets are dropped from pduList. The scheduler then reconstructs the buffer queue from pduList, and resumes the simulator's normal operation.

4.4.2 Methodology

To evaluate the performance of the MOS packet scheduler, we ran a series of simulations, first with only video traffic in the network, and then with audio, video and file transfer services. All of the simulations are run for 10 seconds (simulation time), and the first second of measurements is discarded to account for the network setup and link establishments. The following simulation parameters are set:

- Profile *profP3_10*, according to the IEEE 802.16 standard [9], which specifies a channel bandwidth of 10 MHz and a frame duration of 4 ms.
- Burst profile *QPSK 1/2*, for a maximum physical data rate of 5.82 Mbps.
- Scheduler set to trigger on buffer overflow, find combinations of 15% the buffer size, with a 1% margin (14% to 16%)
- 2^{20} combinations evaluated for size (1.048.576), no limit on matching combinations.
- $mosweight = -0.05$: less penalty inflicted on flows with lower MOS.
- $w\_ftp = 0.8$: FTP packets are 20% more likely to be dropped.
- Video parameters:

  - Attenuation factor $v\_gamma = 0$.
  - P-frames per GoP $v\_pfpgop = 10$.
  - Slices per frame $v\_spframe = 1$.
  - Frames per packet $v\_fppacket = 1$.
  - Loss factor $v\_ltype = 1$, Bernoulli-type losses (1 packet affected per loss).

- Audio parameters:
  - $gamma1 = 0, gamma2 = 30, gamma3 = 15$, matching codec G.711.

- FTP parameters:
  - $data\_a = 2.1, data\_b = 0.3$: Maximum MOS for throughput $\geq$ 400 kbps, minimum for $<$ 10 kbps.

Traffic characteristics

The VOD, VoIP and FTP traffic types in the following simulations are defined as follows:

- **VOD** - VBR traffic from the 'foreman' video sample. This is a 10-second video sample widely used in video simulation work. The sample in question was provided by the authors of [37], a trace file which includes a side information field with the MSE values associated with the loss of each packet.
- **VoIP** - VBR traffic, G.711 audio encoding, averaging 90 kbps with 78-byte packets.
- **FTP** - VBR file transfer traffic, FTP protocol over TCP.

4.4.3 Scenarios with video traffic

Video streaming is the service with the highest potential for noticeable improvements in MOS due to the actions of a MOS-aware scheduler. While in audio streams and file transfers the packets are very similar to each other, in video, as explained in Chapter 2, there are certain types of frames which have a much higher contribution to the overall video quality than other types. From highest to lowest importance, we have I-frames (intra coded picture), P-frames (predictive coded picture), and B-frames (bidirectionally predictive coded picture). Losing a B-frame from the stream can result in minor blurriness and loss of detail – but losing an I-frame will immediately cause skipping, blockiness, and other very undesirable effects.

For our first set of simulations, we choose to evaluate a number of scenarios where the only types of traffic in the network are video streams. We can then judge the performance of the simulator if only applied to the video traffic in the network, and later on compare these results with more complex scenarios involving voice and file transfer communications.

Video scheduling evaluation

For the first scenario, we set up a simple 4-node network as depicted in Figure 4.5. Two VOD flows are created from node 1 to node 3, and another two are created from node 2 to node 3. This scenario is of interest due to the fact that bandwidth saturation should only

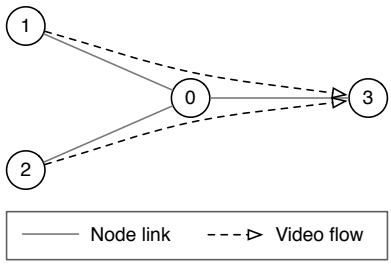


Figure 4.5: Topology for the first video simulation

occur at the central node, node 0. This node will need to balance traffic reservations in order to be able to receive traffic from nodes 1 and 2, and to forward that traffic to node 3.

The results from this simulation are shown in Table 4.4. As it can be observed, the MOS scheduler yields a very noticeable increase in mean MOS, from around 1.21 MOS points to 3.28 points. On a subjective scale, this represents an increase from 'Bad' to 'Fair' in user-perceived quality. Also, the simulator statistic *rd_scheduler_triggered* reports that the queue

Table 4.4: MOS results for the first video simulation

FlowID	**MOS (original)**	**MOS (w/ scheduler)**
(0)	1.23878	3.10224
(1)	1.20280	4.45057
(2)	1.21484	4.38503
(3)	1.19744	1.19170
Average	*1.213465*	*3.282385*

management algorithm was triggered 154 times in node 0, and zero times for the other nodes, which is in agreement with our expectation that the bandwidth saturation would occur in node 0.

Video scenario, 9-node grid topology

We now move to a grid topology with 9 nodes arranged in a square, 3x3 pattern which can be seen in Figure 4.6. This is a common topology for a mesh network, which can be mapped

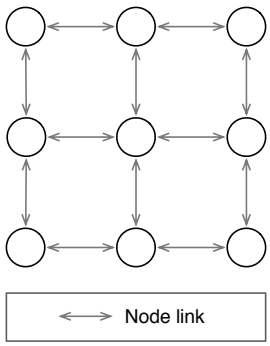


Figure 4.6: 9-node grid topology

to a number of hypothetical real-life deployments. We introduce 9 flows in the network with random source and destination nodes, and observe the behavior of those flows.

The MOS results per flow can be found in Table 4.5. The application of our MOS scheduler results in every flow experiencing an average increase of 1.17 MOS points, a very significant improvement which represents a jump of more than one position in a subjective quality scale.

Table 4.5: MOS results for a 9-flow, 9-node grid network

FlowID	**MOS (original)**	**MOS (w/ scheduler)**
(0)	2.05195	3.84851
(1)	1.99035	3.19960
(2)	1.27461	1.97175
(3)	1.25422	1.39373
(4)	1.29979	3.32081
(5)	2.55572	3.22935
(6)	3.46822	4.49741
(7)	1.34276	3.26908
(8)	2.52196	3.53586
Average	*1.973286*	*3.140677*

The results of a more thorough evaluation of this scenario can be seen in Figure 4.7. The simulation was repeated with an increasing number of flows in the network, going from 2 to 10 flows, and the average MOS for all flows was plotted with the MOS scheduler activated and with the standard *ns2mesh* scheduler only. The gains in user perception are significant. Once

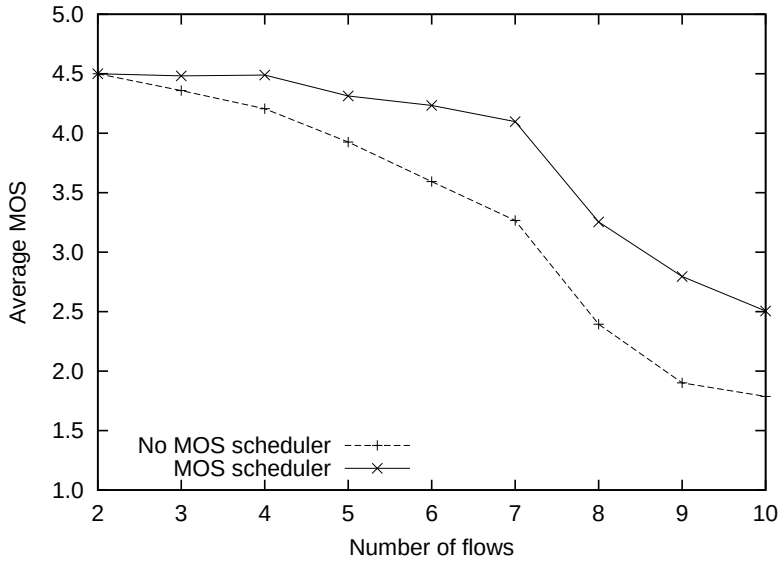


Figure 4.7: Average video MOS in a 9-node grid network for 'n' flows

the network load is increased, with more than three concurrent flows, the application of the MOS scheduler consistently returns improvements of more than 0.5 MOS points. Employing

this scheduler, the network can support up to 7 video flows with a MOS rating of more than 4 points, a very good quality value for the end-user, and with 10 flows the resulting MOS is of 2.5 points, which is about average in the MOS scale.

Video scenario, 5-node ring topology

The third and final simulation with only video flows in the network is of a 5-node ring topology, depicted in Figure 4.8, where we introduced 5 video streams with random source and destination nodes. The ring topology, with random flow distribution, has a higher probability of having links saturated with multiple flows, which is the motivation for testing this scenario.

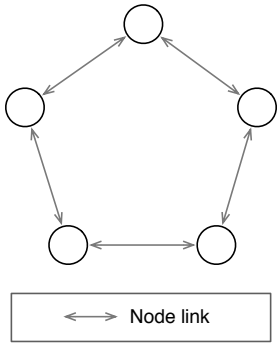


Figure 4.8: 5-node ring topology

Table 4.6 shows the MOS obtained by each flow with and without our scheduler. Although the MOS improvements are not as significant as the ones obtained in the previous simulations, the scheduler algorithm can still manage to obtain an average increase of 0.82 MOS points.

Table 4.6: MOS results for a 5-flow, 5-node ring network

FlowID	**MOS (original)**	**MOS (w/ scheduler)**
(0)	1.61090	2.32876
(1)	2.18730	2.92484
(2)	2.09462	2.90713
(3)	1.78883	2.61793
(4)	1.70875	2.71206
Average	*1.878080*	*2.698144*

Again, we performed a more thorough evaluation on this topology. We successively increased the number of video flows in the network, and calculated the average MOS of all flows both with the MOS scheduler active and inactive. The results can be found in Figure 4.9. The subjective quality measurements are similar to those obtained in Figure 4.7 of the previous scenario, but as was expected from this scenario, faster link saturation causes the video quality to decrease sharply once more than four flows are introduced in the network. Keeping the mean quality above 2.5 MOS points is not possible with more than 5 concurrent flows, even with recourse to the MOS scheduler.

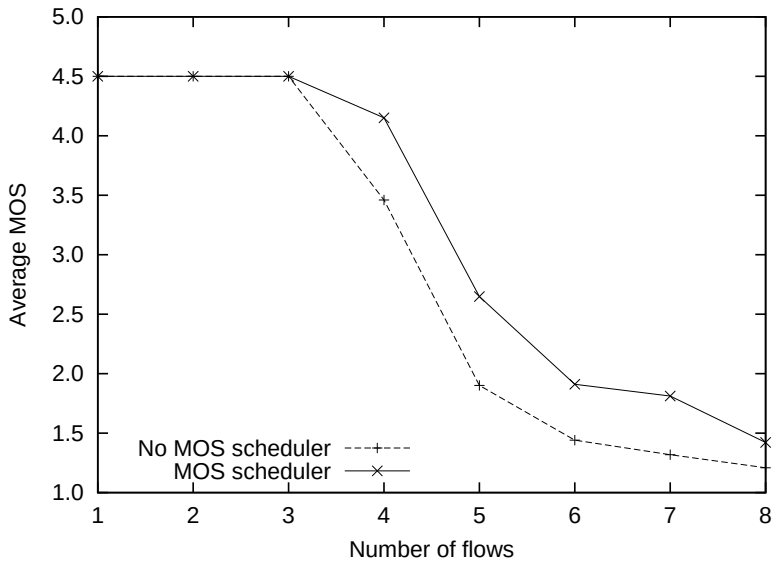


Figure 4.9: Average video MOS in a 5-node ring network for 'n' flows

4.4.4 Audio, video and file transfer services combined

We perform a series of simulations to determine the performance of the scheduler when three different types of traffic are present in the network: video, voice, and file transfer services. As mentioned earlier, voice and file transfer services are less prone to quality degradation due to random packet dropping, essentially due to the fact that the packets of these kinds of services are very similar to each other, and thus do not have large gaps between their importance to subjective quality. The goal here is for the scheduler to be able to balance all three services concurrently, interacting with packets from all services at the same time.

3-node chain topology

For the evaluation of the scheduler operation under these conditions, we first begin with a three-node chain network, as depicted in Figure 4.10. We introduce three VOD flows, three

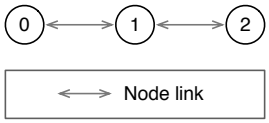


Figure 4.10: 3-node chain topology

VoIP flows, and two FTP flows, all sourced in node 0 and sinked in node 2. The results in average MOS points for each flow, for each service, and for all flows combined, can be seen in Table 4.7.

The MOS gains in this kind of simulation are much less evident than when only video flows are present. The small number and size of VoIP packets also makes it difficult for the scheduler to act on these kinds of flows: dropping a VoIP packet contributes very little for decreasing the buffer utilization, but can have the same impact to the flow's quality as

Table 4.7: MOS results of all services for a 3-node chain network

FlowID	**MOS (original)**	**MOS (w/ scheduler)**
(0)	2.03687	2.42027
(1)	2.13763	2.74232
(2)	2.02424	2.50250
VOD avg	*2.06624*	*2.55503*
(3)	4.33658	4.23935
(4)	4.33205	4.20991
(5)	4.32672	4.17102
VoIP avg	*4.33178*	*4.20676*
(6)	3.50575	3.65429
(7)	3.32311	3.61663
FTP avg	*3.41443*	*3.63546*
All Flows	***3.25286***	***3.44453***

of dropping a much larger packet from a video flow. Table 4.7 shows that the VOD flows experience a gain of about 0.5 MOS points, FTP flows also see a slight increase, and VoIP suffers a small decrease, in part due it being the service obtaining the highest MOS. Overall, the average service quality saw an improvement of 0.22 MOS points.

Audio, video and file transfer, 9-node grid topology

We now simulate traffic flows of the three service types in a 9-node network with a grid topology. Refer to Figure 4.6 given previously for a diagram of this topology. Flows are introduced in the network with random source and destination points, and the number of flows of each traffic type is successively increased to saturate the network links.

Figure 4.11 plots the average MOS of all flows with the same traffic type. The measurements with the MOS scheduler activated are plotted in a solid line, and those with the original *ns2mesh* scheduler are plotted in a dashed line. As expected, the gains are mostly seen on the VOD flows, with an average increase of 0.2 to 0.5 MOS points. The VoIP flows suffer a higher drop rate when network saturation is increased (6 flows per traffic type), due to being the flows with the higher MOS. FTP flows already have the lowest MOS in the *ns2mesh* simulator and thus are relatively unaffected. It can be seen that in the highest saturation scenario, the flows under the MOS scheduler are spaced much closer together than the ones without this scheduler.

Although the primary outcome of these simulations is the user satisfaction, reflected in the MOS values, we also plotted throughput and delay comparisons in Figures 4.12a and 4.12b, respectively. Figure 4.12a shows that throughput remains mostly unchanged despite the application of the MOS scheduler – it even increases slightly for both VOD and FTP services. In terms of packet delay, the use of the MOS scheduler can introduce a penalty of 50 to 100 ms for VOD and FTP flows, as seen in Figure 4.12b. Although the MOS models for VOD and FTP are delay-insensitive, this penalty must be taken into consideration for video flows, since it can cause problems if the receiver's buffer is small and frame delivery deadlines begin to expire.

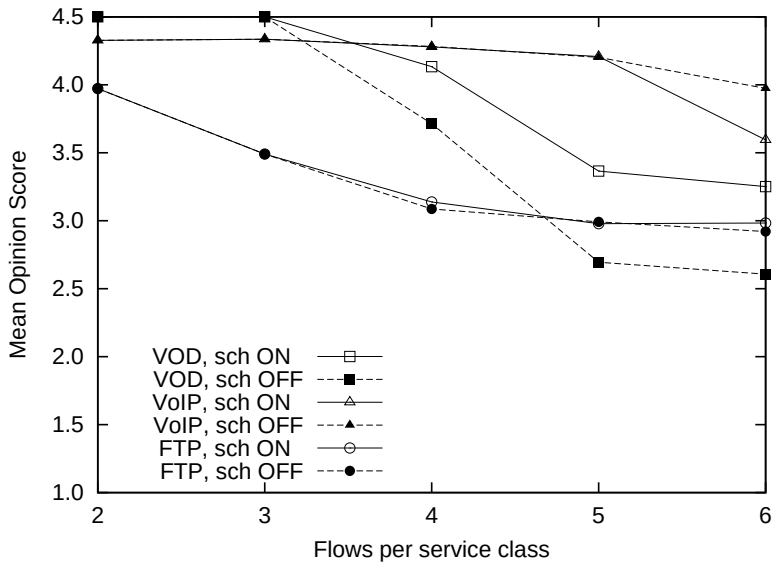
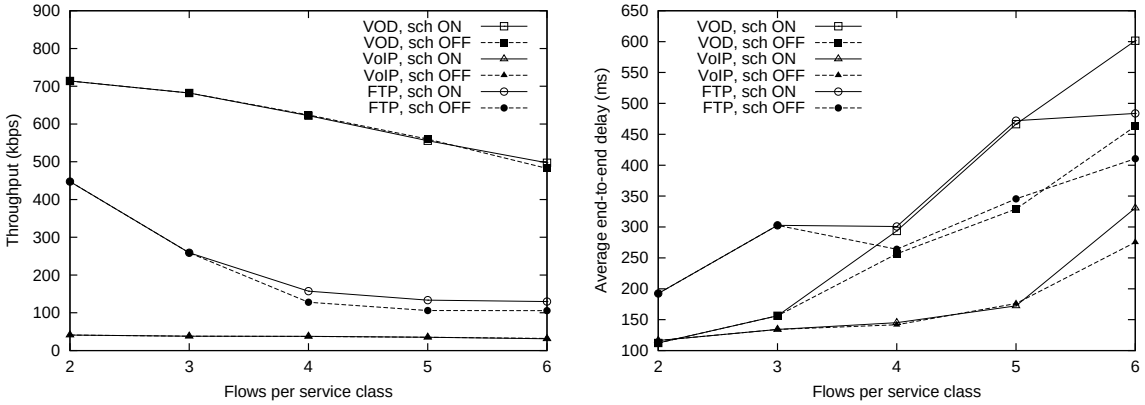


Figure 4.11: Average MOS in a 9-node grid network with 'n' flows per service



(a) Throughput

(b) One-way delay

Figure 4.12: Throughput and delay measurements in a 9-node mesh network with QoE

Audio, video and file transfer, 5-node ring topology

For the final simulation in this series, we repeat the previous scenario but now with a 5-node ring topology network. Again, refer to Figure 4.8 shown previously for a diagram of the topology. The results under this topology are plotted in Figure 4.13.

The results are similar to what was obtained in the 9-node grid network. Due to the faster link saturation caused by the ring topology, the overall MOS of all flows decreases more rapidly than in the previous scenario, and the scheduler activity is more evident, particularly in the VoIP flows. VoIP flows are penalized when compared to the original *ns2mesh* scheduler, so that the video service can be improved. We can again see how the three service types are closer to each other in terms of MOS when the MOS scheduler is activated.

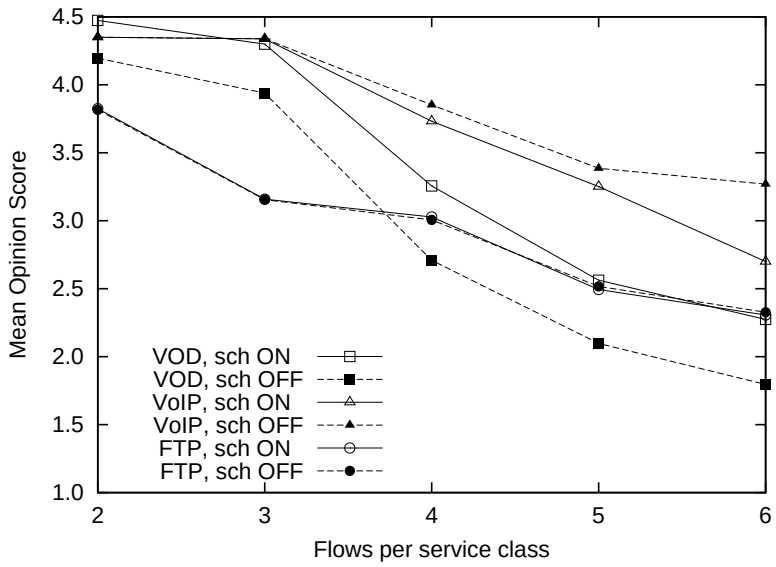


Figure 4.13: Average MOS in a 5-node ring network with 'n' flows per service

4.4.5 Computational effort

One issue that must be considered when implementing this scheduler is the computational effort required to run the scheduler's algorithms. The most resource-intensive processes in the implementation of our MOS scheduler are:

- Evaluating all packet combinations for their total size
- Evaluating each combination for their impact in MOS

The first process is mathematically less complex than the second – the scheduler only needs to sum the size of all packets in a combination and compare that value with a byte range. However, the number of combinations to be evaluated is a power of two of the number of packets in the buffer, which will frequently translate to many millions of combinations for evaluation. We found that considering any number in excess of 2^{20} (1.048.576) leads to simulations which are too intensive for the processing power of a modern personal computer; these simulations can take from 10x to 100x more time to complete than the simulation time (i.e., simulating 5 seconds of traffic can take up to 500 seconds to complete in real-time).

The second process is much more mathematically complex than the first, due to the need to apply the MOS calculation algorithms and the final cost function to determine the combination impact. The number of combinations evaluated here is smaller though, thanks to the preselection of combinations made by the first process. In our experiments, the number of combinations matched by the first process was for the most part inferior to 20.000.

In order to evaluate the consequence of decreasing the number of evaluated combinations, we repeat the 9-node grid scenario presented in 4.4.3, but now progressively decreasing the number of combinations evaluated for MOS (controlled by the *maxcombs* simulation parameter). We run the same scenario with *maxcombs* set to 4000, 2000, 1000, 600, 300, 100, and then with no limit. We also compare the results with the original *ns2mesh* scheduler. The

data is plotted in Figure 4.14; the 'Full search' and 'No scheduler' points are taken from Figure 4.7. Results show a predictable decrease in the MOS of all flows with the decrease

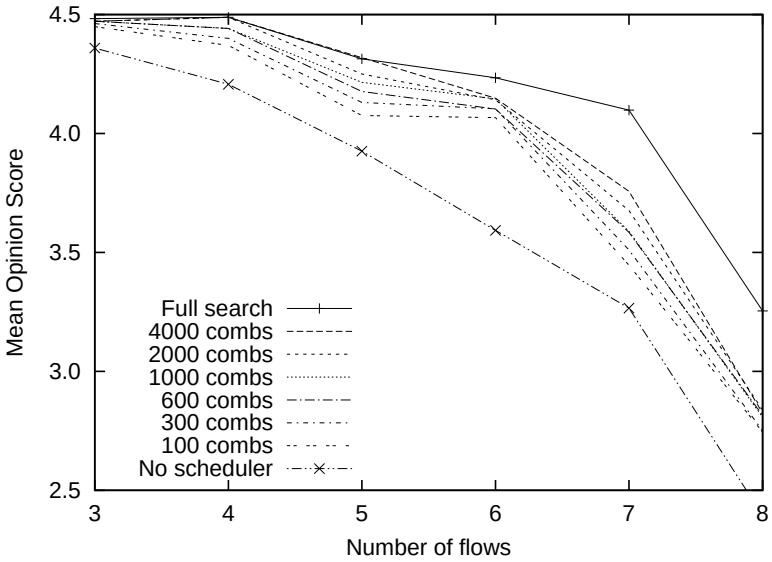


Figure 4.14: Average video MOS plots for varying limits on packet combinations

in evaluated combinations. We can observe that limiting *maxcombs* to 4000 and 2000 does not significantly impact the final quality when less than 6 flows are present in the network at the same time. Also, setting the limit on a small value such as 100 combinations is generally sufficient to obtain a gain of 0.2 to 0.5 MOS points, a good outcome for a tradeoff between quality and computational effort.

4.5 Integrating Quality of Service and Quality of Experience

For our final research topic, we integrated the QoS architecture presented in Chapter 3 with the MOS scheduler described in this chapter. Since both implementations were made on the same foundation, the *ns2mesh* IEEE 802.16 simulator, and since the implementations themselves do not overlap significantly, the development of a simulator which is both QoS- and QoE-aware is straightforward. The only modification required is regarding the number of buffers: in *ns2mesh*, the MOS scheduler acts on a single buffer, whereas on *ns2mesh* with QoS, the scheduler must fetch packets from four separate buffers, one for each service class.

We expect to be able to obtain the advantage of QoS service class differentiation, with the improvements in MOS resulting from the MOS scheduler algorithms. We performed a single set of simulations, in a 9-node grid network (previously depicted in Figure 4.6), with the same settings used for the simulations in Section 4.4. We assigned VOD traffic to the nrtPS service class, due to its variable bit-rate nature (which excludes UGS). VoIP traffic is assigned to the rtPS service class, due to its low bandwidth and low delay requirements. FTP traffic is placed in the Best Effort service class, which is only assigned bandwidth once nrtPS is served.

The MOS results are shown in Figure 4.15. The dashed lines represent results with the

scheduler deactivated, and the solid lines plot the results when the scheduler is active. The

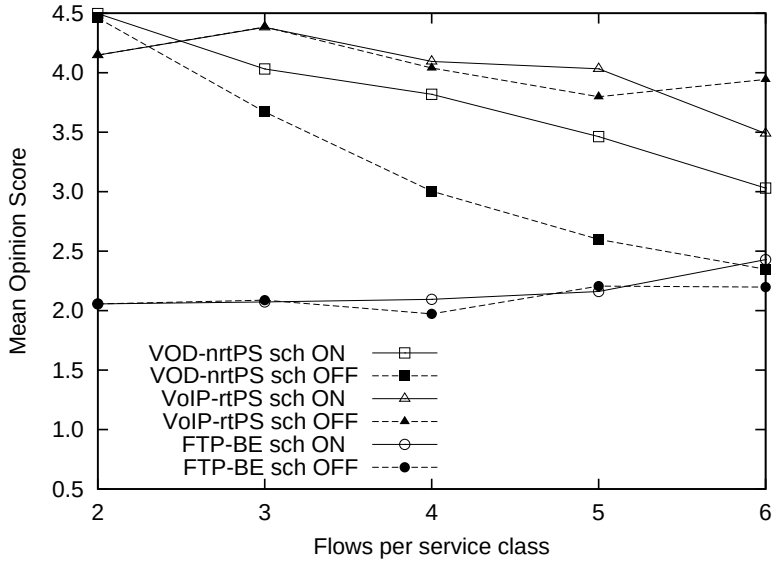


Figure 4.15: Average MOS in a 9-node grid network with QoS enabled

good functioning of the scheduler can be observed, with the VOD flows gaining an average of 1 MOS point when the network becomes saturated. Due to the placement of the FTP flows in a low-priority service class, MOS values for FTP are considerably low. This is a direct outcome of the application of the QoS architecture.

Figures 4.16 and 4.17 plot the throughput and packet end-to-end delays for this scenario, respectively. In Figure 4.16, we can see that the throughput behaves as expected from the

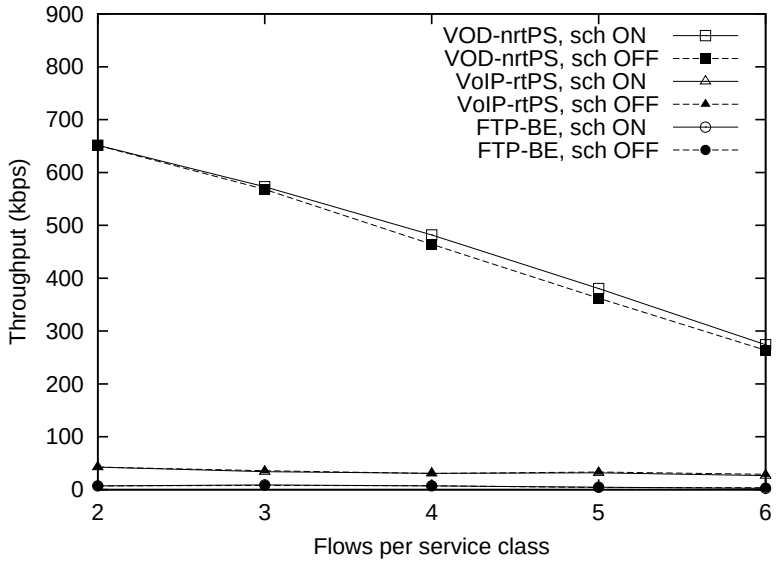


Figure 4.16: Throughput in a 9-node grid network with QoS enabled

service classes chosen for each type of traffic. BE traffic experiences a very low throughput,

due to network saturation caused by the VOD flows in the nrtPS service class. The VOD flows experience only a linear decrease in effective throughput, on par with the increase in number of flows.

In terms of packet delay, we can also see the outcome of QoS service differentiation in this scenario. Figure 4.17 shows that VoIP traffic in the rtPS class is consistently capable

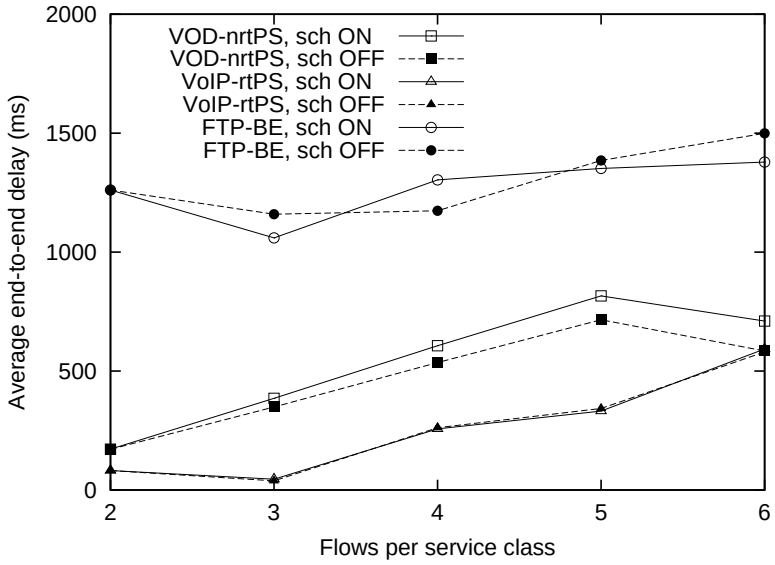


Figure 4.17: End-to-end delay in a 9-node grid network with QoS enabled

of the lowest delays in the network, while FTP packets experience the highest delay due to being marked as Best Effort. VOD traffic placed in the nrtPS class, which is meant for delay-insensitive applications, is subject to average delays when compared to FTP and VoIP.

4.6 Summary

In this chapter we have presented a new packet scheduler for wireless IEEE 802.16 mesh networks, which is designed to provide the best possible MOS of audio, video and file transfer services in the network. We have introduced methods to obtain the MOS of audio, video and file transfer data flows in real-time, a mathematical model for the scheduler implementation, a description of the parameters and variations the scheduler supports, and described the implementation of this scheduler in the *ns2mesh* simulator.

We have performed extensive simulations under a multitude of scenarios, in order to evaluate the performance of the scheduler. In the first set of simulations, we observed the performance in terms of flow MOS for three scenarios with only video traffic present in the network. We observed that the scheduler model is functional and returns positive MOS results in comparison to the plain *ns2mesh* packet scheduler. We evaluated 9-node grid networks and 5-node ring networks, where the flows were distributed randomly in the network, and were able to obtain average gains of about 1 MOS point with the application of this scheduler.

In the second set of simulations, we evaluated the performance of the scheduler with video, voice, and file transfer applications in mesh networks. The results again show a noticeable

increase in average subjective quality, in particular for video flows; they also show how the scheduler is capable of balancing the MOS of all traffic types in high-load situations. We also provided an evaluation on the computational effort required by the scheduler's algorithms, as well as a comparison of the effectiveness of the scheduler in simulations with limits on the number of processed combinations.

To finalize this research topic, we integrated the QoS architecture presented in Chapter 3 with the MOS scheduler of this chapter, and ran a simulation while observing MOS, throughput and packet delays. We were able to confirm the effectiveness of the MOS scheduler, while at the same time observing the differentiation in throughput and packet delay resulting from the usage of QoS service classes.

Chapter 5

Conclusion

The research and work presented in this thesis brings forth new insights and methods to handle Quality-of-Service and Quality-of-Experience in wireless IEEE 802.16 mesh networks. These new methods should prove useful in maximizing the utility of next-generation networks, when it comes to providing audio, video and file transfer services to the end user and maximizing the user's experience.

We approached the topic of QoS in mesh networks by researching and evaluating a new architecture designed to provide QoS support to IEEE 802.16's MESH mode of operation. We also did a thorough performance analysis of this architecture, and provided a number of improvements to increase its robustness and versatility. Through the simulation results we were able to confirm a number of advantages resulting from the architecture, which noticeably improves the ability of the network to deal with specific characteristics of certain types of traffic and provide guarantees in terms of throughput and delay requirements.

Regarding QoE, this work presented a new packet scheduler with the aim of maximizing the utility of audio, video and file transfer applications in a wireless mesh network. We fully described this scheduler's model, its variations and limitations, and its implementation in an IEEE 802.16 MESH mode simulator. Through extensive simulation work we were able to confirm the good functioning of the scheduler, which returned improved MOS performance in every scenario under evaluation. The computational effort of the algorithms was discussed, and we finalized with an integration of the QoS architecture and the MOS scheduler, complemented with an evaluation of the resulting performance in terms of subjective quality (MOS) and objective quality (throughput and delay). We also provided a considerable sum of information on the subject of subjective quality estimation, as well as practical algorithms and mechanisms to perform such measurements in real-time.

Research in this thesis shows that the next-generation networks, which are expected to deal with a great amount of multimedia services, can be improved to perform with more efficiency according to the requirements and behavior of its users, and that end user satisfaction can be significantly improved with the appropriate use of new methods focused on maximizing this user experience.

Future Work

The topics approached in this thesis are open to further research. The following topics would complement and improve the work performed:

- For the IEEE 802.16 QoS architecture:
  - Support for the extended-real-time Polling Service (ertPS) service class introduced in 802.16e.
  - Implementation of physical models in the *ns2mesh* simulator, for more realistic simulations.
  - Extensions for mobility support within the architecture.
  - Evaluation of multi-radio wireless mesh networks (MR-WMNs).
- For the MESH-mode MOS packet scheduler:
  - Implementation and evaluation of the proposed methods for delay awareness.
  - Research and evaluation of better mathematical models for the selection of packet combinations.
  - Native support for video traces of Scalable Video Coding (SVC) files.
  - Evaluation and improvements of the fairness of the scheduler decisions when dealing with multiple traffic types.
- For the integration between the QoS architecture and the MOS scheduler:
  - MOS scheduler improvements with awareness of QoS reservations and service classes.
  - Extensive sets of simulations and evaluation of performance.

Bibliography

[1] L. Nuaymi, "WiMAX: Technology for Broadband Wireless Access". West Sussex, England: John Wiley & Sons, 2007.

[2] IEEE 802.16 Working Group, "The IEEE 802.16 Working Group on Broadband Wireless Access Standards," May 20, 2009. [Online]. Available: http://wirelessman.org/.

[3] J.G. Andrews, A. Ghosh, and R. Muhamed, "Fundamentals of WiMAX: Understanding Broadband Wireless Networking". Upper Saddle River, N.J.: Prentice Hall, 2007.

[4] D. Beyer, N. Waes, and C. Eklund, "Tutorial: 802.16 MAC Layer Mesh Extensions Overview," IEEE 802.16, St Louis. Mar. 11, 2002. [Online]. Available: http://wirelessman.org/tga/contrib/S80216a-02_30.pdf.

[5] S.R. Saunders, A. Aragón-Zavala, "Antennas and Propagation for Wireless Communication Systems". Hoboken, N.J.: John Wiley & Sons, 2. ed., 2007.

[6] Y. Zhang, K.S Tan, P.Y. Kong, J. Zheng, and M. Fujise, "IEEE 802.16 WiMAX Mesh Networking," in *Wireless Mesh Networking: Architectures, Protocols and Standards*, Boca Raton, FL: Auerbach Publications, 2006.

[7] P.S. Mogre, M. Hollick, and R. Steinmetz, "The IEEE 802.16-2004 MeSH Mode Explained," Technische Universität Darmstadt, Tech. Rep. KOM-TR-2006-08, Dec. 4, 2006.

[8] C. Cicconetti, A. Erta, L. Lenzini, and E. Mingozzi, "Performance Evaluation of the Mesh Election Procedure of IEEE 802.16/WiMAX," in *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, 2007, pp. 323-327.

[9] IEEE Computer Society and the IEEE Microwave Theory and Techniques Society, "IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems," IEEE Std 802.16-2004 (Revision of IEEE Std 802.16-2001), 2004.

[10] M. Guizani, P. Lin, S.M. Cheng, D.W. Huang, and H.L. Fu, "Performance Evaluation for Minislot Allocation for Wireless Mesh Networks," in *IEEE Transactions on Vehicular Technology*, vol. 57, no. 6, Nov. 2008, pp. 3732-3745.

[11] M. Cao, W. Ma, Q. Zhang, and X. Wang, "Analysis of IEEE 802.16 Mesh Mode Scheduler Performance," in *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, April 2007, pp. 1455-1464.

[12] N.A. Abu Ali, A.-E.M. Taha, H.S. Hassanein, and H.T. Mouftah, "IEEE 802.16 Mesh Schedulers: Issues and Design Challenges," in *IEEE Network*, vol. 22, no. 1, Feb. 2008, pp. 58-65.

[13] M.H.M. Algamali, J. Wang, and M.Y. Abdullah, "OBSS: Optimal Base Station Scheduler of IEEE 802.16 Mesh mode," in *WiCOM '08: 4th International Conference on Wireless Communications, Networking and Mobile Computing*, Oct. 2008, pp. 1-4.

[14] J. El-Najjar, B. Jaumard, and C. Assi, "Maximizing Network Stability in a Mobile WiMax/802.16 Mesh Centralized Scheduling," in *WIMOB '08: IEEE International Conference on Wireless and Mobile Computing Networking and Communications*, Oct. 2008, pp. 259-265.

[15] S.Y. Wang, C.C. Lin, H.W. Chu, T.W. Hsu, and K.H. Fang, "Improving the Performances of Distributed Coordinated Scheduling in IEEE 802.16 Mesh Networks," in *IEEE Transactions on Vehicular Technology*, vol. 57, no. 4, July 2008, pp. 2531-2547.

[16] N. Bayer, B. Xu, V. Rakocevic, and J. Habermann, "Improving the Performance of the Distributed Scheduler in IEEE 802.16 Mesh Networks," in *VTC2007-Spring: IEEE 65th Vehicular Technology Conference*, Apr. 2007, pp. 1193-1197.

[17] C. Cicconetti, V. Gardellin, L. Lenzini, E. Mingozzi, and A. Erta, "End-to-End Bandwidth Reservation in IEEE 802.16 Mesh Networks," in *MASS 2007: IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems*, Oct. 2007, pp. 1-6.

[18] V. Sharma, A.A. Kumar, S.R. Sandeep, and M.S. Sankaran, "Providing QoS to Real and Data Applications in WiMAX Mesh Networks," in *WCNC 2008: IEEE Wireless Communications and Networking Conference*, Apr. 2008, pp. 2645-2650,

[19] P.S. Mogre, M. Hollick, R. Steinmetz, and C. Schwingenschlögl, "QoS Architecture for Efficient Bandwidth Management in the IEEE 802.16 Mesh Mode," in *WiMAX/MobileFi: Advanced Research and Technology*, 2006, pp. 197-216.

[20] International Telecommunication Union, Telecommunication Standardization Sector, "ITU-T Recommendation P.862: Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," Feb. 2001. [Online]. Available: http://www.itu.int/rec/T-REC-P.862-200102-I/en [Accessed: Apr. 20, 2009].

[21] International Telecommunication Union, Telecommunication Standardization Sector, "ITU-T Recommendation G.107: The E-model, a computational model for use in transmission planning," Feb. 2003. [Online]. Available: http://www.itu.int/itudoc/itu-t/aap/sg12aap/history/g107/g107.html [Accessed: Apr. 20, 2009].

[22] P. Seeling, M. Reisslein, and B. Kulapala, "Network Performance Evaluation Using Frame Size and Quality Traces of Single-Layer and Two-Layer Video: A Tutorial," in *IEEE Communications Surveys and Tutorials*, vol. 6, no. 3, 2004, pp. 58-78.

[23] G. Van der Auwera, P.T. David, and M. Reisslein, "Traffic Characteristics of H.264/AVC Variable Bit Rate Video," in *IEEE Communications Magazine*, Nov. 2008, pp. 164-174.

[24] G. Van der Auwera, P.T. David, and M. Reisslein, "Traffic and Quality Characterization of Single-Layer Video Streams Encoded with the H.264/MPEG-4 Advanced Video Coding Standard and Scalable Video Coding Extension," in *IEEE Transactions on Broadcasting*, vol. 54, no. 3, Sep. 2008, pp. 698-718.

[25] T. Oelbaum, K. Diepold and W. Zia, "A Generic Method To Increase The Prediction Accuracy Of Visual Quality Metrics," in *PCS2007: 26th Picture Coding Symposium*, Nov. 7-9, 2007.

[26] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," in *IEEE Transactions on Image Processing*, vol. 13, no. 4, Apr. 2004, pp. 600-612.

[27] S. Tao, J. Apostolopoulos, and R. Guerin, "Real-Time Monitoring of Video Quality in IP Networks," in *IEEE/ACM Transactions on Networking*, Oct. 2008, pp. 1052-1065.

[28] Information Sciences Institute, University of Southern California, "The Network Simulator - ns-2," Jan. 3, 2006. [Online]. Available: http://isi.edu/nsnam/ns/.

[29] Computer Networking Group, Department of Information Engineering, University of Pisa, "ns2mesh80216: 802.16 MESH mode patch for the Network Simulator 2 (ns-2)," Nov. 13, 2008. [Online]. Available: http://cng1.iet.unipi.it/wiki/index.php/Ns2mesh80216 [Accessed: Feb. 20, 2009].

[30] C. Cicconetti, I.F. Akyildiz, and L. Lenzini, "Bandwidth Balancing in Multi-Channel IEEE 802.16 Wireless Mesh Networks," in *INFOCOM 2007: 26th IEEE International Conference on Computer Communications*, May 6-12, 2007, pp. 2108-2116.

[31] C. Cicconetti, E. Mingozzi, and G. Stea, "An Integrated Framework for Enabling Effective Data Collection and Statistical Analysis with ns2," in *Workshop on ns-2: the IP network simulator (WNS2)*, Pisa, Italy, Oct. 10, 2006.

[32] Ira A. Fulton School of Engineering, Arizona State University, "Video Traces for Network Performance Evaluation," Oct. 27, 2008. [Online]. Available: http://trace.eas.asu.edu/tracemain.html [Accessed: Feb. 5, 2009].

[33] Video Quality Experts Group, "Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assessment," Aug. 25, 2003. [Online]. Available: http://www.its.bldrdoc.gov/vqeg/projects/frtv_phaseII/downloads/VQEGII_Final_Report.pdf.

[34] S. Sengupta, M. Chatterjee, and S. Ganguly, "Improving Quality of VoIP Streams over WiMax," in *IEEE Transactions on Computers*, vol. 57, no. 2, Feb. 2008, pp. 145-156.

[35] R.G. Cole and J.H. Rosenbluth, "Voice over IP performance monitoring," in *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 2, Apr. 2001, pp. 9-24.

[36] F.P. Kelly, "Charging and rate control for elastic traffic," in *European Transactions on Telecommunications*, vol. 8, Jan. 1997, pp. 33-37.

[37] W. Tu, J. Chakareski, and E. Steinbach, "Rate-Distortion Optimized Frame Dropping for Multiuser Streaming and Conversational Videos," in *Advances in Multimedia*, vol. 2008, id. 628970, 2008.

[38] S. Khan, S. Duhovnikov, E. Steinbach, and W. Kellerer, "MOS-Based Multiuser Multiapplication Cross-Layer Optimization for Mobile Multimedia Communication," in *Advances in Multimedia*, vol. 2007, id. 94918, 2007.