Network-Supported Congestion Control for Video Multicasting over IP Networks

by

Ashraf Matrawy

B.Sc., Alexandria University (1993)

M.Sc., Alexandria University (1997)

A thesis submitted to the

Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

December 20, 2002

©Ashraf Matrawy, 2002

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79460-1

Canada

The undersigned recommend to the Faculty of Graduate Studies
and Research acceptance of the thesis

**Network-Supported Congestion Control for Video Multicasting
over IP Networks**

submitted by

Ashraf Matrawy (B.Sc., M.Sc.)

in partial fulfillment of the requirements for

the degree Doctor of Philosophy

A handwritten signature in black ink, appearing to read 'Rafik Goubran', positioned above a horizontal line.

Professor Rafik Goubran, Chair, Department of Systems and Computer Engineering

A handwritten signature in black ink, appearing to read 'Ioannis Lambadaris', positioned above a horizontal line.

Professor Ioannis Lambadaris, Thesis Supervisor

A handwritten signature in black ink, appearing to read 'Sameh Rabie', positioned above a horizontal line.

Dr. Sameh Rabie, External Examiner

Carleton University

December 20, 2002

Abstract

In this thesis, we present our research work in the area of multicast congestion control for video applications. The core work of this thesis is the proposal of the use of a new form of network support to multicast congestion control. Our approach is to bring together simple, loosely-coupled, router mechanisms and adopt them for the multicast case. We use a variant of the Explicit Congestion Notification (ECN) mechanism to notify the sender of a multicast session of potential network congestion. We develop an end-to-end multicast system using this congestion control scheme. This included the development and tuning of an elaborate rate adaptation algorithm that operates at the sender. We build this system on top of a network that applies packet priority-dropping to insure providing minimum video quality during persistent congestion. In particular, the work is targeted at the IETF Assured Forwarding (AF) services networks. We show that the synergy of these mechanisms deals with the heterogeneity of receivers in a scalable manner and avoids the major problems of earlier approaches. We believe that this work is of great value to multicasting applications in future QoS-aware networks.

Acknowledgements

I thank GOD for giving us the bounty of knowledge and I pray that we can use it for sake of a better life for everyone.

I would like to thank my supervisor Prof. Ioannis Lambadaris for guiding me through the different phases of my Ph.D. He helped me with different aspects of my research including choosing the area, building and revising the research plan, introducing me to different sources of knowledge in academia and industry and for securing financial support throughout the entire study span. I gained a lot of knowledge from working with him and learned first hand how to conduct research in a professional manner.

I greatly appreciate the generous financial support I got from CITO (Communications and Information Technology Ontario), the OGSST (Ontario Graduate Scholarship for Science and Technology), and Carleton University.

Last but not least, I would also like to thank my family for being very supportive and for the many compromises they have made in order to help me go through the graduate studies experience.

To my parents: Aly and Tayseer

to whom I am very grateful

and of whom I am very proud

Contents

Acceptance Form	ii
Abstract	iii
Acknowledgements	iv
Table of Contents	vi
List of Figures	xii
List of Acronyms	xvi
1 Introduction	1
1.1 Multicast at different layers of the Internet Protocol	2
1.2 Research overview	3
1.3 Contributions of this thesis	8
1.4 Organization of this thesis	9

Part I: End-to-end IP Multicast	11
2 End-to-end Multicast Congestion Control	12
2.1 Architectural elements of IP multicast congestion control	13
2.1.1 Reliable multicast vs. real-time multicast	13
2.1.2 The regulating parameter	13
2.1.3 Location of the flow/congestion controller	14
2.1.4 The feedback mechanism	15
2.1.5 End-to-end vs. network-support	16
2.2 Evaluation of multicast video	16
2.3 Layered multicast	18
2.3.1 Receiver-driven Layered Multicast (RLM)	19
2.3.2 Layered Video Multicasting with Retransmissions	21
2.3.3 TCP-like congestion control for layered multicast	21
2.3.4 Packet-pair Layered Multicast	22
2.4 Single-rate multicast	23
2.4.1 Scalable Feedback Control	23
2.4.2 Destination Set Grouping	23
2.4.3 Pragmatic General Multicast Congestion Control	24
2.4.4 TCP-friendly Multicast Congestion Control	24
2.5 Conclusion	25

3	Performance of Layered Multicast Video	27
3.1	Problems of Layered Multicast	28
3.2	Fairness on the Internet	29
3.3	Simulations	31
3.4	Congestion Control of layered video	32
3.4.1	Experiment	33
3.4.2	Discussion	37
3.5	Rate Distribution for layered video	40
3.5.1	Linear increase of layered video rates	40
3.5.2	Exponential increase of layered video rates	42
3.5.3	Hyperbolic increase of layered video rates	43
3.6	Study with VBR Traffic	44
3.6.1	Linear increase of layered video rates	48
3.6.2	Exponential increase of layered video rates	48
3.6.3	Hyperbolic increase of layered video rates	48
3.7	Conclusion	49

Part II: Network-Supported IP Multicast 55

4	Evaluation of Router-Techniques to Support IP Multicast Congestion Control	56
4.1	Evaluation Criteria for Router Techniques	57

4.1.1	Complexity	57
4.1.2	Time to deploy	58
4.1.3	Comparison with end-to-end techniques	58
4.2	The Two Categories of Router Techniques	59
4.3	Single-rate with packet filtering	60
4.3.1	Filtering based on sequence numbers	60
4.3.2	Tag-based filtering	62
4.3.3	Efficient Congestion Avoidance Mechanism	62
4.4	Multi-rate (layered) flow control	63
4.4.1	Receiver-Selectable Loss Priorities	63
4.4.2	Network-supported Layered Multicast	65
4.4.3	Router-Assisted Layered Multicast	66
4.5	Conclusion	67
5	Modeling and Simulation of Real-time Video	68
5.1	MPEG4 Video Encoding	69
5.2	Transform Expand Sample (TES) models	71
5.3	Modeling MPEG4 using TES	74
5.4	Implementation and integration into the Network Simulator	75
6	Explicit Congestion Notification for IP Multicast	82
6.1	Overview	83

6.2	RIO (RED with In-Out bits)	84
6.3	Issues With The Use of ECN (Explicit Congestion Notification) . . .	85
6.4	BECN (Backward Explicit Congestion Notification)	87
6.4.1	System architecture	89
6.4.2	Simulation and performance	91
6.4.3	Results for BECN	92
6.5	MECN (Multi-level Explicit Congestion Notification)	96
6.5.1	Difference from BECN	96
6.5.2	Results for MECN	97
6.6	Conclusion	101
7	Multicast Congestion Control for Assured Forwarding	102
7.1	Queue management for Assured Forwarding	104
7.2	End-to-end architecture	108
7.3	The rate adaptation algorithm	111
7.3.1	The rate adaptation equation	111
7.3.2	Round-Trip Time (RTT)	116
7.3.3	Feedback suppression	116
7.3.4	Calculation of probabilities	117
7.3.5	The algorithm to change R_i	118
7.3.6	Changing the equation parameters	119
7.4	Simulation setup	120

7.5	Discussion of simulation results	120
7.5.1	Basic test	120
7.5.2	Heterogeneity test	122
7.5.3	Test of bandwidth adaptation	125
7.5.4	Scalability test	132
7.5.5	Multisession interaction	134
7.5.6	Testing with other topologies	136
7.5.7	Interaction with TCP	138
7.6	Design issues	142
7.7	Conclusion	147
8	Conclusions and Future Work	148
A	Derivation of The Rate Adaptation Equation for Unicast	150
	Bibliography	153

List of Figures

1.1	Multicast at different layers	4
3.1	Simulation setup	31
3.2	RLM starts first, R1 with $w=12$	36
3.3	RLM starts first, R2 with $w=12$	37
3.4	RLM starts first, R1 with $w=4$	38
3.5	RLM starts first, R2 with $w=4$	39
3.6	CBR linear R1	41
3.7	CBR linear R2	42
3.8	CBR Exponential R1	44
3.9	CBR Exponential R2	45
3.10	CBR Hyperbolic R1	46
3.11	CBR Hyperbolic R2	47
3.12	VBR linear R1	49
3.13	VBR linear R2	50
3.14	VBR Exponential R1	51

3.15	VBR Exponential R2	52
3.16	VBR Hyperbolic R1	53
3.17	VBR Hyperbolic R2	54
4.1	Single-rate with filtering	59
4.2	Multi-rate with router flow control	64
5.1	Frame transmission and display orders	71
5.2	Characteristics of the original trace	77
5.3	The I frames model	78
5.4	The P frames model	79
5.5	The B frames model	80
5.6	Comparison between the original and the generated traces	81
6.1	Protocol stack at the sender	88
6.2	Flow control algorithm at the sender	89
6.3	Simulation setup	93
6.4	Throughput for Receiver R3 using BECN	93
6.5	Performance of Receiver R1 using BECN	94
6.6	Performance of Receiver R2 using BECN	95
6.7	Performance of Receiver R3 using MECN	97
6.8	Performance of Receiver R1 using MECN	98
6.9	Performance of Receiver R2 using MECN	99

7.1	Staggered parameter configuration of a two-priority RED queue . . .	106
7.2	End-to-end architecture	110
7.3	Selection of the value of α_i	115
7.4	Output of the MPEG4 generator with no control at the source	121
7.5	Basic test topology	121
7.6	Result of the basic test with bandwidth B	122
7.7	Heterogeneity test topology	122
7.8	High priority throughput	126
7.9	Low priority throughput	127
7.10	Total throughput	128
7.11	Packet loss in high priority	129
7.12	Packet loss in low priority	130
7.13	Rate of feedback messages	131
7.14	Topology of the bandwidth adaptation test	132
7.15	Bandwidth adaptation to changing session membership	133
7.16	Scalability test topology	134
7.17	Scalability test	135
7.18	Topology for testing multisession interaction	136
7.19	Multisession interaction	137
7.20	Arbitrary topology 1	138
7.21	Results for receivers in arbitrary topology 1	139

7.22 Arbitrary topology 2	140
7.23 Results for receivers in arbitrary topology 2	141
7.24 Interaction with TCP - topology 1	142
7.25 Results for interaction with TCP - topology 1	143
7.26 Interaction with TCP - topology 2	144
7.27 Results for Interaction with TCP - topology 2	145
A.1 Illustration of the rate equation parameters	151

List of Acronyms

AF	Assured Forwarding
BECN	Backward Explicit Congestion Notification
CBR	Constant Bit Rate
<i>diffserv</i>	Differentiated Services
DSCP	Differentiated Services Codepoint
DSG	Destination Set Grouping
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
FEC	Forward Error Correction
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IETF	Internet Engineering Task Force
LVMR	Layered Video Multicast with Retransmissions
MECN	Multi-level Explicit Congestion Notification
MPEG	Moving Picture Expert Group

<i>ns-2</i>	Network Simulator version 2
NLM	Network-supported Layered Multicast
PGMCC	Pragmatic General Multicast Congestion Control
PHB	Per Hop Behavior
PLM	Packet-pair Layered Multicast
QoS	Quality of Service
RED	Random Early Detection
RIO	RED with in and out
RIO-C	RED with in and out in the Coupled mode
RIO-D	RED with in and out in the De-coupled mode
RALM	Router-Assisted Layered Multicast
RLM	Receiver-driven Layered Multicast
SFC	Scalable Feedback Control
SLA	Service Level Agreement
SQ	Source Quench
TES	Transform Expand Sample
TCP	Transport Control Protocol
TFMCC	TCP-Friendly Multicast Congestion Control
UDP	User Datagram Protocol
VBR	Variable Bit Rate
WRED	Weighted Random Early Detection

Chapter 1

Introduction

One of the keys to the success of the Internet is the congestion control mechanism of TCP [61] (Transport Control Protocol). This mechanism ensures that service on the Internet does not degrade dramatically during congestion times. However, TCP serves as a solution only for unicast applications. For multicast applications, a counterpart of TCP should exist (in addition to other requirements) before multicast services can be widely deployed on the Internet. This has proven to be a far more difficult problem than the unicast case. Multicast congestion control schemes should *scale* to a large number of receivers. In the same time they should deal with receivers' heterogeneity of networking capabilities as well as their heterogeneity of QoS requirements.

1.1 Multicast at different layers of the Internet Protocol

Multicast is the form of communication where information is to be delivered from one (or more) sources to multiple receivers. This is a form that lies in between unicast, which is one-to-one, and broadcast mode which is one-to-all. Multicast is *one-to-some* communication.

The term *multicast* may refer to one of three levels of multicast:

- *Application-layer multicast*: In this case, all multicast support is provided at the application layer. By running distributed algorithms, receivers of a multicast session organize themselves in an overlay network. All communications are carried out then using unicast between adjacent nodes of this overlay network. This has the advantages of the possibility of immediate deployment since it can utilize all the flow/congestion control capabilities available for unicast. The disadvantages are higher delays and inefficient use of bandwidth. Some packets may traverse the same link back and forth as illustrated in part (a) of Figure 1.1. This is also referred to as overlay or end-systems multicast. For more information check [9, 11] and references within.
- *Network-layer (IP) multicast*: Multicast routers are needed in this case. These routers are responsible for building and managing multicast distribution trees. At the routers, multicast packets are replicated and forwarded down to the mul-

multicast tree leaves. Multicast routers that are on a leaf network, discover and communicate local receivers of a multicast session. The Internet Group Management Protocol (IGMP) [18] is used for this purpose. IP multicast is deployed in the experimental virtual network, the Multicast Backbone (MBONE) [17, 5]. It is implemented using tunneling techniques between multicast routers on the Internet. These tunnels constitute a virtual network on top of the Internet. An illustration of IP multicast is shown in part (b) of Figure 1.1.

- *Physical-layer (local-area) multicast*: This type of multicast utilizes the broadcast capabilities of the underlying physical layer [59]. An example of this is Ethernet. It is carried out by broadcasting the information to all receivers on the physical media. Receivers filter the traffic to get the information destined to them. This serves the purpose of distributing information locally in a LAN environment. There can also be a mapping between either application-layer multicast or IP multicast and physical layer multicast.

In this thesis, we focus on IP multicast and in the rest of this thesis when we refer to multicast we mean IP multicast.

1.2 Research overview

A large number of group communication applications has emerged recently on the Internet [23]. These applications have been the driving force behind the efforts of

Figure 1.1: Multicast at different layers

research and development of the IP Multicast technology. IP Multicast enables multipoint communications over IP networks while preserving bandwidth in a manner that can be extremely significant in cases of large groups. However, deployment of

IP multicast services over the Internet has not been as rapid as needed [16]. One of the main factors that contribute to this slow deployment is the lack of a multicast congestion control scheme that can be as robust as its unicast counterparts (e.g. TCP congestion control for reliable unicast). The two major challenges in the area of multicast congestion control are:

1. The heterogeneity of receivers' networking capabilities as well as the heterogeneity of their QoS-requirements.
2. Maintaining the scalability of the multicast congestion control technique is a difficult task as the number of receivers is unknown to the sender and may grow significantly.

Congestion control of multicast real-time video over IP networks is a special case. The strict requirements of real-time video applications impose more challenges on the already-complicated IP multicasting. Another property of video applications that is of interest to congestion control schemes designers is that these applications can - to some extent - tolerate loss. This property introduces designers with some flexibility that they should approach with caution not to develop a design that is greedy and aggressive in terms of bandwidth utilization. This will be discussed in detail in Chapter 3.

Multicast congestion control techniques in general can be classified into two categories: the sender-based (single-rate) techniques and the receiver-based (multi-rate,

layered) techniques. Techniques that belong to both categories can be pure end-to-end or can use some network support. Chapter 2 presents a detailed discussion of end-to-end techniques and outlines their problems. An evaluation of network-supported multicast is presented in Chapter 4.

The problems of pure end-to-end techniques motivated a number of researchers to adopt network support in their work on multicast congestion control [13, 38, 57, 77]. Our goal in this thesis is to develop a multicast congestion control scheme that relies on network support in the single-source case. We investigated different options for network support and came to the conclusion that our solution should be based on bringing together simple, loosely-coupled, already-existing (or proposed), router mechanisms to construct a sophisticated multicast congestion control scheme. The router mechanisms we assume for our solution are:

- Supporting priority dropping of packets in time of congestion.
- Providing “Congestion Notification” to the multicast sender.

We chose the IETF [2] proposed AF (Assured Forwarding) services [28] architecture for the first requirement coupled with a variant of ECN [19] for the second requirement. We considered AF because it helps us build a simple end-to-end architecture that avoids the problems of earlier approaches to multicast congestion control (Refer to Chapter 2). AF services are also expected to be deployed soon in Internet routers as opposed to other proposed support mechanisms [13, 38, 57, 77] that require

major changes in current router's functionality as we will describe in Chapter 4.

Building on these mechanisms, we develop an end-to-end multicast system. This included the development and tuning of a rate adaptation algorithm at the sender. We show that combining these mechanisms deals with the heterogeneity of receivers in a scalable manner. We also show that our system avoids the major problems with earlier approaches. In particular, our work has the following advantages:

- It is sender-based with feedback support from the routers. Because the feedback is from the routers it is more scalable than earlier sender-based systems.
- We send one video stream, as opposed to multi-rate techniques, to avoid the overhead of handling multiple layers at the receivers. Check the problems of layered multicast in Chapter 2.
- Building our proposal on a network that supports priority dropping results in:
 - Provide a minimum quality even if congestion persists.
 - Dealing with heterogeneity by matching the rate of important part of the video to the slow receivers and give it higher priority. Sending the rest of the encoded video signal as lower priority enables receivers with higher bandwidth to get. This extra (lower priority) part will be dropped by slow receivers who only receive higher priority video.

1.3 Contributions of this thesis

The main contribution of this thesis is the exploration of network-supported IP multicast. Next generation networks possess a considerable amount of intelligence. Investing this intelligence in favor of multicast congestion control is worth investigation and this is mainly what we target in this research.

We summarize the contribution of this research in the following points:

1. Studying the performance of the layered approach to multicast congestion control for video applications. We identify and demonstrate the different problems that the layered multicast protocols encounter (Chapter 3).
2. Classification and evaluation of router techniques that support congestion control in IP multicast applications (Chapter 4).
3. Modeling and software realization of a traffic generator for MPEG4 encoded video using the Transform Expand Sample (TES) methodology. The generator helps obtain more realistic simulation results as opposed to using other approximate input models such as assuming CBR or general VBR input sources (Chapter 5).
4. Design of the novel architecture of network-supported congestion control for IP multicast using different ECN techniques. We compare the performance of different ECN techniques like BECN [66] and MECN [66] for use with IP

Multicast. We conclude that using a variation of MECN is the best option for use with IP multicast (Chapter 6).

5. Design of an elaborate rate adaptation algorithm that runs at the video multicast sender to allow the sender to deal with heterogeneous receivers (Chapter 7).
6. Design and performance study of our architecture and the rate adaptation algorithm in the context of the IETF Assured Forwarding service networks (Chapter 7).

We published the research work presented in this thesis in several international conferences and journals [41]-[50].

1.4 Organization of this thesis

This thesis is organized in two major parts. This is intended to logically separate the research presented in this thesis. This also represents the chronological order of the work done in this thesis and how we moved from one point to the next in our research plan.

Part I is where we started investigating the area of **end-to-end** multicast congestion control for video applications. We started by a survey of the background information about the most commonly cited approach, the end-to-end layered approach, presented in Chapter 2. This chapter outlines the challenges in this area of research. In Chapter 3, we present our study on the performance of the layered

multicast approach. Through simulations, we show problems that are inherent in this approach. Based on both the background survey (Chapter 2) and the performance study (Chapter 3), we came to the conclusion that we need to investigate the use of network-support in multicast congestion control.

Part II includes our research in the area of **network-supported** multicast congestion control. We started by classifying and evaluating some proposals of router techniques that support multicast congestion control and show which direction we would pursue in the area. This work is presented in Chapter 4. Since most of the evaluation of current networking protocols and architectures relies on simulation, we devoted some time to model and develop a reliable simulation model for real-time multicast in IP networks. Chapter 5 presents our work on building the multicast MPEG4 simulation environment. This chapter includes our modeling and development of the MPEG4 traffic generator using the TES (Transform Expand Sample) methodology and its integration into the network simulator *ns-2* [52]. Chapter 6 presents the end-to-end architecture that is based on priority-based networking and ECN. In the same chapter, a comparison between the use of different ECN techniques with IP multicast is presented. Chapter 7 expands the work of Chapter 6 by focusing on the IEFT AF services model. It presents a detailed discussion of the rate adaptation algorithm that enables senders to deal with heterogeneous receivers. It also presents some design issues and discusses the architecture limitations. Chapter 8 concludes the thesis and presents suggestions for future research in this area.

Part I

End-to-end IP Multicast

Part I presents our investigation of the area of end-to-end multicast congestion control for video applications. This part includes a survey of background information on end-to-end multicast congestion control in Chapter 2 followed by a study of the performance of layered multicast video in Chapter 3. The conclusions we obtained from this part motivated our research on network-supported multicast which we present in Part II.

Chapter 2

End-to-end Multicast Congestion Control

In this chapter, we survey and discuss pure end-to-end techniques for multicast congestion control. We start by a discussion of the different elements of a multicast congestion control scheme. A congestion control scheme for multicast video imposes specific requirements for these elements. The requirements are discussed in this chapter along with evaluation criteria for the performance of multicast video. We present several examples from the literature of both single-rate multicast applications and layered (multi-rate) multicast applications.

2.1 Architectural elements of IP multicast congestion control

These elements/issues are the components and the choices that constitute the multicast congestion control architecture. Based on the application, a choice for each of these elements is made. Each element does not necessarily exist in each application but we present them for the sake of completeness. Detailed studies of these elements can be found in [6, 58, 74].

2.1.1 Reliable multicast vs. real-time multicast

Reliable multicast is used by applications such as the distribution of software updates where accurate delivery of data is desired and some delay or low throughput can be accepted. The other case is that of real-time multicast for applications such as video conferencing where some errors can be tolerated for the sake of high throughput and low delay. This difference between these two modes of multicast congestion control results in selecting different sets of the following architectural elements.

2.1.2 The regulating parameter

Congestion control schemes in general can be categorized in two groups based on the regulating parameter. The first group is those schemes that use a TCP-style congestion window at the sender or at the receiver(s) to control the load they provide

to the network. Each transmitted packet decreases the number of available spots in the window by one and each acknowledged packet increases that number by one. If at any time, the number of available spots is zero, transmission is halted until at least one packet is acknowledged as received. The size of the congestion window (number of available spots) is increased if there is no congestion and is decreased with the detection of congestion. Window-based congestion control proved to be far more difficult in the multicast case than in the case of unicast [74].

The second group of schemes is those schemes that use the source's rate as the regulating parameter. The idea is to keep that rate below a level that is believed to be suitable for the network. This rate is calculated using either a model or a simple increase/decrease algorithm. Except for the layered multicast approach, rate-based congestion control depends on feedback from receivers. In the model-based case, the feedback messages contain measurements for some parameters that are used in model calculations. In the simple increase/decrease case, the feedback acknowledges whether there is congestion in the network or not. The most general algorithm for increase/decrease is the AIMD [10] (Additive Increase/Multiplicative Decrease). Rate-based is more common than window-based in multicast applications.

2.1.3 Location of the flow/congestion controller

In the literature, multicast congestion control techniques can be classified into two categories: the sender-based (single-rate) techniques and the receiver-based (multi-

rate, layered) techniques. The location of the flow/congestion controller can be either at the sender or at the receiver or can be shared between them in some architectures. Both categories apply to both the reliable and the real-time cases and in both cases the regulating parameter can be window-based or the rate-based.

In the sender-based techniques, the sender performs the flow/congestion control functionality using a single stream to all receivers that receive the same rate. The other case is that of the receiver-driven where the source data is sent in layers and receivers are then allowed to receive as many layers as they can. There is a scheme that combines the receiver-based approach with feedback to the source where the source data is replicated and sent in different streams with different rates (qualities). Receivers get only one of those replicated streams according to its available bandwidth.

2.1.4 The feedback mechanism

Some congestion control schemes require feedback. All reliable multicast applications must have a feedback mechanism to insure the correct delivery of data. Some real-time schemes use feedback as well. Feedback can be collected using a hierarchy of the receivers that ends at the sender. Another method is to use broadcasting to share receivers reports.

Current multicast congestion schemes that require feedback rely on reports from receivers. In this thesis, we propose the alternative of relying on network Explicit

Congestion Notification to provide feedback to the sender.

2.1.5 End-to-end vs. network-support

The majority of multicast architectures in the literature are pure end-to-end. They require the collaboration of the sender and the receivers. This has the advantage of immediate deployment in best-effort Internet. Disadvantages are: the slow response time to congestion due to the end-to-end delay and fairness problems caused by aggressive protocols [20]. Some network-support to congestion control is needed in network routers [74]. Providing this support for multicast applications is the main contribution of this thesis.

2.2 Evaluation of multicast video

In this section, we define the criteria we use to evaluate different video multicast protocols. We summarize these criteria in the following paragraphs.

- **Dealing with heterogeneity:** This is the first problem that needs to be solved for any multicast application. That is developing a methodology for one (or more) sender(s) to be able to communicate with several receivers and satisfying the receivers requirements simultaneously. A multicast congestion control mechanism that does not solve this problem efficiently is obviously not an acceptable one. For multicast video, solutions for this problem can take ad-

vantage of the fact that not all receivers need to receive the same information. In other words, receivers can get different levels of quality of the visual information. This translates into different rates delivered to different receivers. In the reliable multicast case, all receivers need to get the exact same information because of the nature of the delivered data.

- **Scalability:** The second important issue is scalability of the congestion control scheme. A multicast congestion control scheme should be able to deal with the heterogeneity of receivers in a *scalable* manner. In the case of multicast video, the number of receivers is unknown to the sender and may grow significantly. In general, feedback mechanisms in multicast applications are the main source of scalability problems.
- **Fairness:** Fairness in bandwidth sharing between competing protocols on the Internet is a serious issue and a very essential one for the robustness of the Internet. In particular, new protocols should be tested for fairness to TCP (TCP-Friendliness) before they are deployed on the Internet. TCP's congestion control results in a fair share of bandwidth between competing TCP flows. It is based on reducing the sender's offered load when congestion is detected in the network.

For multicast video, there are a few issues with fairness. First, most video applications are based on UDP (User Datagram Protocol) [62] which is known

to be unfair to TCP because UDP simply does not have a congestion control mechanism. Second, video applications can tolerate loss to some extent as opposed to data distribution applications. This may result in video applications that are not responsive enough simply because they can tolerate packet loss. Finally, the scope of some multicast sessions, in terms of the number of receivers, magnifies the effects of unfair behavior by multicast video.

- **Other criteria:** In addition to these criteria, there are other criteria that are important as well. These are the utilization of the available bandwidth at the receivers, stability of the received video, and time to converge to this stable level of quality at the receiver.

2.3 Layered multicast

Layered multicast is based on the ability to generate the source data in a layered format and sending the layers as different multicast groups [67]. Each layer contains a subset of the information being sent. Receivers decide on how many layers (or equivalently, multicast groups) they can join using some bandwidth inference technique. Layers should be joined in a *cumulative* manner which means joining them in order of their relevance. Basic layers will contain minimum information necessary to get basic quality and they should be joined first.

In the following sections we present a brief description of some of the proposed

layered multicast schemes. This is not intended to be an exhaustive survey but we present the schemes that are most commonly cited in the literature and that represent the main methodologies for this group of schemes.

2.3.1 Receiver-driven Layered Multicast (RLM)

In this section, we overview the RLM protocol which is one of the earliest proposals in this area and is the most commonly cited layered multicast scheme. The features we discuss here are relevant to our work presented in Chapter 3. Receiver-driven Layered Multicast (RLM) [51, 53] is a transport protocol that allows receivers to adapt the quality of the video they receive according to their available bandwidth. In RLM, layered video coding is used where the video signal is encoded into a number of layers. The lowest layer contains the basic information, and each subsequent layer provides progressive enhancement. The sender sends each video layer to a separate IP multicast group and takes no active role in rate adaptation. Each receiver subscribes to a certain set of video layers by joining the corresponding IP multicast group. The advantage of receiver-based control over sender-based control is that the burden of adaptation is moved from the sender to the receivers, resulting in enhanced system scalability.

Each receiver tries to achieve the optimal level of subscription of video layers. Optimal level of subscription is the maximum level a user can get within the *available* link capacity. The basic adaptive control works as follows: when a receiver detects

congestion, it drops a layer, and when there is spare bandwidth available, it adds a layer. Deciding whether the current reception level is optimal is a crucial step in the control loop. If congestion is detected, this means that the current subscription level is too high. In RLM, congestion is detected with packet loss rate. On the other hand, there is no explicit feedback on whether the current subscription level is too low, so in RLM, join-experiments are carried out to find if the receiver is capable of handling the next video layer. If congestion is detected after the experiment, the receiver drops the newly added layer. If congestion is not detected the experiment is successful. A failed join-experiment can bring congestion to the network, resulting in degraded video quality to both the receiver that initiated the experiment and possibly other receivers that share the congested link. Therefore, a learning algorithm is proposed so as to minimize the frequency and duration of join-experiments without impacting the video layer convergence rate. To decrease the frequency of failed join-experiments, a *join-timer* is set for each video layer and *exponential backoff* is applied on the layers that would possibly bring congestion. If a receiver has been stable for some time, it becomes much slower in dropping the current layer. When it detects a loss, it waits for some time and then it starts to measure the loss in a window and if within this window the loss is greater than a threshold, it drops the highest layer. This behavior plays an important role in RLM's interaction with TCP. A detailed discussion of the protocol receiver state machine can be found in [51]. The performance of RLM is studied in [24, 34, 47] with a modification proposed in [76].

2.3.2 Layered Video Multicasting with Retransmissions

LVMR [36, 37] is a system for distributing MPEG-encoded video over a best-effort heterogeneous network. It deals with heterogeneity using layered encoding and layered transmission in the same fashion of RLM. It has two major contributions to the area of layered multicast. First, it adjusts the video reception rate at receivers using a hierarchy of agents in the network that control the rate. These agents help receivers decide to join/drop a layer. This decreases the volume of control information than the case of RLM where receivers used to broadcast their subscription information within the group. Second, it introduces the concept of recovery using retransmission from designated local receivers to reduce recovery time. While the results in [36, 37] show improvement over RLM, they introduce a considerable overhead specially in the case of real-time video. For this system to be widely used, the agent hierarchy must be deployed and tested on the Internet. To the best of our knowledge, there is no reported study of LVMR fairness on the Internet.

2.3.3 TCP-like congestion control for layered multicast

In an attempt to address the fairness issues of RLM, the authors of [73] proposed a TCP-like congestion control scheme for multicast applications. It is based on the generation of periodic bursts that are used for bandwidth inference and synchronization points (SP) that indicate to the receivers when they can join a layer. The TCP-friendly behavior results from the significant amount of rate decrease in case

of congestion detection in a TCP fashion. The results of the analytical analysis and simulations in [73] show that the protocol achieves a fair sharing of bottleneck bandwidth with other protocol instances and with TCP flows. However, Legout et al. authored a paper [34] on the “pathological” behaviors of both RLM and this TCP-friendly version. Through simulations [34], the authors show that the two protocols (RLM and its TCP-friendly version) are responsible for generating transient periods of congestion, instability, and periodic losses. They also show that these problems are inherent in the protocols and are difficult to correct.

2.3.4 Packet-pair Layered Multicast

After presenting their critique in [34], Legout et al. presented PLM [35] where they tried to avoid all the problems they pointed out in other layered protocols. PLM is based on the generation of packet pairs (PP) to infer the available bandwidth. In their scheme, source packets are generated and sent in pairs and by measuring the delay between the two packets in subsequent pairs, receivers can infer network congestion status. The authors of PLM claim that using PP’s does not have any of the weaknesses of other layered multicast protocols. However, PLM is based on the requirement that all routers in the network implement their Fair Queueing paradigm [33] that allows to theoretically devise nearly ideal congestion control protocols [34].

2.4 Single-rate multicast

In sender-based techniques, a single rate is sent to all receivers and feedback is collected from receivers to change the sending rate.

2.4.1 Scalable Feedback Control

SFC [8, 72] is one of the earliest works in the area of sender-based techniques. A video conferencing tool called IVS (Internet Videoconferencing System) [72] was built using this scheme. It used feedback messages from receivers with information on packet loss to estimate the “group” reception status. Scalability is an obvious problem with this approach because receiving feedback from all receivers simply overwhelms the network. A proposal to use representatives of receivers groups was introduced in [15] and presented mechanisms to select representatives. However, changing representatives is a major overhead for this approach.

2.4.2 Destination Set Grouping

DSG [65] was proposed as a compromise between sender-based and receiver-based systems. In DSG, the sender replicates the source data in a small number of streams that contain the same data with different rates. In the case of video, this means sending the same video in different qualities. Receivers select one of the streams according to their available bandwidth. They can change this stream with another when the network conditions change. Receivers are also divided into sets that receive

the different streams. Feedback is collected from each set so that the rate for this set is adjusted to meet its receivers capabilities. In addition to feedback implosion, redundancy is a major issue with this scheme. Recently however, the authors of [31] advocated that the layered approach is not always better than their scheme.

2.4.3 Pragmatic General Multicast Congestion Control

PGMCC [64] is a TCP-friendly protocol which is suitable for both non-reliable and reliable data transfers. It is based on PGM (Pragmatic General Multicast) [70]. PGMCC uses window-based TCP-like congestion control based on positive ACKs that are exchanged between the sender and a group of representatives called the *ackers*. PGMCC shows fairness to TCP but the results also show that it may cause large variations in the sending rate. Moreover, selection of the *acker* is very crucial to the performance of PGMCC [75].

2.4.4 TCP-friendly Multicast Congestion Control

An extension for equation-based congestion control [21] to multicast applications was recently presented in [75]. This requires calculation of the round-trip time and collecting and processing feedback. While the authors demonstrated favorable aspects of their design, they stated that it has a weakness in the startup phase. It can take a long time for sufficiently enough receivers to measure their round-trip time and report it to the sender. So it is not suitable for short-lived multicast sessions.

2.5 Conclusion

In this chapter, we have presented end-to-end multicast congestion control. The elements of a multicast congestion control architecture were presented first followed by the evaluation criteria. Prominent examples of the two classes of schemes, the receiver-based and the sender-based, were presented and discussed.

In general we can identify two major problems with sender-based techniques:

1. Because of the approach reliance on feedback from *all* receivers, a single slow receiver may drag down the data rate for the whole group.
2. Feedback from all receivers is not scalable. Some solutions are based on selecting an agent or a representative of the group to send the feedback. This presents the overhead of selecting this agent and changing it with changing network conditions.

On the other hand, although the receiver-based techniques are a good solution to the heterogeneity problem, they have a number of other problems that are common to most of these techniques:

1. In a best-effort IP network, which drops packets uniformly at congestion time, packets from the basic layer may be lost which makes receiving higher layers useless.
2. Layered techniques assume that all layers (multicast groups) will follow the same

multicast tree even when they are sent separately. This can not be guaranteed in IP networks.

3. Most of these techniques have fairness problems due to the way they react to congestion and the distribution of data across the layers.

Although the majority of recently proposed schemes use the layered approach, light feedback systems can outperform layered systems [26] if they are designed to avoid the problems we reported earlier.

In the next chapter, we present our study of layered multicast that shows problems that are inherent in the layered approach and how we believe that they can not be corrected.

Chapter 3

Performance of Layered Multicast Video

In this chapter, we present a performance study of the layered approach to multicast congestion control for video applications. Our study is based on simulation. In Chapter 2, we presented the major problems that arise from the underlying assumptions that this approach is based on. Here we investigate some problems that this approach will create when it is used in current IP networks. In particular we study: a) How the congestion control mechanism of layered video interacts with that of TCP. b) The effect of the distribution of video traffic across layers in layered video on fairness to TCP. c) The effect of VBR (Variable Bit Rate) video on fairness to TCP. We show that the fairness of layered video to TCP (when they compete for bandwidth) is affected by all the above factors. We also show that fairness of layered video comes at

the expense of instability of the video quality and poor link utilization.

3.1 Problems of Layered Multicast

We consider the most commonly cited layered video protocol in the literature. This is the Receiver-driven Layered Multicasting (RLM) proposed by McCanne [51, 53] which we presented in Section 2.3.1. It couples layered video encoding with layered multicasting (sending each layer as a separate multicast group). RLM is a good solution to the heterogeneity problem as it moves the control (of the rate that a certain receiver gets) to the receiver rather as opposed to the sender-driven approaches.

Designers of RLM focussed on its ability to reach the maximum number of layers in existence of the problem of receivers heterogeneity. They also showed that it adapts to congestion. This is all from *the video-receiver point of view*. Additional metrics are needed to give more insight into the protocol performance. In [24], the authors investigated the stability, the scalability of RLM and their relation to the protocol fairness among RLM receivers. They showed that RLM shows significant instability with VBR traffic. They also showed that when an RLM receiver achieves its optimal number of layers, other RLM receivers may not be able to get good quality. They insisted that for a video receiver, stability is the most important metric as frequent changes in quality usually are not tolerable. In [34], the authors investigated a list of problems with RLM. Among these problems, they noted that RLM competes very aggressively with TCP and they related that to the high loss threshold that RLM

uses. In this chapter, we try to focus on the extent and the cause of such RLM unfairness to TCP.

3.2 Fairness on the Internet

The Transmission Control Protocol (TCP) [61], widely used in the Internet for providing reliable end-to-end transport, incorporates a window-based congestion control algorithm to reduce congestion in the network. The protocol starts by increasing its sending window in a “slow start” fashion until the connection either experiences congestion somewhere in the network or is limited by the offered window of the receiver. When congestion is inferred via packet loss, TCP halves its sending window and begins a phase called “congestion avoidance”. In this phase, the window is additively increased during periods of non-congestion and multiplicatively decreased when congestion is detected. This results in a fair sharing of network bandwidth even though the protocol operates in a distributed manner [10]. Unfortunately, not all the protocols on the Internet behave in this way which creates problems of unfairness in terms of bandwidth allocation between competing flows.

In [39], a set of tests were provided to classify flows as *unresponsive*, *not TCP-Friendly* or simply using disproportionate bandwidth. An *Unresponsive* flow [20] is one failing to reduce its offered load at a router in response to an increased packet drop rate. A flow that is *not TCP-Friendly* [20] is one whose long-term arrival rate exceeds that of any conformant TCP in the same circumstances. Unresponsive flows

do not use end-to-end congestion control, and do not reduce their load on the network when subjected to packet drops. Such behavior will result in both unfair allocation of bandwidth and the danger of congestion collapse. Congestion collapse occurs when an increase in the network load results in a decrease in the useful work done by the network. It was first reported in the mid 1980's [56]. Unfair allocation occurs when TCP responds to congestion by reducing its transmission window while other non-TCP flows does not. The largest unresolved issue with respect to either unfair allocation or congestion collapse is due primarily to the increasing deployment of applications that do not use end-to-end congestion control [20]. Video applications are a key example in this category of applications. Some work has been done on providing rate control and adaptation techniques [30, 68, 69, 73].

One of the features that most video applications share is that they rely on a non-TCP protocol (usually UDP with some application level control or UDP with RTP). This is because video applications are both loss-tolerant, to some extent, and time-sensitive. Using TCP for these applications introduces delays because of the congestion control mechanisms that TCP uses. On the other hand, the level of reliability that TCP provides is not required for video applications as it is in the case of data transfer. That is why most of the video applications on the Internet are based on UDP. As these applications become widespread, it is important to ensure that they are *able to co-exist* with the current TCP-based applications on the Internet. In the case of video applications, congestion control should be augmented in a level

Figure 3.1: Simulation setup

higher than UDP. The reason is that UDP does not provide any congestion control capabilities to the applications that operate on top of it as opposed to TCP.

We considered RLM for our study because it is the foundation for all recent layered video protocols. In addition to the way RLM handles congestion, we believe that fairness is affected by other factors as well. Hence, we first study the bandwidth allocation between *competing flows of RLM and TCP*. Both RLM and TCP are non-stationary and non-linear [51] which makes a formal analysis of their interaction very difficult. Our study is based on simulation. Our results show that fairness is sensitive to RLM's congestion control mechanism, to the rate distribution across layers, and to the burstiness of video traffic.

3.3 Simulations

In this section, we describe the simulation setup used for our experiments. All the simulations were carried out using the LBNL network simulator *ns-2* [52]. The topology is illustrated in Fig. 3.1. One source of video traffic is used. As for TCP traffic,

ftp1 and ftp2 are two sources representing ftp applications.

There are two receivers, R1 and R2 with different link capacities, 250 Kbps and 100 Kbps respectively. The ftp application is running on top of TCP where ftp1 is sending its packets to R1 and ftp2 is sending to R2. All simulations were run for 6000 simulated seconds. The senders and receivers are connected via a link of 1 Mbps as shown in the figure. The video source is modeled as a set of traffic sources, each source is representing one layer of video. We will describe the characteristics of each source and the number of layers later with each experiment. Buffers at the nodes are managed using a simple drop-tail strategy where incoming packets are dropped when the buffer is full. All simulations are done using Tahoe TCP. In the rest of this chapter, when we refer to TCP we mean Tahoe TCP unless otherwise stated. We used the RLM implementation in *ns-2*.

We chose the propagation delay for the links to be relatively small (10msec) so that delay is not a factor in TCP's reaction. We wanted to make sure that high delay does not affect the fairness study we are conducting.

3.4 Congestion Control of layered video

In Section 2.3.1, we described how RLM reacts to congestion. Now we present our study of its interaction with TCP. This is important as TCP traffic constitutes the majority of Internet traffic. Hence, before introducing layered video to the Internet, it has to be tested for fairness to TCP. We want to show that the way RLM (and any

layered video transport protocol) changes its layer subscription has a direct impact on TCP.

3.4.1 Experiment

We started by reproducing the results in [51]. Then we built the simulation setup in Fig. 3.1 to run our experiments using *ns-2*. We simulated an RLM video multicasting session from the video source to R1 and R2. In the same simulation, we transferred two files, each of them is 5MB. One file from ftp1 to R1 and the other from ftp2 to R2. The video source is modeled as a set of CBR sources that emulate a video source with different layers with CBR stream at each layer. We used four CBR sources emulating ONE video source with four layers. The rates for the layers are 32Kbps, 64Kbps, 128Kbps, and 256Kbps. Rates are distributed in an exponential fashion with 32Kbps for the basic layer. The rate is doubled as we go one layer up.

Based on this setup, we simulated different video/data transmission scenarios. a) when RLM starts first, b) when both RLM and TCP start at the same time, c) when TCP starts first. We present only the case where RLM starts first as it is the case with severest unfairness. The two other cases do not present a significant addition to the results presented here. Interested readers may refer to our work in [44] for complete results of two other cases, namely the case where TCP starts first and the case where both protocols start at the same time.

For the results we show some measured metrics (number of video layers, RLM

throughput, TCP throughput, etc.) for each of R1 and R2 with different values of W . W is the length of the window in packets during which RLM measures its packet loss (as implemented in *ns*). This parameter controls how fast RLM reacts to congestion. An RLM receiver drops its highest layer of video if a certain percentage of packets is lost from a measuring window of W packets. With large values of W , RLM reacts to loss very slowly which causes competing protocols (in our case TCP) to lower their sending rates first.

It is worth mentioning that the frequency of joining a higher layer is dependent on a join timer, T_J , which is maintained by each receiver for each layer. This set of timers are given values based on the results of the join experiments [51]. The initial setting and the maintenance of these timers play a crucial role in shaping the behavior of RLM and how aggressive it behaves when it tries to join more layers.

In this set of experiments, first we start RLM at the beginning of the simulation, then after 100 seconds we start TCP (enough time for RLM receivers to reach stability). In Fig. 3.2, for R1 we show the number of received layers, RLM throughput, TCP window size in packets, TCP throughput, video packet loss ratio¹ shown against time (in log scale), and finally the link utilization.

In Fig. 3.2, $W = 12$, which is fairly large, this means that when RLM reaches stability (in this case 3 layers for R1), it will respond to packet loss if it loses 25%²

¹for an RLM receiver at a given time t , this is the total number of lost packets till time t divided by the total number of packets received till time t

²value used in ns-2 implementation of RLM

of the packets received in a window of 12 packets. The result is a very persistent behavior from R1. It tries to keep layer 3 despite the attempt of TCP to get some bandwidth and despite the increasing packet loss. TCP can not finish earlier than 1500 seconds and window size keeps oscillating with a maximum value of 10. This value is fairly small considering the available bandwidth on the link. In the figure, the video packet loss ratio is high only in the transient phase until RLM reaches stability. Also, we can notice the stability in RLM throughput and the high link utilization.

In Fig. 3.3, for R2 we see almost the same behavior for the same value of $W=12$. The difference here is that when R2 gets two layers (96Kbps), it leaves only 4Kbps for TCP. This is shown in the very small TCP window size and the higher loss than R1's case. This leads to R2 dropping of its highest layer (layer 2) after 1500 seconds. This drop lasts for around 500 seconds that were enough for TCP to finish the file transfer. This drop is shown also in both RLM throughput and in the link utilization. It is clear how slow RLM is in this case in responding to congestion. For a fairly long time it maintained its subscription to three video layers and led TCP to bandwidth starvation.

In [44], we repeated our experiment for different values of W . There was no significant difference in comparison with our earlier experiment until W is set to a small value. We selected $W = 4$. This selection results in a much faster response from RLM to packet loss. In Fig. 3.4, R1 drops layer 3 after 1000 seconds which allows

Figure 3.2: RLM starts first, R1 with $w=12$

TCP to finish earlier compared to the case shown in Fig. 3.2 (which corresponds to $W = 12$). R2 in Fig. 3.5 behaves in the same fashion. It drops its highest layer faster than the earlier case of Fig. 3.3 and again TCP finishes earlier.

Figure 3.3: RLM starts first, R2 with $w=12$

3.4.2 Discussion

From these results it is clear that in order for RLM to perform as described in [51], it shows an unfair behavior towards other protocols that try to share the bandwidth

Figure 3.4: RLM starts first, R1 with $w=4$

with RLM. This unfairness was reported in [24] towards other RLM flows. Our results show that RLM is also unfair to TCP. This unfairness is the result of the way TCP reacts to packet loss by reducing its window size with the first packet loss. Meanwhile, the mechanism that RLM employs to react to congestion is different. It is

Figure 3.5: RLM starts first, R2 with $w=4$

very reluctant to drop its highest layer (and hence reduce its rate) when it reaches its optimal number of layers. Also, RLM performance is greatly dependent on the setting and maintenance of a set of parameters. Setting these parameters to certain values may result in a fairer behavior from RLM towards TCP. However, there is a tradeoff

between fairness and the quality of the received video that should be resolved before setting these parameters to certain values. More analysis of RLM's other problems can be found in [24, 34, 44].

3.5 Rate Distribution for layered video

In this section, we study the effect of the distribution of the bandwidth of the video source across the layers in layered multicast video. In particular we compare the impact on fairness by distributing the layered video rates in linear, exponential or hyperbolic fashions. We are using RLM and the simulation setup of Fig. 3.1. Also, for RLM we are using $W = 4$ to insure fast response from RLM. The ftp file size in these experiments is 10MB. The video source is modeled as a set of CBR sources to represent the layers. In the three cases presented here, we are producing approximately the same average rate for the whole source.

3.5.1 Linear increase of layered video rates

In this case, the source is modeled as a set of 10 CBR layers each with rate of 25Kbps. This distribution has an advantage of stability for the video and reduction in big oscillations in the network. Also a good link utilization as the amount of bandwidth gain or released by RLM is not large. The disadvantages are, unfairness to TCP. Compare results in Fig. 3.6 and Fig. 3.7 to Sections 3.5.2 and 3.5.3. TCP

takes longer time to finish because in this case RLM release only 25Kbps at time when congestion is detected. From the video application's point of view, convergence in this case takes longer time. There is also the overhead of managing 10 layers of video at the receiver (note that they are sent as 10 different multicast groups).

Figure 3.6: CBR linear R1

Figure 3.7: CBR linear R2

3.5.2 Exponential increase of layered video rates

In this case, the source is modeled as a set of 4 CBR layers with rates, 16Kbps, 32Kbps, 64Kbps, and 128Kbps. The basic layer is 16Kbps. With this distribution, TCP gets

a bigger share of bandwidth compared to the two other methods of distributing the rates (linear and hyperbolic). It is clear that this happens only because of the granularity of the layers. Upon congestion detection, RLM drops the highest layer which corresponds to the highest bandwidth among all layers. The disadvantages are clear too. Large oscillations in subscription level, throughput and most importantly, link utilization. Results for this distribution are in Fig. 3.8 and Fig. 3.9.

3.5.3 Hyperbolic increase of layered video rates

In this case, we used 6 layers. The rates are distributed as 100Kbps for the base layer, and 50Kbps, 50Kbps, 25Kbps, 25Kbps, 25Kbps for the higher layers in ascending order. Because the basic layer is 100Kbps, we increased the bandwidth for R2 from 100Kbps to 150Kbps. The results in this case depend on both the granularity of the layer and available bandwidth for the receiver. In Fig. 3.10, R1 has 250Kbps, RLM's throughput is very stable, and hence RLM is unfair to TCP as shown in the low TCP throughput. Link utilization is very high. In the same experiment, R2 that has only enough bandwidth for the first 2 layers dropped the whole session for around 400 seconds, receiving no layers. This is because in order to get the minimum quality, the receiver has to secure 100Kbps at least. From Fig. 3.11, we can see that TCP got all the bandwidth when RLM dropped to zero layers.

Figure 3.8: CBR Exponential R1

3.6 Study with VBR Traffic

In this section, we study the effect of bursty video sources on RLM performance and on fairness to TCP. The source in these experiments is modeled as a set of VBR

Figure 3.9: CBR Exponential R2

sources. Each layer is represented by an on-off source. The distribution of the “on” periods is Pareto with mean 500ms and shape parameter of 1.5. The “off” periods have the exact same distribution. The rates of the “on” periods are set to replicate the experiments in Section 3.5, i.e., linear, exponential, and hyperbolic. The rate for

Figure 3.10: CBR Hyperbolic R1

the “off” periods are all set to zero. We chose Pareto because we wanted to use a source that exhibits self-similarity. It is known that the aggregation of a large number of alternating on-off sources with either (or both) “on” or “off” periods being heavy-tailed (Pareto) results in an aggregate traffic that exhibits long-range dependence.

Figure 3.11: CBR Hyperbolic R2

We can see from the results in the cases in Sections 3.6.1, 3.6.2, and 3.6.3 that they are close to their counterpart in Section 3.5. But with VBR, RLM is more unstable and link utilization is lower.

3.6.1 Linear increase of layered video rates

In this case, source is the on-off source we described earlier. The rates for the “on” periods are 50Kbps each. With the number of layers set to 10. The average output of the source in this case is approximately 250Kbps. Results are in Fig. 3.12 and Fig. 3.13. Advantages and disadvantages are the same as those in Section 3.5.1 but here, as we mentioned earlier, RLM is more unstable and the link utilization is low.

3.6.2 Exponential increase of layered video rates

We used 6 layers for the source. The rates for the “on” periods in each layer are, 32Kbps, 64Kbps, 128Kbps, 256Kbps, 512Kbps, and 1024Kbps. Results are in Fig. 3.14 and Fig. 3.15. Results are similar to those in Fig. 3.8 and Fig. 3.9 with less stability for RLM and less bandwidth utilization.

3.6.3 Hyperbolic increase of layered video rates

We used 6 layers for the source. The rates for the “on” periods in each layer are, 100Kbps, 75Kbps, 75Kbps, 50Kbps, 50Kbps, and 50Kbps. The bandwidth for R2 is increased to 150Kbps. Results are in Fig. 3.16 and Fig. 3.17. In the same manner of exponential rate distributions, hyperbolic VBR rates show similar performance to that of Fig. 3.10 and Fig. 3.11 with less stability for RLM and less bandwidth utilization.

Figure 3.12: VBR linear R1

3.7 Conclusion

We have presented a study of the performance of layered video in IP networks. We have used RLM as an example of layered video transport protocol. Our results show

Figure 3.13: VBR linear R2

that fairness is affected by the congestion control mechanism of the video protocol, by the granularity of the layers rates, and by the burstiness of the video source.

From the results in Section 3.5, we conclude that the distribution of video information in this layered fashion will always have problems because of the granularity

Figure 3.14: VBR Exponential R1

of the layers rates. This is a characteristic that is inherent in the layered approach to video multicasting.

Based on this study and on the discussion in Chapter 2, we believe that meeting all the requirements of video multicasting is hard to achieve by pure end-to-end

Figure 3.15: VBR Exponential R2

approaches. Some network support is required. This is why we build our future work in this area on a QoS-aware network architecture.

Figure 3.16: VBR Hyperbolic R1

Figure 3.17: VBR Hyperbolic R2

Part II

Network-Supported IP Multicast

Part II includes our research in the area of network-supported multicast congestion control. We started by classifying and evaluating some proposals of router techniques that support multicast congestion control in Chapter 4. Chapter 5 presents our work on building the multicast MPEG4 simulation environment. Chapter 6 presents the end-to-end architecture that is based on priority-based networking and ECN and Chapter 7 expands the work of Chapter 6 by focusing on the IEFT AF services model. Chapter 8 concludes the thesis and presents suggestions for future research in this area.

Chapter 4

Evaluation of Router-Techniques to Support IP Multicast Congestion Control

The majority of the techniques proposed for multicast congestion control are so far pure end-to-end techniques. This chapter presents an overview of router techniques that support congestion control of IP multicast applications. We discuss the criteria we use for the evaluation of the different router techniques we present. Two general categories of these router techniques can be identified in the literature. First, single-rate techniques that utilize filtering of multicast packets by the routers. Second, receiver-based multi-layer techniques that rely on routers group/flow control of multicast sessions. For each of these categories, there are advantages and disadvan-

tages that makes it more suitable for some applications than others. We evaluate a number of techniques that belong to each of the two categories.

4.1 Evaluation Criteria for Router Techniques

In this section, we present the criteria that we use to evaluate the router techniques we present in this chapter. First we consider the complexity of the techniques, both time and space complexity. Then we investigate the time it may take before it can be deployed on the Internet. Finally, we check how it compares with current end-to-end techniques. The rationale behind this order is that there is a major agreement between researchers and network designers that complexity should be pushed to the edges of the network whenever possible. So we will not consider techniques that introduce a considerable level of complexity to routers' architecture as "good" techniques. Then, if it proves to have moderate complexity, we consider the deployment time. We also compare it with end-to-end techniques. There is no need to consider the technique if it is too complex or if it may take a long time before it can be used. Even if it proves to be a better performer than end-to-end techniques.

4.1.1 Complexity

We consider the complexity of the techniques in terms of both space and processing time requirements. Techniques that require keeping a state per flow are considered

complex in terms of their memory requirement. On the other hand, if the technique uses little memory but is CPU intensive, it is considered as inefficient and most probably will not be adopted by router manufacturers.

4.1.2 Time to deploy

The architecture, size, and use of the Internet make it difficult to change. This is the reason why we consider *the time to deploy* any of the router techniques a very important factor in the assessment of any proposal targeting the current Internet. Most of the techniques we will discuss shortly claim better performance than existing end-to-end techniques. This claim sometimes overlooks the fact that it may take years before the implementation of this technique becomes feasible if ever. The reason is that these techniques require major changes in the current routers architecture/functionality.

4.1.3 Comparison with end-to-end techniques

If the technique proves to be of moderate complexity and can be deployed in the near future, we then consider how it compares with end-to-end techniques. We look for its scalability and how it handles heterogeneity.

Figure 4.1: Single-rate with filtering

4.2 The Two Categories of Router Techniques

We consider multicast congestion control mechanisms that rely on router support. We categorize these mechanisms into two categories. The first one is the sender-based, single-rate techniques that rely on *packet filtering* at the router. Filtering is dropping packets at the routers during congestion based on some criteria such as the priority of the packet. The second category is multi-rate (layered) techniques that rely on sending the data in layers and let routers (rather than senders or receivers) control the subscription to the layers and the flow control of each layer.

4.3 Single-rate with packet filtering

This class of techniques is based usually on an active queueing mechanism at the router. This differs from the normal queueing operation of the router in that it discards packets during congestion based on some criteria as opposed to the traditional drop tail strategy. In general, techniques that belong to this category consist of four parts. These are:

1. Some mechanism to detect or predict congestion
2. Report this to a regulating point (the router)
3. Decide on the rate
4. Then comes the actual regulation of the rate which is in this case using filtering

For reliable multicast, these techniques are usually combined with an FEC technique or a local retransmission and recovery mechanism. A general framework for these techniques is shown in Figure 4.1. (Note that in the figure we represent higher rates with thicker arrows)

4.3.1 Filtering based on sequence numbers

The authors in [38] present a protocol that is generally a representative of packet filtering techniques. They target their work at reliable content distribution from a

single server to a large number of receivers. For reliability, they recommend the combination of their protocol with an FEC reliability protocol.

The filtering part of [38] is based on a sequence number that is included in each packet. Assuming that routers receive congestion reports and that an algorithm exists that translates these reports into a fraction f_i to be transmitted to interface i of the total rate R received from the source. If a packet is received at the router for possible transmission at interface i with a sequence number n , a real number b ($0 \leq b \leq 1$) is formed by reversing the bits of the binary representation of the sequence number n . The packet is allowed to go on interface i if $b \leq f_i$, otherwise the packet is filtered out. The mathematical rationale behind this can be found in [38].

The protocol, however, does not specify how the rate should be calculated and leaves it to be based on the type of fairness desired by the protocol implementation. This calculation can increase the complexity of the approach since it will be performed in the router. On the other hand, the filtering process itself is simple and easy to implement. It also does not require a per flow state in the router. The time required to deploy this approach is not easy to predict since there are some components that the protocol does not specify. One final note is on scalability: the approach requires the collection of congestion reports from receivers and sending it upstream. Without special handling of this feedback, this is likely to cause scalability problems.

4.3.2 Tag-based filtering

Clerget [13] advocates the necessity of network support for multicast congestion control. He presents a scheme for UDP multicast flow control that relies on a *tag* calculated at the source and included in the packet. Routers filter multicast flows based on this tag and drop packets that are below a certain threshold. So the approach has two elaborate functions, tagging at the source and choosing a threshold. This is not however, adding to the complexity of the router design. Deployment of this approach should not be a problem as it requires only adding the simple filters at the routers and the major part which is *tagging* takes place at the source. The simulation results [13] also shows its fairness to TCP.

4.3.3 Efficient Congestion Avoidance Mechanism

A technique that is based on a combination of Explicit Congestion Notification (ECN) and Random Early Detection (RED) is proposed in [14]. The approach is called Efficient Congestion Avoidance Mechanism (ECAM). The source sends information in one multicast flow using a single rate. While congestion is building, the routers detect that by monitoring the average queue size. Comparing the queue size to RED's thresholds, the router acts differently. For flows that are marked by the source as ECN capable, the router may send an ICMP [60] message to the source to let it know about the incipient congestion. This results in rate reduction at the source to react to congestion.

This approach relies on ECN and RED which are already proposed for unicast and are used in some routers in the Internet. So ECAM is not adding more complexity, except for the part that requires sending an ICMP message back to the sender. Since the algorithm is proposed for both unicast and multicast, the authors did not provide any special attention to the problem of feedback (ICMP messages) implosion which may be caused by the replication of multicast packets at the routers. Also, the handling of heterogeneity in is not very efficient since the source in this case will adapt to the slowest receiver.

4.4 Multi-rate (layered) flow control

In this category of techniques, the data is sent in layers. Routers manage these layers and keep track of receivers subscription to the layers. In this category, the overhead on the routers is usually more than the filtering case. A general framework for this category is illustrated in Figure 4.2.

4.4.1 Receiver-Selectable Loss Priorities

The authors of [25] proposed RSLP (Receiver-Selectable Loss Priorities) as a means to implement a simple two-level priority dropping scheme at the routers. They combine that with receiver-driven layered multicast and claim to achieve better performance than RLM [51]. Receivers in this case subscribe to as many layers of data (or video)

Figure 4.2: Multi-rate with router flow control

as their bandwidth allows them. These layers are sent by the source as different multicast groups. The receiver has the option in this case (with RSLP implemented at the routers) to subscribe to a layer at higher priority or lower priority. During congestion, the router attached to the congested link drops packets associated with groups mapped as lower priority at this router. In this case, the receiver rather than the sender sets the priority. The router, however, has to keep track of the priorities per layer (multicast group). This has the advantage of more feasibility of changing packet priorities at multicast branch points. The priority here is not associated with the packet, it is set at the router for certain layer. Flow management is based on the information request by the receiver.

The complexity of this approach is simpler than the case of implementing a full-priority dropping mechanism at the router. However, there should be a good signaling technique between receivers and routers. In addition to that, there is the complexity of keeping track of the priority per multicast flow at each router. The results in [25] shows that it avoids the problems of general layered multicast schemes.

4.4.2 Network-supported Layered Multicast

In the same fashion of [13], the authors of [57] start their paper by advocating network-supported congestion control. Specifically, they propose the use of priority dropping in the routers to enhance the traditional receiver-driven layered multicast. They call their scheme NLM (Network-supported Layered Multicast). In NLM, the sender hierarchically encodes the data into several CBR layers and sends all of them in a *single* multicast session. The layer number should be included in the header of the packet so that the router is aware of the layer that this packet belongs to in a multicast session. Receivers join a *session* with all its layers and it is up to the router to decide on how many layers to send to the receiver based on the congestion status of the router. There are three functions at the router, filtering of a certain layer going out to a certain interface, removing or adding a layer to a certain interface based on signaling coming from the receiver, and keeping track of the session state tables.

The simulation of NLM in [57] shows better performance than RLM [51] and the authors argue that their results indicate good responsiveness, fairness, and stability

of goodput. However, they did not address the complexity of their protocol. Apart from receivers signaling to add/drop a layer, the router in NLM has to do almost all of the flow/congestion control functions. At each interface, it has to keep track of which layer belongs to which session and decide how many layers to send. This introduces a considerable level of state and processing overhead at the routers. This definitely makes NLM not suitable for immediate deployment on the Internet.

4.4.3 Router-Assisted Layered Multicast

RALM [77] (Router-Assisted Layered Multicast) is another variant of layered multicast that is based on router assistance. The basic idea of RALM is router-controlled suspension/retry for layered multicast. An RALM-aware router monitors the queue status of each of its outgoing links. If congestion is detected on one of the links, the router immediately suspends some of the current transmitted groups on that link temporarily. The choice of which group is suspended is based on the importance of the data they are carrying. This importance is set by the sender. Routers will try to re-activate a suspended group on an outgoing link when congestion is relieved on this link. If a group is suspended on all its outgoing links, the router will send a leave message upstream for this group. The router then will not retry for this group unless a receiver explicitly joins this group later. The sender and receiver has a fairly larger role in this scheme than in the case of NLM.

However, there is a considerable overhead for the router in RALM. States of the

groups have to be monitored by the routers. Also, all suspend/retry operations are carried out by the router. However, RALM is designed in a manner that allows incremental deployment, if all routers in the network are not RALM-aware, RALM defaults to RLM.

4.5 Conclusion

We have presented an evaluation of router techniques to support congestion control for IP Multicast. We grouped them into two main categories. Then, we presented and evaluated a number of these techniques. We can conclude that these techniques solve some of the problems that are common in pure end-to-end techniques. This comes at the expense of increased complexity. In some cases, deployment of the proposed technique on the Internet may not be feasible in the near future. We can also say that, in general, filtering techniques present routers with less overhead than the layered multicast case. Router support for multi-rate techniques is not likely to be widely deployed in the near future.

Chapter 5

Modeling and Simulation of Real-time Video

Because of the very complex nature of current networking architectures, mathematical analysis is not feasible in many networking research problems. Simulation is a more common tool for studying networking systems properties.

In this chapter, we present our work [49] that enables the study of MPEG4 properties and performance using simulation. We develop a traffic generator that is able to produce traffic that has almost the same first and second order statistics (marginal distribution and auto-correlation function) as an original trace of MPEG4 frames that is generated using an MPEG4 encoder. We model and generate this traffic based on the Transform Expand Sample (TES) [63, 29, 54] methodology and using *TESool* [55].

We present the model and show the performance of the generator in terms of good matching to the characteristics of the modeled trace. We also show how we implemented this model in the network simulator *ns-2* [52] so that we can generate the same MPEG4 trace encoded in different quality levels and divide it into different priority levels. The generator proved to be useful for other research projects and currently it is used by different research groups in the networking research community. It is now part of the *ns-2* contributed code [43].

5.1 MPEG4 Video Encoding

The transfer of digital video over the Internet has been made feasible by the advancement of digital video encoding techniques. These techniques reduce the bandwidth required for this transfer to a practical level. This is one of the motivations behind establishing the ISO Moving-Picture-Experts-Group (MPEG) family of standards [1, 3] for digital video encoding. This family of standards includes MPEG1, MPEG2, and MPEG4. MPEG7 is under development.

MPEG4 is a suitable video encoding technique for the Internet [32]. The following summarizes its features:

- It is targeted at providing high video quality at relatively low bit rates.
- It allows real images to co-exist with computer-generated counterparts and also allows their separation and their receiving different treatment due to interaction

with the user.

- It allows interactivity of the users at the decoder side.
- Scalable content, which means encoding once and playing back at different speeds.
- Real-time adaptive encoding that results in changing the sending rate in reaction to changing network conditions. This enhances network utilization and enables MPEG4 senders to be more responsive to changes in network conditions.
- Scalable encoding that differentiates between important objects and less important objects. This results in streams that are suitable for transmission over networks that support priorities

MPEG4 generates video in three different frame types (I, P, and B) that serve to encode different portions of the video signal in different levels of quality. They can be described as follows:

- I frames (intra-coded frames) contain the information that results from encoding a still image, i.e., with no reference to any other image. They are points of reference and random access in the video stream. They can be decoded without the need for any other frames. The compression rate for these frames is the lowest of all frame types.
- P frames (predictively coded frames) require information from previous I frames

and/or P frames for encoding and decoding. By exploiting temporal redundancies, P frames achieve higher compression rates than that for I frames.

- B frames (bidirectionally predictively coded frames) require information from the previous and following I frames and/or P frames for encoding and decoding. They are predicted from both previous and following frames. They have the highest compression ration among all frame types.

Reference frames must be transmitted first. Therefore the transmission order and display order may differ, as described in Fig. 5.1. At the beginning, there is always an I frame. The first I frame and the first P frame serve as a reference for the first two B frames. At the first time, the first I frame is the reference for the first P frame. The second I frame serve as a reference for the B frames that are displayed before it, that is why it has to be sent first.

Figure 5.1: Frame transmission and display orders

5.2 Transform Expand Sample (TES) models

TES is a versatile methodology for generating data that closely match (in terms of its marginal distribution and auto-correlation function) any set of given observation of a time series. The derivation of TES models is performed in two phases [54]. First, a correlated sequence, with uniform marginal in $[0,1]$, (also called background TES process) is formed as follows:

$$U_n^+ = \begin{cases} U_0 & n = 0 \\ \langle U_{n-1}^+ + V_n \rangle & n > 0 \end{cases}$$

or

$$U_n^- = \begin{cases} U_n^+ & n \text{ even} \\ \langle 1 - U_n^+ \rangle & n \text{ odd} \end{cases}$$

where U_0 is a random variable uniformly distributed in $[0,1]$. $\{V_n\}$ is a sequence of iid random variables independent of U_0 , called the innovation sequence (with probability density f_v). The operator $\langle \rangle$ denotes the modulo-1 operation which for every real x is defined by:

$$\langle x \rangle = x - \max\{\text{integer}(n): n \leq x\}.$$

Both U_n^+ and U_n^- can generate lag-1 autocorrelations in the range $[0,1)$ and $[-1,0)$ respectively. Regardless if the probability law of the innovation density f_v , it can be shown [55] that the TES background sequence forms a stationary Markovian process with a uniform in $[0,1)$ marginal distribution. A smoothing operation called *the stitching transform* is used to get rid of the inherent discontinuity of the operator

[54]. So in short, in the first phase of the TES model development, the objective is to generate a time series of correlated random variables with uniform marginals $[0,1)$. The amount of correlation depends on the structure of the density function f_v of the innovation sequence $\{V_n\}$, which in general may be arbitrary.

In the second phase, synthetic sample data, called *the foreground sequence*, which resembles the real sample, may be derived from *the background sequence* using the inversion technique in [55]. This inversion technique allows the transformation of any uniform random variable to one with arbitrary distribution. More specifically, it is assumed that a histogram $H(\cdot)$ of the empirical video sequence has been built consisting of J cells, where the j^{th} cell is positioned on the interval $[l_j, r_j)$ and is characterized by a probability p_j . Such a histogram can be inverted as follows:

$$H^{-1}(x) = \sum_{j=1}^J [l_j + (x - c_{j-1}) \frac{w}{p_j}] I_{[c_{j-1}, c_j]}(x)$$

where I_A is the indicator function of the set A , $w_j = r_j - l_j$ is the width of cell j , and $c_j = \sum_{i=1}^j p_i$, where $1 \leq j \leq J$, and $(c_0 = 0, c_J = 1)$. The random variate $\{H^{-1}(x)\}_{n=1}^{\infty}$ has a marginal distribution $H(\cdot)$ and autocorrelation that depends upon the innovation density f_v .

The modeling process is carried out in two steps: the first step essentially captures the autocorrelation of the input time series using innovation f_v and applying a smoothing technique using the stitching transform [55]. This would result in the generation of the background sequence. The second step is to invert the background sequence using the empirical histogram, which always guarantees the matching of the

marginals. The task of finding a suitable f_v that approximates the empirical autocorrelation is carried out via a heuristic search using *TEStool* [55]. The matching of the histogram is guaranteed, while the matching of the autocorrelation is carried out in an interactive manner with visual feedback for the best f_v .

5.3 Modeling MPEG4 using TES

In this section, we present our model for MPEG4 traffic generation. We used a sequence of MPEG4 frames that was generated by encoding a news broadcast from the BBC at a rate of 30 frames/sec that was used in [27]. Here are the steps we followed to generate the traffic:

- We used a trace of frame sizes (in bits) of an MPEG4 encoded video. Fig. 5.2 shows the characteristics of the trace as generated while developing the model using the *TEStool*. Window w1 of the figure shows the sample path of the frame sizes (in bits), in w2 the autocorrelation function is shown, while w3 shows the histogram of the distribution of the frame sizes.
- We separated this trace into three traces, one for each frame type (I, P, and B). This facilitates the generation of a trace that matches the original trace. We experimented with the original trace as a whole without separation and it proved to be very difficult to develop a single model for this trace. This is due to the periodic nature of the autocorrelation function of the original trace [29].

After separation of frames, we model each of the traces using a separate TES model using the methodology described in Section 5.2.

- Fig. 5.3, Fig. 5.4, and Fig 5.5 show the models of the I frames, P frames, and B frames, respectively. In each figure, w4 shows the innovation density f_v that was described in Section 5.2. In all the figures, the generated data is shown in the darker trace while the original is shown in the lighter color trace. From the figures, we notice that the histogram always matches the original data while there is a little difference in both the autocorrelation function and the sample path.
- To generate the MPEG4 trace, we interleave generating samples from the three developed models to match the sequence that is shown in Fig. 5.1. The final trace that is generated by interleaving can be compared to the original trace as shown in Fig. 5.6.

5.4 Implementation and integration into the Network Simulator

We implemented this generator in software and integrated it into the Network Simulator *ns-2* [52]. This simulator is a discrete event simulator targeted at networking research. It provides substantial support for simulation of TCP,

routing, and multicast protocols over wired and wireless (local and satellite) networks.

We start by using *TEStool* to generate the histograms and the innovation density for the three frame types. Then we provide this as an input (the CDF of both the frame size distribution and the of the innovation density) to the code we wrote in C++. The code then uses each of the three models to generate a frame size of each type in the order mentioned in Fig. 5.1.

For our simulations, we need to change the rate and hence the quality of the generated video. To do that, we scale the generated trace up or down according to the feedback from the network. This is justified by the fact the actual traces generated from the MPEG4 encoder for the same video with different qualities have *exactly* the same sample path in different scales. We integrated this code into the *ns-2* simulator to generate traffic for simulation of IP networks. The code was contributed to *ns-2* contributed modules [43].

Figure 5.2: Characteristics of the original trace

Figure 5.3: The I frames model

Figure 5.4: The P frames model

Figure 5.5: The B frames model

Figure 5.6: Comparison between the original and the generated traces

Chapter 6

Explicit Congestion Notification for IP Multicast

In this chapter, we investigate the use of Explicit Congestion Notification with IP multicast. We start by presenting an overview of this work and discuss the RED [22] (Random Early Detection) buffer management strategy and its variant that we use in this chapter. We follow that by a discussion on how appropriate is the use of Explicit Congestion Notification [19] in real-time multicast video. We present our proposal of using BECN [66] (Backward ECN) in multicast congestion control for video applications. This includes a discussion of the end-to-end architecture. The chapter is concluded by our work on the use of MECN [66] (Multi-level ECN) in our architecture. We compare the results obtained using BECN with those obtained using MECN.

6.1 Overview

We present a comparison between the use of Explicit Congestion Control (ECN), Backward ECN (BECN), and Multi-level ECN (MECN), for IP multicast congestion control. The work in [48] and [50] is based on this comparison. We implement two of these ECN techniques in an IP network that supports priority dropping of packets during congestion using RIO [12] (RED with In/Out bits). RIO is an extension of RED that supports service differentiation in packet networks. This extension recognizes the priority of packets when they need to be dropped and drops lower priority packets first.

ECN techniques will provide early notification to the sender and/or the receivers about congestion that it is developing in the network. Based on which ECN technique is used, the sender and/or the receivers will react to this notification to reduce the congestion in the network. We discuss the feasibility of using ECN as an end-to-end mechanism before presenting the use of BECN (Backward ECN) in our proposal. After that, we use MECN that sends back quantitative reports that indicates *how congested* the router is. We evaluate how MECN (Multi-level ECN) can improve the performance of our architecture.

We investigate this in the context of multicasting adaptively-encoded MPEG4 over IP. We take advantage of two MPEG4's properties in our architecture: First, the ability to encode real-time video adaptively to target a certain rate. Second the ability to generate encoded video in two (or more) priority levels, one with basic

information and the other with enhancement information.

6.2 RIO (RED with In-Out bits)

Random Early Detection (RED) [22] is a buffer management strategy that is used for congestion avoidance/control in IP networks. RED routers try to early detect upcoming congestion by computing an average of the queue size in the router. A sustained long queue is a sign of network congestion. When a packet arrives, a RED router checks the average queue size against specified *min* and *max* thresholds. Based on this check, one of three actions is taken. IF the average queue length is less than *min* then no action is taken. IF average queue length is greater than *min* but less than *max* then with probability, the packet is dropped. Finally, if the average queue length is greater than *max* the packet is dropped.

We define a QoS-aware network model that supports our architecture for multicasting. We require the network to:

1. Support priority dropping of packets in time of congestion
2. Provide congestion notification from routers to the end systems (sender and/or receiver).

Our choice for fulfilling the first requirement is the RIO [12] buffer management strategy. RIO will be revisited in detail in Section 7.1. Using RIO in the network

routers, packets are marked with different priorities and are treated according to this priority when they need to be dropped during network congestion.

To achieve differentiation between different priority traffic classes, different sets of RED parameter values would need to be maintained for each class. In [12], two priority classes (called In and Out classes) were considered. In this case, two sets of parameters need to be maintained. Each set would affect arriving packets in its priority class based on its own RED parameters. RED will be managing a separate virtual queue for each traffic class. Certain calculations are performed to get the probability for dropping a packet. These calculations can be based on each virtual queue or by coupling them together [12].

6.3 Issues With The Use of ECN (Explicit Congestion Notification)

As described in Section 6.2, in RED buffer management, if the queue size is between its *min* and *max* thresholds, the packet is dropped with a probability. In the case of TCP unicast applications, dropping this packet will signal the end-to-end congestion avoidance mechanism of TCP that the network is congested.

Explicit Congestion Notification (ECN) [19] was proposed for congestion detection and avoidance with TCP. In ECN, the packet is marked and sent to the receiver if the queue size is between its *min* and *max* thresholds. The receiver in this case, marks a

flag in the TCP header of an ACK message and sends it back to the sender. Based on the information in this ACK, the TCP sender reacts by reducing its congestion window as well as its slow start threshold. The sender then sends a notification to the receivers that it did so to stop the receiver from sending more ACKs back. This mechanism forces the TCP sender to react early before congestion develops without the need to drop packets. However, considering this methodology for real-time video multicasting has some disadvantages:

1. This approach is not appropriate for use in multicast real-time applications because it is an end-to-end solution that requires *every* receiver to send a message back. This will lead to the *feedback implosion* problem described in Section 2.4.
2. It takes a round trip time (RTT) before the sender reacts. This is not suitable for delay-sensitive real-time video applications.

From this discussion, we conclude that in order to use ECN with real-time multicast applications, a variant of ECN needs to be devised for this purpose. This variant should enable reacting to congestion in less than a round-trip time. Also, it should not result in feedback from *all* receivers. BECN is a variant of ECN that satisfies these conditions and it is presented in the following section.

6.4 BECN (Backward Explicit Congestion Notification)

The authors in [66] proposed an extension to ECN using feedback from the router to the sender as an indication of congestion. This feedback message is sent, with a probability, if the average queue length is between its *min* and *max* thresholds or if it is greater than *max* threshold. The packet is still marked to prevent other routers from sending more feedback messages for the same packets. The sender responds to that in the same manner it does in the case of ECN. They called it Backward ECN (BECN). This is done using the existing IP signaling mechanism the Internet Control Message Protocol (ICMP) [60]. Sending an ICMP message to the sender from the router has an advantage over ECN which is the lower time it takes before the sender can react. Also, because it is an IP level mechanism it can work with transport protocols other than TCP.

In our architecture, we use BECN with the flow controller of the video application that operates on top of UDP. We implemented BECN in our simulations to work on parameters of the virtual queues. We send BECN messages¹ back to the video sender based on the status of each virtual queue, thus sending back information on which priority level is experiencing problems at the routers. We implemented BECN

¹In the case of multicast, the Internet Group Management Protocol (IGMP) [18] is used instead of ICMP

Figure 6.1: Protocol stack at the sender

and integrated it in ns-2 [52]. The authors in [66] also proposed that a variation of BECN may send some quantitative indication of the level of the congestion that is developing in the router. They called that Multilevel ECN or MECN. A performance study for BECN can be found in [40].

```

REPEAT every t Seconds
  IF nofeedback messages received
    THEN increase Total-Rate;
        reduce BasicLayerPortion;
        reduce t
  ELSEIF feedback increasing at ALL priority levels
    THEN reduce TotalRate;
        increase t
  ELSEIF feedback increasing at basic layer
    THEN reduce BasicLayerPortion;
        increase t
END REPEAT

```

Figure 6.2: Flow control algorithm at the sender

6.4.1 System architecture

In this section, we overview our preliminary proposed architecture for video multicasting. This architecture will be revised and enhanced in Chapter 7.

1. The flow/congestion control is sender-based.
2. The sender marks the video packets (MPEG4) with two different priorities assigning basic-information packets to the higher priority and assigning enhancement-information packets to the lower priority. During congestion, basic video quality can still be received because the information required to decode basic quality video was transmitted with a high priority. This is basically sending the video information in layers (different priorities) within one stream. This removes the

burden of dealing with different layers (multicast groups) at the receiver and ensuring that all packets will follow the same multicast tree (refer to Section 2.3 for problems with the layered multicast approach).

3. While congestion is developing, routers that run RIO will send back BECN messages to the sender with information on the priority level that caused the problem.
4. Based on the rate of these feedback messages, the sender runs an algorithm to search for an operating point (total sending rate and ratio between priority levels) that will reduce this feedback messages rate. The protocol stack² at the sender is shown in Fig. 6.1. Figure 6.2 shows a pseudo-code for the flow control algorithm (functionality of the *decide* box in Fig 6.1). The algorithm used in this chapter depends on experimentation in setting the magnitude of rate change. It will be revisited in Section 7.3 for a more elaborate rate adaptation methodology. Note that if the problem is only at the enhancement layer it is not fixed until both layers are congested simply because this enhancement layer will get dropped where congestion is experienced.
5. The main contribution of our architecture is that the sender will try to match the sending rate of the high priority video with the sending rate of slow receivers. This will allow them to get useful information in the time of persistent

²We show only the functionality that we proposed or modified

congestion. This is also how we deal with heterogeneity. Note that a receiver may get a 400Kbps out of the 1Mbps original signal and still can reconstruct a comprehensible video because the information was encoded with 400Kbps basic information layer. Without using our architecture, a slow receiver may still get the same rate from the same source but because the packets carry information with same level of importance (priority), nothing comprehensible can be reconstructed.

6.4.2 Simulation and performance

We used simulations to perform initial testing for our proposal. We used *ns-2* for running the simulations. In Fig. 6.3, we show the basic topology we used for simulations. There is one MPEG4 source that sends MPEG4 traffic using the generator we introduced in Section 5.1. The trace we used for the generating the model is from a news broadcast. Properties of the trace can be found in [49]. Three receivers R1, R2 and R3 are connected to the sender through a router as shown. The link from the sender to the router is 2Mbps and so is the link from the router to R3. We made this as high as the link from the sender to the router so that R3 will not have any problems receiving the full video multicast with no packet loss. R1 has a 400Kbps link to router and R2 has a 800Kbps one.

6.4.3 Results for BECN

The goal of our simulations is to see if and how the sender flow controller will converge to a point where both receivers will maximize the utilization of their links and keep the information received by each of them reconstructable. From Fig. 6.4, we can see the performance of R3 which did not have any loss.

In order to see how this works, we check Fig. 6.5 and Fig. 6.6. Both receivers R1, and R2 maximized their link utilization and more importantly they secured the basic layer during congestion. For R1, in part A of the figure, we can see that most of the received packets are from the basic layer while R2 is getting almost the same rate in basic layer and a higher rate in the enhancement layer. The sender always tries not to increase the rate for the basic layer to more than a level that is appropriate for the slowest receiver, R1. In part B of both figure we can see that the loss rate is much higher in the enhancement layer, specially for R1. We should also note the relation between the reduction of total rate and the increase of BECN messages. The times of higher loss rates are relatively short because BECN reduces the time the sender takes to react to upcoming congestion. There is also a need to reduce the number of BECN messages back to the sender.

Figure 6.3: Simulation setup

Figure 6.4: Throughput for Receiver R3 using BECN

Figure 6.5: Performance of Receiver R1 using BECN

Figure 6.6: Performance of Receiver R2 using BECN

6.5 MECN (Multi-level Explicit Congestion Notification)

In this section, we present our use of the MECN (Multi-level) [66] in our proposed architecture.

6.5.1 Difference from BECN

The difference between BECN and MECN is that instead of just sending back a BECN message to the sender, the MECN message contains quantitative information about the congestion status of the router queue that sent the message. There is no description in [66] on how MECN should be implemented. The method we used in this study is sending different values that indicate how big is the queue size with reference to the RED parameters min and max for a certain virtual queue i . In our study, we send back of three values to the sender as follows:

$$\begin{aligned}
 v_1 : & \text{ if } \{min \leq \text{QueueSize}(i) < (min + \frac{max - min}{2})\} \\
 v_2 : & \text{ if } \{(min + \frac{max - min}{2}) \leq \text{QueueSize}(i) < max\} \\
 v_3 : & \text{ if } \{\text{QueueSize}(i) \geq max\}
 \end{aligned}$$

The sender reacts to these values with different values for rate reduction following the same algorithm in Fig. 6.2. Receiving the value v_3 at the sender means that packets are being *dropped* and hence the rate reduction should be considerably high.

Values v_2 and v_1 represent lower levels of congestion and should result in lower rate reduction.

6.5.2 Results for MECN

We repeated the same simulation of Section 6.4.2 using the simulation setup of Fig. 6.3. The same end-to-end architecture of Section 6.4.1 is used in this part of the study.

The results show that using MECN gives a better convergence for the rates at the two different priorities than using BECN. We can see that by comparing fig-

Figure 6.7: Performance of Receiver R3 using MECN

Figure 6.8: Performance of Receiver R1 using MECN

Figure 6.9: Performance of Receiver R2 using MECN

ures 6.7, 6.8, and 6.9 to their counterparts of Section 6.4. In Fig. 6.7, the throughput of R3 is almost the same as that of Fig. 6.4 with a tendency to increase towards the end of simulation time. The difference is in the distribution of the rate among the two priority levels and the changes in the rate assigned to each layer. High priority layer is adjusted to 400Kbps to match the rate of R1. This is much better than the case of BECN where the rate was kept in that range with a much higher fluctuation. This would result in a change in the received quality that is not likely by desirable by the end user.

To check how this affects the two receivers individually, we first check Fig. 6.8. The rate for the high priority layer received by R1 converges to almost 400Kbps with little variation. This comes at the expense of high loss ratio at the low priority layer and an increasing loss at the high priority layer. Fig. 6.9 shows that a bigger portion of the traffic is from the higher layer specially towards the end of the simulation although it is not as good as it is in case of R1.

So the difference in results from BECN lies in a better matching of the higher priority rate (specially for the slowest receiver) and a smoother rate at that layer. The drawback is that it is still not accurate since the amount of increase or decrease of the rate is still determined using experimentation. In Section 7.3, we develop a more elaborate rate adaptation to get an accurate match of the available bandwidth and reduce the loss ratio.

6.6 Conclusion

In this chapter, we investigated the use of different ECN techniques for multicast congestion control. Using ECN in the same fashion it is used with TCP is not suitable for multicasting. Our proposal of using BECN or MECN has the following advantages:

1. Sending the video as one stream is much easier to handle than multiple multicast groups as opposed to the layered multicast approach discussed in Chapters 2 and 3.
2. Sending all layers within one stream with different priority labels over a network that supports priority dropping ensures that a minimum quality will be received in the time of network congestion
3. Scalability issues are minimized when the feedback to the sender is provided from the network routers rather than from receivers
4. Future network architectures support QoS in one way or the other so it is required to study how video multicasting will work in this environment

The results we obtained in this chapter show that this approach is promising. It also shows that using MECN gives the same overall performance as BECN but with the advantage of having a better convergence towards the target rate for the high priority video with a much smoother rate change. Chapter 7 concentrates on formalizing the rate adaptation process to make it more accurate and to reduce the loss ratio.

Chapter 7

Multicast Congestion Control for Assured Forwarding

This chapter presents our work on developing an architecture for multicasting real-time MPEG4 over IP networks that provide service differentiation. In particular, this chapter is targeted at video multicasting in the IETF Assured Forwarding (AF) [28] style networks. The architecture in this chapter assumes loss differentiation in the network using AF and assumes the network's ability to provide ECN messages to the sender in the manner explained in Chapter 6. For this network model, we build an end-to-end architecture and develop a rate adaptation algorithm that can operate in both unicast and multicast applications with a minor modification.

We considered AF because it helps us build a simple end-to-end architecture that avoids the problems of earlier approaches to multicast congestion control. It is

also expected to be deployed soon in Internet routers as opposed to other proposed support mechanisms (in Chapter 4) that require major changes in current router's functionality.

For the sake of simpler experiments, we do not completely implement the IETF proposed AF service. In particular, we do not consider marking/policing at the edge routers, instead packets are marked at the sender. Although our main focus is multicast, the architecture we propose here along with our rate adaptation algorithm operates equally well in both unicast and multicast applications with a minor modification. In this thesis, we only present the multicast case as it proved to be more challenging than unicast.

The simulation results [41, 42, 45, 46] show how the rate adaptation algorithm accommodates different receivers with different networking capabilities and provides receivers with different levels of quality by taking advantage of the queue management capabilities of the AF service. We test how the architecture scales to a large number of receivers, how multiple multicast sessions interact, and how it interacts with TCP.

The chapter is organized as follows. The queueing model of AF and the end-to-end architecture are presented in Sections 7.1 and 7.2 respectively. We discuss the rate adaptation algorithm in detail in Section 7.3. Simulation setup is explained in Section 7.4 and results are discussed in Section 7.5. In Section 7.6 we discuss some of the design issues and limitations.

7.1 Queue management for Assured Forwarding

In this section, we describe the queue management scheme of the Assured Forwarding service model we consider for our work. The choice of this model is motivated by the design goal of providing different levels of quality to receivers and guaranteeing (with a high probability) a minimum quality during congestion. We consider IP networks that support priority-dropping as a means of providing different levels of services to its users. Priority-dropping is the process of packet dropping during congestion by routers based on a priority level assigned to the packet at the sender or at an edge router. Assured Forwarding [28] is a good candidate to meet these requirements. AF-compliant routers achieve that by employing (AQM) Active Queue Management techniques that recognize packet priorities and enqueue, dequeue, and drop packets based on this priority.

Assured Forwarding provides assurance of packet delivery with a high probability. It also accepts traffic exceeding user's pre-assigned profiles with the understanding that the excess traffic is not expected to receive the same high probability of delivery. IETF RFC2597 [28] recommends providing four different classes of AF and allocation of a certain amount of resources to each class at each AF-compliant router. In the same RFC, three drop precedence levels (within each AF class) are also recommended. As the name suggests "drop precedence" is used in priority-dropping of packet within each AF class. It determines which packets should be dropped first within each class when needed.

Implementing priority-dropping in AF is based on Active Queuing Management techniques. In particular, AF is usually based on variants of Random Early Detection (RED) [22]. RED is a buffer management technique that is used for congestion avoidance in IP networks. RED routers try to early detect upcoming congestion by computing an average of the queue size in the router. A sustained long queue is a sign of network congestion. When a packet arrives, a RED router checks the average queue size against specified min_{th} and max_{th} thresholds. Based on this check, one of three actions is taken:

```

IF (QueueAverageLength < minth)

    THEN no action is taken

IF (minth < QueueAverageLength < maxth)

    THEN with probability, the packet is dropped

IF (QueueAverageLength > maxth)

    THEN packet is dropped

```

We assume that AF routers support one of RED's extensions for service differentiation. We namely consider both RIO (RED with In/Out bits) [12] and WRED (Weighted RED) [4]. Both RIO and WRED maintain a single queue for each AF class. Then, within a certain AF class, they maintain a different set of parameters for each drop precedence level and treat each of these levels as a different virtual queue. The difference between RIO and WRED is that WRED uses *one* average queue length

Figure 7.1: Staggered parameter configuration of a two-priority RED queue

to make dropping decisions while RIO uses *two*¹ averages to make dropping decisions. WRED calculates its average queue length based on the total number of packets in the queue. RIO maintains one average for the number of *in* (most important) packets and another average for the *out* packets. The number of RIO *out* packets can be calculated based on the total number of packets in the whole queue (coupled mode or RIO-C) or based on the number of *out* packets only (de-coupled mode or RIO-D).

In this thesis, we present the results of using a queue model with two drop precedence levels. However our architecture along with the simulation models we developed can support more priority levels without any change. We use the staggered configuration of parameters. This is setting $min_{th}(n-1) > max_{th}(n)$, where level $n-1$ is higher in priority than level n within a certain class. Check Fig. 7.1 for the staggered parameters setting of a RED queue with two priorities. *This configuration offers the maximum sheltering and isolation of classes* for both RIO and WRED. For RIO, in addition to staggered parameters, the choice of coupled versus de-coupled mode de-

¹two or more according to the number of drop precedence levels

termines which class is more sheltered. The coupled mode (RIO-C) drops the lower priority class in favor of the higher priority class which is more protected. RIO-C also allows bandwidth borrowing among classes when the load is light. In the de-coupled mode (RIO-D), lower priority is more protected than RIO-C because in this case its average queue length is based solely on the number of lower priority packets in the queue.

To summarize, results presented in this chapter are based on the following assumptions:

- A single RIO or WRED queue for a single class AF service.
- Two drop precedence levels within this class.
- The staggered setting of parameters is used to set the drop precedence levels as in Fig. 7.1.
- We do not use policing/shaping at network ingress points. Instead we perform flow control and packet marking at the sender
- We also assume that routers can provide Congestion Notification messages upstream to the sender with information about the router's congestion status. These messages are sent based on the specifications of BECN (Backward ECN) [66] as described in Chapter 6.

7.2 End-to-end architecture

We build an end-to-end architecture based on the AF model described in Section 7.1. The presented results in this chapter are based on testing the architecture in the context of multicasting adaptively-encoded MPEG4 video. For the multicast architecture:

- We do not define an active role for the receivers in our architecture other than deciding to join a multicast session.
- The architecture is based on the encoder's ability to produce video packets in two (or more) groups of packets that have different relevance to the decoding process at the receiver. In other words, some packets are very important to receive in order to be able to decode basic quality of the video, other packets (which are less important to receive) improve the video if they are included in the decoding process.
- The sender marks the packets with two (or more) different priorities² with basic information marked with high priority and enhancement information is marked with lower priority. Thus, during congestion basic quality can still be received. This results in emulating the layered approach (refer to Section 2.3) within one stream. This removes the burden of dealing with different layers

²In the rest of this chapter, we refer to "packet drop precedence levels" as "packet priorities" and we use the term "layer" to denote the packets marked with a certain "priority"

(multicast groups) at the receiver and ensuring that all packets will follow the same multicast tree.

- While congestion is developing, routers that run RIO (or WRED) will send back BECN messages to the sender with information on the priority level that caused the problem (i.e. which virtual queue).
- The sender runs a rate adaptation algorithm to search for an operating point (a sending rate for each priority level) that will reduce this feedback messages rate. The protocol stack at the sender is the same as that of Fig. 6.1. Discussion of the algorithm follows in detail in Section 7.3.
- **The major contribution of our architecture is how it deals with heterogeneity of receivers.** The sender will try to lower the sending rate for the high priority to a rate suitable for the slowest receiver. This will allow the receiver to get useful information in the time of persistent congestion while marking excess packets with lower priority to allow faster receivers higher utilization of their bandwidth.

The motivations/advantages of this work can be summarized in the following points:

1. Sending packets as one stream is much easier to handle than multiple streams and ensures that they all follow the same multicast tree.

Figure 7.2: End-to-end architecture

2. Sending all layers within one stream with different priorities over a network that supports priority dropping ensures that a minimum quality will be received in the time of network congestion.
3. Scalability issues is minimized when the feedback to the sender is provided from the routers rather than from receivers.
4. The required router functionality are already in use in today's routers or at least

have been proposed for use. So this architecture will not present the network with added complexity.

An illustration of the end-to-end architecture is shown in Fig. 7.2.

For this architecture to operate in unicast mode, the rate adaptation algorithm behaves differently at the lower priority by lowering the rate in the same fashion of the high priority level, because there is only one receiver to accommodate. In the rest of this chapter, the multicast case is discussed as it is the more complex one.

7.3 The rate adaptation algorithm

7.3.1 The rate adaptation equation

Assume that MPEG4 traffic is generated at the source and divided into L layers marked with L different priorities of the same AF class. All layers of the same source are sent in one multicast stream. Also we assume that this is the number of different priorities (and hence virtual queues) recognized at the routers for this AF class. Let $R_i(t)$, $1 \leq i \leq L$, be the rate (in packets/sec) of layer i at the source at time t .

We also consider

$$P_i(t) = P_i^{Max}(t) + P_i^{Send}(t) P_i^{MinMax}(t)$$

where $P_i(t)$ is the probability that virtual queue i will generate a feedback message

at time t . Also at time t we have

$$P_i^{Max}(t) = \text{Prob}\{\text{QueueSize}(i) \geq \text{max}_{th}\}$$

$$P_i^{MinMax}(t) = \text{Prob}\{\text{min}_{th} \leq \text{QueueSize}(i) \leq \text{max}_{th}\}$$

$$P_i^{Send}(t) = \text{Prob}\{\text{Send feedback message} \mid \text{min}_{th} \leq \text{QueueSize}(i) \leq \text{max}_{th}\}$$

We derived $P_i(t)$ from the specification of BECN (Backward ECN) [66].

In Appendix A, for the unicast case, we show the derivation of the relation between the rate (in packet/sec) of layer i at time t at the source, $R_i(t)$, the probability $P_i(t)$ and $N_i(t)$; $1 \leq i \leq L$, which is the rate (in packet/sec) of feedback messages generated by the router's virtual queue i at time t . The relation is represented by equation A.2:

$$R_i^{new} = R_i^{old} \left(1 - \frac{P_i^{new} - P_i^{old}}{P_i^{old}}\right) + a \frac{(N_i^{new} - N_i^{old})}{P_i^{old}}; \quad 0 \leq a \leq 1$$

where a represents the fraction of traffic $R_i(t)$ contributed to the total traffic coming into virtual router i .

This equation is very helpful in describing the relation between the sending rate and changing quantities $P_i(t)$ and $N_i(t)$ in the unicast case. Unfortunately, this relation does not hold in the multicast case since the values of P and N will be taken from different routers at different values of t which results in inconsistency between these values. This equation, however, lead us to deriving an approximate relation for the multicast case. Considering the changes from *old* to *new* values of $R_i(t)$ and $P_i(t)$ in a small interval Δt , we use the following equation to update the rate $R_i(t)$ in the

multicast case:

$$R_i^{new} = R_i^{old}(1 - \alpha_i \Delta P_i), \quad 0 < \alpha_i < 1 \quad (7.1)$$

where

$$\Delta P_i = P_i^{new} - P_i^{old}$$

The rationale behind using this equation is to always change $R_i(t)$ in the opposite direction of change of $P_i(t)$ with a step α_i . We change α_i to control how much $R_i(t)$ changes in reaction to changing network conditions. $|\Delta P_i|$ can assume values between 0 and 1. At these extreme values, changes in R_i^{new} can be either no change at all (0%) or very high change (100%).

Equation (7.1) is subject to the constraints:

$$\begin{aligned} R_i^{min} &\leq R_i \leq R_i^{max} \\ R^{min} &\leq \sum_{i=1}^L R_i \leq R^{max} \end{aligned}$$

where R_i^{min} and R_i^{max} are the value limits of the rate at layer i respectively and R^{min} and R^{max} are the limits for the total source rate. These values depend on the limitations imposed by the video encoder, its outgoing link speed, and the minimum accepted video quality at the receiver.

7.3.1.1 Selection of the value of α_i

We select $\alpha_i = C_i \sqrt{|\Delta P_i|}$ where C_i is a constant for layer i and where $0 < C_i < 1$.

The use of C_i sets the maximum rate change (when $|\Delta P_i| = 1$) to $C_i R_i^{old}$ at layer i .

We also need to relate the value of α_i , which is the step of change in the rate $R_i(t)$, to the value of $|\Delta P_i|$. If $|\Delta P_i|$ is very close to 1, the value of C_i will limit the change. On the other hand, when $|\Delta P_i|$ is very small (≤ 0.1), we still need to react with a tangible change of rate so that the algorithm will not get caught in a local minimum in its search of an optimum operating point.

The choice of square root function was motivated by our design goal of being able to react to very small changes of network conditions (when $|\Delta P_i| \leq 0.1$) as the square root of these small values is greater than the actual value ($\sqrt{x} > x$; $|x| \leq 1$). This helps to react to congestion while it is developing. Check parts (a) and (b) of Fig. 7.3 for different choices of α_i . Part (b) shows the significant advantage the square root choice has over other choices. We experimented with functions other than the square root but this one gives the smoothest rate changes.

Through simulations, we found that operating with values of C_i ranging from 0.05 to 0.25 keeps the system stable. Setting C_i to value greater than 0.25 may result in rate changes that are greater than 25% in one round-trip time which is not practical for a video application. A value of 0.1 at the high priority layer gave the best performance based on the criteria of (1) minimum packet loss ratio in the high priority layer and (2) matching of the source's rate to receivers' bandwidth capacities.

a) ΔP_i varies from 0 to 1 b) Enlargement of the $[0,0.1]$ intervalFigure 7.3: Selection of the value of α_i

7.3.2 Round-Trip Time (RTT)

Routers send feedback messages to the sender with values of P_i^{new} that indicate the congestion status of the routers. The sender will evaluate the feedback from all routers every Δt and decide on a new rate R_i^{new} . Routers calculation of P_i^{new} is presented in Section 7.3.4. The value of Δt will depend of the sender's estimation of the Round-Trip Time (RTT) from the routers that send the feedback information.

In their feedback, routers will echo the time stamp of the last packet they received from that sender. If a router has to send a feedback message, it will wait for the next packet to arrive and immediately send its feedback to this sender to help the sender accurately estimate the RTT.

We select the RTT value that corresponds to the router that has the worst situation at the high priority layer. That is the router with $Max(P_i^{new})$ in its feedback message.

7.3.3 Feedback suppression

The value of $P_i(t)$ will be estimated by routers and sent back to the sender. To reduce feedback, routers will send feedback messages with a probability instead of sending a feedback message for every packet that causes a problem. From simulations, we found that sending 2% to 5% of the feedback messages kept feedback volume reasonable and in the same time kept the sender responsive to changes in network conditions.

7.3.4 Calculation of probabilities

At each router, the quantities $P_i^{Max}(t)$ and $P_i^{MinMax}(t)$ are calculated using real-time measurements from the network rather than being based on an analytical model. The reason for this is that in the general case where all kinds of traffic flows are coming into the routers queues, it is very hard to assume a certain model for the input traffic.

Routers will estimate the probability of a certain event based on the fraction of time this event happened compared to the total time of operation. We bias the probability estimation by giving more weight to newer values to make the estimate a better representative of the current state of the network. Otherwise, after a long time of operation these values will converge to a constant value. We used the scheme used in [75] for measuring loss intervals. The probability is observed at each virtual queue i in k subsequent time intervals and give these intervals different weights w_i , $1 \leq i \leq k$. The length of the time interval is based on the speed of the link(s) incoming into the router queues. To calculate $P_i(t)$ (whether $P_i^{Max}(t)$ or $P_i^{MinMax}(t)$) at the end of an interval m we use:

$$P_i(m) = \frac{\sum_{j=1}^k w_j P_{m-j}}{\sum_{j=1}^k w_j} \quad (7.2)$$

Values of w_i are chosen so that recent probabilities have the same weight, while weights for older probabilities are smaller. The choice of a value for k is important. Large value of k will result in a smooth estimation of $P_i(m)$ while very large values of k will reduce the sender's responsiveness to changes in network conditions. For most of our simulations, we used $k = 10$, and $w = \{4, 4, 4, 4, 4, 2, 2, 2, 1, 1\}$. $P_i^{Send}(t)$

is calculated using the approach in [22]. It depends on the average queue size and on the RED parameters.

7.3.5 The algorithm to change R_i

Now we describe the algorithm used by the sender to calculate R_i^{new} every Δt (or RTT) seconds.

REPEAT every RTT

REPEAT for every layer i

If Nofeedback

increase R_i by 1%

else If ($\Delta P_i > 0$)

reduce R_i using Eqn. (7.1)

else If ($\Delta P_i < 0$)

If (i NOT highest priority layer)

increase R_i using Eqn. (7.1)

else If ($\Delta P_i = 0$)

If $P_i^{new} > 0.9$

reduce R_i by 3%

END REPEAT for every layer i

END REPEAT every RTT

Note that the values of 1% and 3% are chosen to conservatively increase/decrease

the rate as at this point we can not be sure exactly which direction ΔP_i is moving. Note also that the rate of the highest is increased only when there is no feedback to make sure that it is not increased beyond slow receivers capacities.

7.3.6 Changing the equation parameters

Routers calculate values of P_i^{new} and send them back to the sender that should select one of them as a new value of P_i^{new} every Δt seconds. There are different criteria that we consider for the sender to change $P_{i^{new}}$:

- At the highest priority layer ($i = 1$), the sender selects $MAX(P_1^{new})$ received during Δt . This results in accommodating of the router with the worst congestion situation. This is done subject to the constraint $R_i(t) \geq R_i^{min}$ to avoid the problem where a slow portion of the receivers drag the sending rate down dramatically³.
- At lower priority layers ($i > 1$), the sender selects $MIN(P_i^{new})$ received during Δt . This results in maximizing $R_i(t)$ at layer i subject to $R_i(t) \leq R_i^{max}$. The reason behind this is to let receivers with extra available bandwidth utilize their links. Slow receivers will have these packets dropped by their routers.

³This is known as the “drop to zero” problem.

7.4 Simulation setup

We carried out simulations using the network simulator *ns-2* [52] version 2.1b8a. Only one AF class is assumed with two drop precedence levels (priorities). In one experiment (Section 7.5.2), we compare the performance of our proposal in the case of using RIO-C, RIO-D, or WRED (refer to Section 7.1) while in all other experiments only one of those models is used. Queue parameters configuration (min_{th} and max_{th}) at the routers for these priority levels are the same in all experiments. The results we show are for simulations that are 300 seconds long. We use $C_i = 0.1$ (of Eqn. 7.1) for both priority levels ($i = 1, 2$) except for the TCP experiment where we use $C_1 = 0.25$ (Section 7.5.7). The simulation starts with equal rates at the two priority levels ($R_1(t) = R_2(t)$).

Our work in this chapter depends on the MPEG4 that we generate based on the method presented in Chapter 5 [27, 43, 49]. In Fig. 7.4 we show a an output trace of our MPEG4 generator where no control is applied to the source.

7.5 Discussion of simulation results

7.5.1 Basic test

In this basic test, the goal is to check how the sender will adjust its rate for the high priority packets to closely match the bandwidth available at the receivers. In this experiment, we only have one sender and one receiver (Fig. 7.5), so the sender

Figure 7.4: Output of the MPEG4 generator with no control at the source

should match its rate of the high priority to the bandwidth B with low packet loss. For each of the following values of B , 100Kbps, 200Kbps, 400Kbps, and 1Mbps, the simulation (of 300 seconds) is repeated and the average rate at each priority level is plotted against values of B in Fig. 7.6. We can see that the average rate at the high priority level is very close to the available bandwidth (i.e. 100Kbps, 200Kbps, etc.) while the low priority rate is a very small percentage of the available bandwidth. This experiment uses RIO-C for queueing.

Figure 7.5: Basic test topology

Figure 7.6: Result of the basic test with bandwidth B

7.5.2 Heterogeneity test

This experiment is a test of how the architecture (with its rate adaptation algorithm) deals with heterogeneity. We simulate the topology of Fig. 7.7 where one sender is multicasting MPEG4 encoded video to three receivers who have different bandwidths (100Kbps, 200Kbps, and 400Kbps) through one router. In addition to this goal, this experiment is repeated three times to compare the use of RIO-C, RIO-D, and WRED.

Figure 7.7: Heterogeneity test topology

- *Throughput of high priority layer:* The throughput is shown in Fig. 7.8. We can see in the figure that the algorithm tries to bring down the rate for the high priority layer (for the three receivers) to a rate less than a 100Kbps which is the rate of the slowest receiver. The difference between parts (a), (b), and (c) of the figure is due to the way each queuing mechanism protects its highest priority level. Note that in all simulations we used the same set of parameters for the two priority levels. In RIO-C, this layer gets all of the 100Kbps, while in RIO-D and WRED it gets a lower rate because the lower priority layer is more protected with these two mechanisms. Also in this figure we can see that highest priority layer is more aggressive with RIO-C. Both RIO-D and WRED offer more isolation and protection and RIO-D offers best stability at this layer.
- *Throughput of lower priority layer:* This is shown in Fig. 7.9. In this layer, the algorithm allows receivers with higher capacities to better utilize their bandwidth. The best performance comes with RIO-D because of its isolation of classes. It allowed each receiver its maximum share at this layer. Although this is not the best option of R1 as it does not make use of the low priority in this case with a high loss rate as seen in Fig. 7.12. WRED is better than RIO-C but still not as good as RIO-D. The reason of the lower rates of RIO-C at this layer is that RIO-C protects the high priority layer. This protection lets the high priority layer be aggressive and eventually brings down the rate down for the lower priority layer.

- *Total throughput:* Fig. 7.10 shows the summation of the throughput at both layers for each receiver. It is clear that maximum utilization is achieved with RIO-D. This however should not be taken as an absolute metric for judging the performance as we will comment in conclusion 7.7.
- *Packet loss at high priority layer:* In Fig. 7.11, we can see that the loss ratio at the high layer for R2 and R3 is zero all the time. However, for R1, while the loss ratio is high only in transient phase in the beginning of the simulations for RIO-D and WRED, it has high values at different times during the simulation for RIO-C. The reason is as we mentioned earlier that RIO-C favors the high priority layer and allows this layer to become aggressive. This results in overshoots of the rate at this layer causing these bursts high of losses as in part (a) of Fig. 7.12.
- *Packet loss at low priority layer:* The reader should refer to Fig. 7.12. It can be seen that RIO-C has the lowest loss rate at this layer for the three receivers because it does not allow the rate at this layer to go as high as the cases of RIO-D and WRED.
- *Feedback:* In Fig. 7.13, the rate of feedback messages in message/sec is shown. The results are consistent in the three cases where R3 generates the lowest feedback and R1 and R2 generate more feedback. Note that these messages have a very small size compared to the data sent (one message is 40 bytes).

For most of our simulations, the feedback measured at the source upstream is around 5% of the data delivered downstream to the receivers.

7.5.3 Test of bandwidth adaptation

In this experiment, we test how the rate adaptation algorithm adapts to changing network condition which in this case is the changing receiver membership of a multicast session. The topology of Fig. 7.14 is used to run the following scenario. R1 and R2 join the MPEG4 multicast session that is conducted by the Multicast Source. After a 100 seconds, R3 joins the session, and after 200 seconds, R4 joins the session as well. The bandwidth of each link is as shown in the figure.

Fig. 7.15 shows the throughput of each receiver at each priority level as well as the total throughput. In the interval $[0,100)$, R1 and R2 receive between 250Kbps and 300Kbps of high priority to accommodate R1 and R2 gets the rest of its 400Kbps (a little more than 100Kbps) in lower priority. Both R3 and R4 get nothing until R3 joins the session after 100 seconds which lowers the rate at the high priority to an average of 170Kbps to accommodate R3. Both R1 and R2 still utilize their bandwidth with more of the lower priority traffic. After 200 seconds, R4 joins the session, and again the sender adjusts the high priority rate to 80Kbps and allows the other receivers more lower priority to fully utilize their links.

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.8: High priority throughput

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.9: Low priority throughput

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.10: Total throughput

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.11: Packet loss in high priority

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.12: Packet loss in low priority

a) Using RIO-C

b) Using RIO-D

c) Using WRED

Figure 7.13: Rate of feedback messages

Figure 7.14: Topology of the bandwidth adaptation test

7.5.4 Scalability test

In this experiment, the topology of Fig. 7.16 is simulated for 300 seconds to test how the architecture handles a larger number of receivers. There is one sender and 60 receivers. Receiver 1 to 15 have a 100Kbps link each, receivers 16 to 30 have a 200Kbps link each, receivers 31 to 45 have a 300Kbps link each, and receivers 46 to 60 have a 400Kbps link each. The link between the sender and the router is 1Mbps. Based on the results from Section 7.5.2, we use RIO-D.

Fig. 7.17 shows a representative receiver of each receiver group. We repeated the simulations several times and found out that receivers with the same bandwidth have identical performance in term of the metrics we present here. R10 represents the 100Kbps receivers, R20 represents the 200Kbps receivers, R40 represents the 300Kbps receivers, and R50 represents the 400Kbps receivers. Parts (a), (b), and

a) Throughput of R1

b) Throughput of R2

c) Throughput of R3

d) Throughput of R4

Figure 7.15: Bandwidth adaptation to changing session membership

(c) show the expected rates for the four receivers. They are compliant with results from Section 7.5.2. Part (d) shows that all receivers have zero packet loss except for R10 that has an average loss of approximately 5%. Part (e) shows the differentiation between receivers in loss of the lower priority. Taking parts (a-e) into consideration we come to the realization that receivers get the same rate at the high priority layer while getting different rates with different packet loss ratios at the lower priority level. *Part (f) shows that the feedback from each receiver is much lower than the case of Fig 7.13. The aggregate of these receivers (the 100Kbps group) is close to that of R1 in Fig. 7.13. This experiment shows that the architecture scales well with an increasing number of receivers.*

7.5.5 Multisession interaction

This experiment shows how multiple multicast sessions using our architecture would share the bandwidth of a bottleneck link. In Fig. 7.18, three senders are multicasting to three receivers in three one-to-one sessions as shown in the figure. The three

Figure 7.16: Scalability test topology

Figure 7.17: Scalability test

sessions compete for the link between Router1 and Router2 which is slower than the aggregate of the three incoming links between the senders and Router1.

Figure 7.18: Topology for testing multisession interaction

Fig. 7.19 shows the allocation of bandwidth of the Router1-Router2 link between the three sessions. Part (a) shows that at the high priority level, they share the bandwidth of the link fairly with an average rate of 170Kbps each while at the lower priority level their rates tends quickly to zero.

7.5.6 Testing with other topologies

In this section, two more complex topologies are used to test the architecture and the rate adaptation algorithm. Fig. 7.20 shows the first topology. Three routers connect the video multicast source to the nine receivers with links that have the shown capacities. Throughput of the receivers at the high priority layer, the low priority layer, and the total throughput of each receiver is shown in Fig. 7.21. As in earlier experiments, all the receivers receive the same rate for the high layer which is around 80Kbps. Each receiver gets higher rate at the lower priority layer, each

Figure 7.19: Multisession interaction

according to its available bandwidth. The total is shown in part (c) of the figure.

Fig. 7.22 shows the same number of receivers with the same bandwidth as Fig. 7.20 with the difference in the topology of the connecting routers. In this experiment, the routers form a loop. As shown in Fig. 7.23, the receivers get the same throughput as the case of Fig. 7.21. This shows that the change in topology of the connecting routers did not have an effect neither on the performance of our architecture nor on the rate adaptation algorithm.

Figure 7.20: Arbitrary topology 1

7.5.7 Interaction with TCP

This final experiment tests the very important aspect of bandwidth sharing with TCP. Fig. 7.24 shows that we have one multicast session and one TCP session. We use Tahoe TCP and we assign TCP traffic to the high priority level. Part (a) of Fig. 7.25 shows that the multicast session gets a higher rate but by calculating the averages we find out that the average rate for TCP is 210Kbps while that of the multicast session is 280Kbps. This is a ratio of 3:4 which is acceptable given the fact that the multicast traffic represents a UDP-based multimedia application. Note that the video traffic shown here is all high priority since the competing TCP traffic pushes the lower priority video to reduce its rate to zero.

We also have to mention that in order to get this result, C_1 should be set to value near the high end of the range discussed in Section 7.3.1.

a) High priority throughput

b) Low priority throughput

c) Total throughput

Figure 7.21: Results for receivers in arbitrary topology 1

Figure 7.22: Arbitrary topology 2

Part (b) shows the difference of packet loss between the two competing flows. Finally part (c) shows that our architecture generates a relatively low feedback message rate given the fact that our feedback message has almost the same size as the TCP ACK. This is not provided for the sake of comparison as TCP is expected to generate more feedback because it acknowledges every packet for 100% reliable transmission. Part (c) is intended to show that the feedback generated by our architecture can be acceptable in an operational network.

Fig. 7.26 shows another test of interaction with TCP. In this case we test one video flow interacting with two TCP flows that start at different times. This experiment demonstrates more the interaction of TCP with each of the video priority levels. TCP packets are assigned to the high priority level.

Fig. 7.27 shows that the video source starts at the beginning of the simulation time. Quickly the rate adaptation algorithm manages to arrange the traffic to accommodate

a) High priority throughput

b) Low priority throughput

c) Total throughput

Figure 7.23: Results for receivers in arbitrary topology 2

Figure 7.24: Interaction with TCP - topology 1

R1 and R5 with a rate that is less than 200Kbps for the high priority layer (part (a) of the figure). The rest of the receivers obtain higher rates at the lower priority layer as shown in part (b). After a 100 seconds, 'TCP source1' starts to generate traffic destined to the 'TCP receiver' and after 200 seconds 'TCP source2' does the same.

In the time interval $[100,200]$, the TCP traffic from Source1 takes all the bandwidth that the lower priority video occupied. The rate adaptation algorithm utilizes all the available bandwidth and sends the video in high priority at a rate that is very close to 200Kbps. In the interval $[200,300]$, the second TCP source starts transmission. In this case, since all the traffic is now in high priority, the video source is more reluctant to share with the new TCP source, which shares the bandwidth allocated to the TCP source1 as shown in part (c) of the figure.

7.6 Design issues

In this section, we present some notes on our design that might have an impact on the implementation and deployment of our approach.

a) Throughput of the high priority

b) Packet loss in high priority

c) Feedback of TCP and Video

Figure 7.25: Results for interaction with TCP - topology 1

Figure 7.26: Interaction with TCP - topology 2

- The values of R_i^{min} , R_i^{max} , R^{min} , and R^{max} are not enforced in our simulations. So the ratio of R_1 to R_2 ⁴ may not be practical in our results but we do that intentionally in our simulations to test the control mechanism we propose without imposing limitations on the rates. Applying these limits to the methodology presented in this thesis will result in a better and more practical performance of our methodology.
- In our approach, we naturally assume that multicast receivers should at least have their bandwidth greater than R_1^{min} (minimum rate at the high priority layer).
- To accommodate higher heterogeneity of receivers bandwidth, a higher number

⁴the high priority layer is denoted by 0 and the lower layer is denoted by 1

a) High priority throughput video

b) Low priority throughput video

c) Total throughput video

d) TCP throughput

Figure 7.27: Results for Interaction with TCP - topology 2

of priority levels may be used. We note here that neither the architecture nor the rate adaptation algorithm needs to be changed in order to support a higher number of layers.

- We drop a low priority *multicast* packet at the input interface of a router if the virtual queue (of the output interface) that the packet is supposed to be forwarded to is experiencing a high loss rate. This keeps these queues operating with smaller average lengths and allows non-multicast traffic to get a better share of the queues at the routers' output interfaces. Note that the packet loss ratio we show in these graphs are the total of those dropped at the input interface and in the virtual queues.
- Our work is also applicable to the *unicast* case although we do not present these results here. The difference in the unicast mode is the consideration of $\max(P_i^{new})$ at all priority levels so that the rate at each layer meets what's available at the single receiver of the unicast flow. Also in unicast, we do not apply the dropping at the input interface described above.
- The architecture assumes that there will be loss at lower priority levels and hence we recommend the use of some Forward Error Correction (FEC) mechanism at the lower priority levels for applications that require a higher degree of reliability.
- The comparison between the three queuing mechanisms we considered reveals that selecting one of them is not a trivial task. While RIO-D and WRED result

in better utilization of bandwidth, they also result in high loss rate in the lower layer to a level that may render it undesirable (check loss rate of R2 at lower layer). On the other hand, RIO-C offers different qualities with lower loss rates at the expense of less bandwidth utilization.

7.7 Conclusion

This chapter presented an architecture and a rate adaptation algorithm for real-time video transport in Assured Forwarding networks. It can operate in either unicast or multicast modes and we presented the simulation results for the multicast case. We showed how it enables users with different bandwidth capabilities to receive the same video multicast in different qualities. This differentiation is based on encoding the video into two levels of priorities (more levels can be used without change in the methodology). We can see how the algorithm always attempts accommodating the slowest receiver at the high priority level and suitably increases the rate at the lower priority level. We also presented some tests using simulation of the performance of our architecture and the rate adaptation algorithm.

Chapter 8

Conclusions and Future Work

In this thesis, we have presented our work in the area of congestion control for video multicasting applications in IP networks. The thesis includes a survey and a presentation of background information on multicast congestion control. This starts with the pure end-to-end schemes. We presented the problems that are reported in the literature concerning the layered approach to congestion control of multicast video. This thesis added the study in Chapter 3 to the this list of problems. This study shows that sending the video in layers and handing the control to the receivers will definitely result in a performance deficiency. From this point, we advocated our way into building a network-supported congestion control scheme for video applications.

Our solution philosophy is to exploit future network architectures to provide some support to the multicast architecture we propose. The proposed architecture takes advantage of the priority-based architecture of Assured Forwarding style networks to

provide receivers with different levels of service according to their networking capabilities. This is achieved using a sophisticated rate adaptation algorithm that operates at the sender and uses feedback information from routers. The methodology we used for soliciting feedback is scalable as shown through simulation results. Also through simulation results we show several desired properties of our solution to the multicast congestion control problems such as scalability, dealing with heterogeneity, adaptation to changing network conditions, and interaction with TCP.

Our work can be extended for more development of multicast architectures in future QoS-aware networks such as Diffserv networks. Except for a few attempts (check [7, 71] and references within), the area of IP multicasting in a Diffserv network is severely under-investigated. There is a lot of open research points in the different areas such as the definition of end-to-end policies and SLA's (Service Level Agreements) for multicast traffic. Admission control to multicast sessions is an issue that needs special attention.

Appendix A

Derivation of The Rate Adaptation Equation for Unicast

In this appendix, we present the derivation of the rate adaptation equation for unicast applications.

Assume that MPEG4 traffic is generated at the source and divided into L layers with different L priorities. Also we assume that this is the number of different priorities (and hence priority queues) that RIO can recognize at the routers.

Let $R_i(t)$; $1 \leq i \leq L$, be the rate of layer i (in packet/sec) at the source at time t . At the router, let $N_i(t)$; $1 \leq i \leq L$, be the rate of feedback messages (in packet/sec) generated by the router's virtual queue i at time t . $N_i(t)$ that can be calculated using:

$$N_i(t) = \lambda_i(t) P_i(t) \tag{A.1}$$

where

$$P_i(t) = P_i^{Max}(t) + P_i^{Send} P_i^{MinMax}(t)$$

is the probability that queue i will generate a feedback message at time t and

$$\lambda_i(t) = \text{arrival rate at RIO virtual queue } i \text{ at time } t$$

$$P_i^{Max}(t) = \text{Prob}\{\text{QueueSize}(i) \geq \text{max}\} \text{ at time } t$$

$$P_i^{MinMax}(t) = \text{Prob}\{\text{min} \leq \text{QueueSize}(i) \leq \text{max}\} \text{ at time } t$$

$$P_i^{Send} = \text{Prob}\{\text{Send a BECN message} \mid \text{min} \leq \text{QueueSize}(i) \leq \text{max}\}$$

Check Fig. A.1 for an illustration.

Figure A.1: Illustration of the rate equation parameters

differentiating both sides of A.1 we get

$$\frac{dN_i(t)}{dt} = P_i(t) \frac{d\lambda_i(t)}{dt} + \lambda_i(t) \frac{dP_i(t)}{dt}$$

by approximation in an interval Δt , this turns to

$$\frac{\Delta N_i}{\Delta t} = P_i^{old} \frac{\Delta \lambda_i}{\Delta t} + \lambda_i^{old} \frac{\Delta P_i}{\Delta t}$$

where $\Delta N_i = N_i^{new} - N_i^{old}$, $\Delta \lambda_i = \lambda_i^{new} - \lambda_i^{old}$, and $\Delta P_i = P_i^{new} - P_i^{old}$.

We can relate $R_i(t)$ to $\lambda_i(t)$ using

$$R_i(t) = a \lambda_i(t); \quad 0 \leq a \leq 1$$

since $R_i(t)$ contributes to $\lambda_i(t)$ only a portion that is equal to a , it receives only a of the feedback $N_i(t)$. We substitute that in A.1 and we get

$$R_i^{new} = R_i^{old} \left(1 - \frac{P_i^{new} - P_i^{old}}{P_i^{old}}\right) + a \frac{(N_i^{new} - N_i^{old})}{P_i^{old}} \quad (\text{A.2})$$

This equation is subject to the constraints:

$$R_i^{min} \leq R_i \leq R_i^{max}$$

$$R^{min} \leq \sum_{i=1}^L R_i \leq R^{max}$$

The values of R_i^{min} , R_i^{max} , R^{min} , and R^{max} depend on the video encoder and on the outgoing link speed.

Since the expression $P_i = P_i^{Max} + P_i^{Send} P_i^{MinMax}$ represents the probability of sending back a feedback message, it may assume the value *zero* if the queue size is less than *min*. This is not going to affect the calculations since in this case a feedback message will not be sent.

Bibliography

- [1] MPEG Pointers and Resources. *<http://www.mpeg.org>*.
- [2] The Internet Engineering Task Force. *<http://www.ietf.org>*.
- [3] The web page of ISO/IEC Moving Picture Experts Group.
<http://mpeg.telecomitalialab.com>.
- [4] Advanced QoS Services for the Intelligent Internet. *Cisco White Paper*, 1997.
- [5] K. C. Almeroth and M. Ammar. Multicast Group Behavior in the Internet's Multicast Backbone (MBONE). *IEEE Communications Magazine*, June 1997.
- [6] A. Basu and S. J. Golestani. Architectural Issues for Multicast Congestion Control. *in Proc. of NOSSDAV*, 1999.
- [7] R. Bless and K. Wehrle. Group Communication in Differentiated Services Networks. *Internet QoS for the Global Computing, Workshop at CCGRID*, 2001.
- [8] J-C. Bolot, T. Turletti, and I. Wakeman. Scalable Feedback Control for Multicast Video Distribution in the Internet. *in Proc. of ACM SIGCOMM*, October 1994.

- [9] Y. Chawathe, S. A. Fink, S. McCanne, and E. Brewer. A Proxy Architecture for Reliable Multicast in Heterogeneous Environment. *In Proc. of ACM Multimedia*, 1998.
- [10] D. Chiu and R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, pages 17:1–14, 1989.
- [11] Y. Chu, S. Rao, and H. Zhang. A Case for End Systems Multicast. *In Proc. of ACM SIGMETRICS*, 2000.
- [12] D. Clark and W. Fang. Explicit Allocation of Best Effort Packet Delivery Service. *IEEE/ACM Trans. on Networking*, 6(4):362–373, August 1998.
- [13] A. Clerget. TUF: A Tag-based UDP Multicast Flow Control Protocol. *INRIA Technical-report 3728*, July 1999.
- [14] A. Darcinschi and S. Fdida. Efficient Congestion Avoidance Mechanism. *In Proc. of IEEE LCN*, November 2000.
- [15] D. DeLucia and K. Obraczka. Multicast Feedback Suppression using Representatives. *in Proc. of IEEE INFOCOM*, 1997.
- [16] C. Diot et al. Deployment Issues for the IP Multicast Services and Architecture. *IEEE Network*, January/February 2000.

- [17] H. Eriksson. MBONE: The Multicast Backbone. *Communications of The ACM*, pages 54–60, August 1994.
- [18] W. Fenner. Internet Group Management Protocol, Version 2. *RFC 1112*.
- [19] S. Floyd. TCP and Explicit Congestion Notification. *ACM Computer Communications Review*, pages 10–23, October 1994.
- [20] S. Floyd and K. Fall. Promoting the Use of End-to-End Congestion Control in the Internet. *IEEE/ACM Trans. on Networking*, August 1999.
- [21] S. Floyd, M. Handley, and J. Padhye and J. Widmer. Equation-based Congestion Control for Unicast Applications. *in Proc. of ACM SIGCOMM*, pages 43–56, August 2000.
- [22] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Trans. on Networking*, pages 397–413, August 1993.
- [23] B. Girod. Video Communications - Advances and Applications. *in Proc. of GLOBECOM*, 1999.
- [24] R. Gopalakrishnan et al. Stability and Fairness Issues in Layered Multicast. *In Proc. of NOSSDAV*, 1999.
- [25] R. Gopalakrishnan et al. A Simple Loss Differentiation Approach to Layered Multicast. *In Proc. of IEEE INFOCOM*, 2000.

- [26] S. Gorinskey and H. Vin. The Utility of Feedback in Layered Multicast Congestion Control. *in Proc. of NOSSDAV*, June 2001.
- [27] R. M. Guirguis and S. Mahmoud. Transmission of Real-time Multi-layered MPEG-4 Video over ATM/ABR Service. *In Proc. of IEEE ICC*, 2000.
- [28] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. *RFC 2597*, June 1999.
- [29] M. R. Ismail, I. E. Iambadaris, M. Devetsikiotis, and A. R. Kaye. Modeling Prioritized MPEG Video using TES and a Frame Spreading Strategy for Transmission in ATM Networks. *In Proc. of INFOCOM*, 1995.
- [30] D. Towsley, J. Padhye, J. Kurose and R. Koodli. A Model Based TCP-Friendly Control Protocol. *in Proc. of NOSSDAV*, 1999.
- [31] T. Kim and M. H. Ammar. A Comparison of Layering and Stream Replication Video Multicast Schemes. *In Proc. of NOSSDAV*, 2001.
- [32] R. Koenen. MPEG4 Overview. *In IEEE Spectrum*, February 1999.
- [33] A. Legout and E. Biersack. Beyond TCP-friendliness: A new paradigm for end-to-end congestion control. *Technical Report, Institut Eurocom*, <http://www.eurocom.fr/legout/Research/research.html>, November 1999.
- [34] A. Legout and E. Biersack. Pathological Behaviors for RLM and RLC. *In Proc. of NOSSDAV*, 2000.

- [35] A. Legout and E. Biersack. PLM: Fast Convergence for Cumulative Layered Multicast Transmission Schemes. *In Proc. of SIGMETRICS*, pages 13–22, 2000.
- [36] X. Li, S. Paul, and M. Ammar. Layered Video Multicast with Retransmission (LVMR): Evaluation of Hierarchical Rate Control. *in Proc. of INFOCOM*, pages 1062–1072, 1998.
- [37] X. Li, S. Paul, P. Pancha, and M. Ammar. Layered Video Multicast with Retransmission (LVMR): Evaluation of Error Recovery Schemes. *in Proc. of NOSSDAV*, 1997.
- [38] M. Luby, L. Vicisano, and T. Speakman. Heterogeneous Multicast Congestion Control based on Router Packet Filtering. *Presentation at RMRG meeting, Pisa, Italy*, June 1999.
- [39] J. Mahdavi and S. Floyd. TCP-Friendly Unicast Rate-Based Flow Control. *submitted to end-2-end mailing list*, January 1997.
- [40] R. Makkar. QoS Control for IP Networks: Buffer Management and Backward Explicit Congestion Notification (BECN). *M.Eng. thesis, Carleton University, Ottawa, Canada*, December 2000.
- [41] A. Matrawy and I. Lambadaris. A Rate Adaptation Algorithm for Multicast Sources in Priority-based IP Networks. *accepted for publication in IEEE Communications Letters*.

- [42] A. Matrawy and I. Lambadaris. A Real-time Video Multicast Architecture for Assured Forwarding Services. *accepted for publication in IEEE Transactions on Multimedia*.
- [43] A. Matrawy and I. Lambadaris. MPEG4 generator code for ns. *A link can be found at <http://www.isi.edu/nsnam/ns/ns-contributed.html>*.
- [44] A. Matrawy and I. Lambadaris. RLM Interaction with TCP: Congestion Control and Fairness Issues. *Carleton University Technical Report SCE-02-05*, December 2001.
- [45] A. Matrawy and I. Lambadaris. Simulation Results for Multicast Video for Assured Forwarding Networks. *Carleton University Technical Report SCE-02-11*, October 2002.
- [46] A. Matrawy and I. Lambadaris. Real-time Transport for Assured Forwarding: An Architecture for both Unicast and Multicast Applications. *To appear in Proc. of IEEE ICC*, 2003.
- [47] A. Matrawy, I. Lambadaris, and C. Huang. On Layered Video Fairness on IP Networks. *In Proc. of IEEE GLOBECOM*, 2001.
- [48] A. Matrawy, I. Lambadaris, and C. Huang. Comparison of the Use of Different ECN Techniques for IP Multicast Congestion Control. *In Proc. of IEEE ECUMN*, 2002.

- [49] A. Matrawy, I. Lambadaris, and C. Huang. MPEG4 Traffic Modeling using The Transform Expand Sample Methodology. *In Proc. of 4th IEEE International Workshop on Networked Appliances*, January 2002.
- [50] A. Matrawy, I. Lambadaris, and C. Huang. Multicasting of Adaptively-encoded MPEG4 over QoS-aware IP Networks. *In Proc. of IEEE ICC*, 2002.
- [51] S. McCanne. Scalable Compression and Transmission of Internet Multicast Video. *Ph.D. Thesis, University of California-Berkeley*, December 1996.
- [52] S. McCanne and S. Floyd. ns Network Simulator. <http://www.isi.edu/nsnam/ns/>.
- [53] S. McCanne and V. Jacobson. Receiver-driven Layered Multicast. *in Proc. of ACM SIGCOMM*, pages 117–130, September 1996.
- [54] B. Melamed. An Overview of TES Processes and Modeling Methodology. *In Performance Evaluation of Computer and Communication Systems*, 1993.
- [55] B. Melamed et al. TES-Based Modeling for Performance Evaluation of Integrated Networks. *In Proc. of INFOCOM*, 1992.
- [56] J. Nagle. Congestion Control in IP/TCP Internetworks. *RFC 896, IETF*, January 1984.
- [57] K. Nakauchi, H. Morikawa, and T. Aoyama. A Network-Supported Approach to Layered Multicast. *In Proc. of IEEE ICC*, 2001.

- [58] K. Obraczka. Multicast Transport Protocols: A Survey and Taxonomy. *IEEE Communications Magazine*, January 1998.
- [59] S. Paul. *Multicasting On The Internet And Its Applications*. Kluwer Academic Publishers, 1998.
- [60] J. Postel. Internet Control Message Protocol. *RFC 792*.
- [61] J. Postel. Transport Control Protocol. *RFC 793*.
- [62] J. Postel. User Datagram Protocol. *RFC 768*.
- [63] D. Reininger et al. Variable Bit Rate MPEG video: Characteristics, Modeling and Multiplexing. *In Proc. of ITC*, 1994.
- [64] L. Rizzo. pgmcc: A TCP-friendly Single-rate Multicast Congestion Control Scheme. *In Proc. of SIGCOMM*, 2000.
- [65] M. Ammar S. Y. Cheung and X. Li. On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution. *Tech-Report: GIT-CC-95-25*, Georgia Institute of Technology, July 1995.
- [66] J. Hadi Salim, B. Nandy, and N. Seddigh. A Proposal for Backward ECN for the Internet Protocol (IPv4/IPv6). *Internet Draft, draft-salim-jhsbnns-ecn-00.txt*.
- [67] N. Shacham. Multipoint Communications By Hierarchically Encoded Data. *in Proc. of INFOCOM*, 1992.