Centro de Informática
U·F·P·E

Pós-Graduação em Ciência da Computação

"An Adaptive-Predictive Architecture for Streaming Scalable Encoded Video"

by

Stênio Flávio de Lacerda Fernandes

PhD Thesis

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, April/2006

Universidade Federal de Pernambuco - UFPE
Centro de Informática - CIn

An Adaptive-Predictive Architecture for Streaming Scalable Encoded Video

by
Stênio Flávio de Lacerda Fernandes

A Thesis Proposal submitted to the Graduate Faculty of the Centro de Informática of the Universidade Federal de Pernambuco in Partial Fulfillment of the requirements for the degree of DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

Advisor: Prof. Djamel Fawzi Hadj Sadok, PhD
Co-Advisor: Prof. Ahmed Karmouch,
University of Ottawa, Canada

Recife, April/2006

Fernandes, Stênio Flávio de Lacerda
An adaptive-predictive architecture for streaming Scalable Encoded Video / Stênio Flávio de Lacerda Fernandes. – Recife : O Autor, 2006.
xx, 191 folhas : il., fig., tab.

Tese (doutorado) – Universidade Federal de Pernambuco. Cin. Ciência da Computação, 2006.

Inclui bibliografia e apêndices.

1. Ciência da computação – Redes de computadores. 2. Transmissão de vídeo na Internet – Controle de congestionamento – Scalable Encoded Vídeo. 3. Técnicas estatísticas – Análise das séries temporais – Predição. 4. Servidor de vídeo – Arquitetura adaptativa. I. Título.

004.72	CDU (2.ed.)	UFPE
004.65	CDD (22.ed.)	BC2006-370

Tese de Doutorado apresentada por **Stênio Flávio de Lacerda Fernandes** a Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"An Adaptive-Predictive Architecture for Streaming Scalable Encoded Video"**, orientada pelo **Prof. Djamel Fawzi Hadj Sadok** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Aluízio Fausto Ribeiro Araújo
Departamento de Sistemas de Computação - CIn / UFPE

Prof. Paulo Romero Martins Maciel
Departamento de Sistemas de Computação – CIn / UFPE

Profa. Judith Kelner
Departamento de Informação e Sistemas – CIn / UFPE

Prof. Ahmed Karmouch
Universidade de Ottawa - CA

Prof. Edmundo Roberto Mauro Madeira
Instituto de Computação / UNICAMP

Visto e permitida a impressão.
Recife, 17 de abril de 2006

Prof. FRANCISCO DE ASSIS TENÓRIO DE CARVALHO
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Dedication

To God, for His help when I was passing through difficult times
"God is our refuge and strength, a very present help in time of trouble." (Psalms 46:1)

To the memory of my father
José de Lucena Fernandes

To my mother
Maria da Penha de Lacerda Fernandes

To my wife Nina, for her love, encouragement and prayers for my success
"I've got you under my skin"

To my son Victor and to my daughter Alice, for their love and patience
"I've found a reason for me… and the reason is you!"

Acknowledgments

I am sincerely grateful to my advisor Prof. Djamel Sadok for giving me the privilege and honor to work with him over the last four years. I am also very fortunate to have Prof. Ahmed Karmouch as my co-advisor. Without their support, insightful advice, admirable judgment, and their demand for finest research, this thesis would not be possible. I am also particularly grateful to the committee members, specifically Prof. Judith Kelner (UFPE), Prof. Edmundo Madeira (UNICAMP), Prof. Aluísio Araújo (UFPE), and Prof. Paulo Maciel (UFPE) for their insightful comments and encouragement. They helped me to refine my work, thus providing a direct contribution to this thesis.

As a member of the Networking and Telecommunications Research Group (GPRT) at UFPE, I had the pleasure to work with several outstanding researchers, graduate and undergraduate students, who made my PhD years exciting and gratifying. It would be impossible to mention them all here. I am especially grateful to Prof. Carlos Kamienski (CEFET-SP) for being an awesome friend and colleague. I would also like to express my appreciation to my friend Prof. Denio Mariz (CEFET-PB) for his calm encouragement, and to Prof. Kelvin Dias (UFPA) and Dr. Dave Cavalcanti (Phillips Research) for frequent technical discussions. The office staff members of our research group have been wonderful. I especially want to thank Manuela “Manu” Melo for her patience and caring of all graduate students.

I express my gratitude to my wife Nina for taking good care of me, especially in the past four years. I extend appreciation to my kids for remarkable moments in Recife (Sunday morning soccer) and in Ottawa (tobogganing, ice skating, skiing, and winter walking). I would like to thank my parents for teaching me the value of education.

Finally, I am grateful for the funding support from the following agencies: CNPq (141553/2002-2) and CAPES (BEX - 0016/04-7).

Summary

CHAPTER 1. INTRODUCTION 1

1.1 RESEARCH PROBLEM 1

1.2 RESEARCH SCOPE 6

1.3 OVERVIEW OF THE PROPOSED SOLUTION..7

1.4 OUTLINE OF THE THESIS PROPOSAL 8

CHAPTER 2. BACKGROUND AND RELATED WORK: CONGESTION CONTROL AND MULTIMEDIA STREAMING IN THE INTERNET 9

2.1. CURRENT CONGESTION CONTROL MECHANISMS IN THE INTERNET 10

2.2. NEW APPROACHES FOR CONGESTION CONTROL MECHANISMS IN THE INTERNET.......18

2.3. SCHEMES FOR MULTIMEDIA STREAMING OVER THE INTERNET 28

2.4. INTEGRATED DESIGN FOR CONGESTION AND RATE CONTROL OF MULTIMEDIA STREAMS 34

CHAPTER 3. PERFORMANCE ANALYSIS OF STREAMING MULTIMEDIA FLOWS WITH EXPLICIT FEEDBACK NOTIFICATION 35

3.1. RELATED WORK 36

3.2. PERFORMANCE EVALUATION SCENARIO..40

3.3. VIDEO STREAMING WITH TFRC 46

3.4. VIDEO STREAMING WITH EXPLICIT FEEDBACK NOTIFICATION 58

3.5. GENERAL REMARKS 64

CHAPTER 4. AN ARCHITECTURE FOR STREAMING SCALABLE ENCODED VIDEO 68

4.1. RELATED WORK 69

4.2. DEFINITIONS AND NOTATION..73

4.3. DESIGN RATIONALE..74

4.4. SYSTEM COMPONENTS..76

4.5. PERFORMANCE ANALYSIS..95

CHAPTER 5. FORECAST AND CONTROL OF SCALABLE VIDEO STREAMING..... 118

5.1. SIGNAL PROCESSING TECHNIQUES FOR NETWORK TRAFFIC ANALYSIS 119

5.2. TECHNIQUES FOR MULTISCALE ANALYSIS AND FORECASTING................................120

5.3. DESIGN RATIONALE..128

5.4. THE PREDICTION UNIT..131

5.5. THE DECISION UNIT - DU 133

5.6. SIMULATION RESULTS 138

5.7. CONCLUDING REMARKS 149

CHAPTER 6. CONCLUDING REMARKS 152

6.1. SUMMARY OF THESIS RESEARCH AND CONTRIBUTIONS 152

6.2. FUTURE WORK 155

APPENDIX 1 - FILTERS IN TIME DOMAIN 163

APPENDIX 2 - DIGITAL VIDEO AND THE MPEG-4 FGS STANDARD 166

SCALABLE VIDEO ENCODING 167

MPEG-4 FGS VIDEO CODING METHOD 168

REFERENCES 171

List of Figures

Figure 1 - Forecasts for Video delivery services 2
Figure 2 - Infrastructure for delivering video services in different formats 3
Figure 3 - Digital Item Adaptation Architecture (from the reference [24]) 7
Figure 4 - Convergence of CADPC 27
Figure 5 – Simulated Network: a Dumbbell Topology 41
Figure 6 - Original Video Signal 46
Figure 7 - TFRC#1 and TFRC#3 48
Figure 8 - ECDF - TFRC#1 and TFRC#3 49
Figure 9 - TFRC#2 and TFRC#4 50
Figure 10 - ECDF - TFRC#2 and TFRC#4 50
Figure 11 - TFRC#5 and TFRC#7 51
Figure 12 - ECDF - TFRC#5 and TFRC#7 51
Figure 13 - TFRC#6 and TFRC#8 52
Figure 14 - ECDF - TFRC#6 and TFRC#8 53
Figure 15 - TFRC#9 and TFRC#11 54
Figure 16 - ECDF - TFRC#9 and TFRC#11 54
Figure 17 - TFRC#10 and TFRC#12 55
Figure 18 - ECDF - TFRC#10 and TFRC#12 55
Figure 19 - TFRC#13 and TFRC#15 56
Figure 20 - ECDF - TFRC#13 and TFRC#15 56
Figure 21 - TFRC#14 and TFRC#16 57
Figure 22 - ECDF - TFRC#14 and TFRC#16 57
Figure 23 - CADPC#7 and CADPC#8 60
Figure 24 - ECDF - CADPC#7 and CADPC#8 61
Figure 25 - CADPC#19 and CADPC#20 62
Figure 26 - ECDF - CADPC#19 and CADPC#20 62
Figure 27 - CADPC#23 and CADPC#24 63
Figure 28 - ECDF - CADPC#23 and CADPC#24 63
Figure 29 - Combination of BL and EL 74
Figure 30 - Adaptive-Predictive Architecture for Video Streaming 76
Figure 31 - The Adaptive-Predictive Unit 77

Figure 32 – Filter Unit (DLPF) 78
Figure 33 - Kesten's Rule Simulation 82
Figure 34 - Mirozahmedov’s Rule Simulation 83
Figure 35 - Gaivoronski's rule Simulation 84
Figure 36 - Trigg & Leach's rule simulation 85
Figure 37 - Whybark's rule simulation 86
Figure 38 - Dennis' rule simulation 87
Figure 39 - Smoothing Spline simulation 89
Figure 40 - LOWESS simulation 90
Figure 41 - Tukey's Smoothing simulation 91
Figure 42 – Causal FIR Filter simulation 93
Figure 43 - STES simulation 95
Figure 44 - BL and EL - Movie: The Firm 98
Figure 45 - BL and EL - Movie: Star Wars 98
Figure 46 - BL and EL - Movie: Oprah 98
Figure 47 - BL and EL - Movie: Toy Story 98
Figure 48 - Histogram for B Frames - BL 99
Figure 49 - Histogram for P Frames - BL 99
Figure 50 - Histogram for I Frames - BL 99
Figure 51 - Histogram for Frames - BL (Average) 99
Figure 52 - Histogram for B Frames - EL (2Mbps) 100
Figure 53 - Histogram for I Frames - EL (2Mbps) 100
Figure 54 - Histogram for P Frames - EL (2Mbps) 100
Figure 55 - Histogram for - EL (2Mbps) - Average 100
Figure 56 - Smoothed Traffic - DLPF: Trigg & Leach 103
Figure 57- Smoothed Traffic - DLPF: Gaivoronski 103
Figure 58- Smoothed Traffic - DLPF: Kesten 103
Figure 59- Smoothed Traffic - DLPF: Whybark 103
Figure 60- Smoothed Traffic - DLPF: Tukey Smoothing 103
Figure 61 – Comparison of the Mean Quality (BL) : ABR x Trigg – The Firm 104
Figure 62 – Comparison of the Mean Quality (EL) : ABR x Trigg – The Firm 104
Figure 63 – Relative Difference for the Mean Quality: (ABR x Trigg) to BL 105
Figure 64 – Relative Difference for the Mean Quality: (ABR x Trigg) to EL 105
Figure 65 – Empirical Cumulative Distribution Function: DLPF (Trigg) x ABR x BL 105

Figure 66 – Empirical Cumulative Distribution Function: DLPF (Trigg) x ABR x EL............105
Figure 67 - Notched Boxplots - Mean Quality: ABR x DLPF (Trigg & Leach).......................107
Figure 68 - Notched Boxplots - Coefficient of Variation: ABR x DLPF (Trigg & Leach) 107
Figure 69 - Notched Boxplots - Mean Quality: ABR x DLPF (Gaivoronski)..........................107
Figure 70- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Gaivoronski) 107
Figure 71 - Notched Boxplots - Mean Quality: ABR x DLPF (Kesten) 108
Figure 72- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Kesten)...................108
Figure 73 - Notched Boxplots - Mean Quality: ABR x DLPF (Whybark)...............................109
Figure 74- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Whybark) 109
Figure 75 - Notched Boxplots - Mean Quality: ABR x DLPF (Dennis) 109
Figure 76- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Dennis)...................109
Figure 77 - Notched Boxplots - Mean Quality: ABR x DLPF (Tukey's Smoothing) 110
Figure 78- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Tukey's Smoothing)110
Figure 79 - Notched Boxplots - Mean Quality: ABR x DLPF (STES) 110
Figure 80 - Notched Boxplots - Standard Deviation: ABR x DLPF (STES) 110
Figure 81- Notched Boxplots - Mean Quality: ABR x DLPF (Dennis) 111
Figure 82- Notched Boxplots - Standard Deviation: ABR x DLPF (Dennis) 111
Figure 83- Notched Boxplots - Mean Quality: ABR x DLPF (FIR) 112
Figure 84- Notched Boxplots – Standard Deviation: ABR x DLPF (FIR)..............................112
Figure 85 - Performance Comparison of DLPF algorithms - Star Wars 112
Figure 86- Notched Boxplots – Standard Deviation: ABR x DLPF (STES)............................113
Figure 87- Notched Boxplots – Standard Deviation: ABR x DLPF (Dennis)..........................114
Figure 88- Notched Boxplots – Standard Deviation: ABR x DLPF (FIR)..............................114
Figure 89- Performance Comparison of DLPF algorithms - Oprah 115
Figure 90 - Notched Boxplots – Standard Deviation: ABR x DLPF (STES)..........................116
Figure 91 - Notched Boxplots – Standard Deviation: ABR x DLPF (Dennis)........................116
Figure 92 - Notched Boxplots – Standard Deviation: ABR x DLPF (FIR).............................117
Figure 93- Performance Comparison of DLPF algorithms - Toy Story 117
Figure 94 - Wavelet Filters. (a) Haar Filter (b) Daubechies 123
Figure 95 - Simulated Available Bandwidth from Heavy Tail distributions.............................124
Figure 96 - Energy Distribution over Different Scales – Definition #1 125
Figure 97 - Energy Distribution over Different Scales – Definition #2 126
Figure 98 - Noise level for Uniform, Poisson and Gamma PDF at several scales - Log2 (Energy) 126

Figure 99 - The Adaptive-Predictive Unit 129
Figure 100 - Filter Unit (DLPF) - Redesigned.. 131
Figure 101 - The Functional Block Prediction Unit 132
Figure 102 - Functional Description of the Prediction Unit 133
Figure 103 - The control decision functional block - DU.. 134
Figure 104 - Finite State Machine for the Control Decision functional block 136
Figure 105 - Prediction Error (MAPE) Evaluation after Filtering: Overall result..................... 140
Figure 106 - Prediction Error (MPE) Evaluation after Filtering: Overall result........................ 140
Figure 107 - Prediction Error (MPE) Profile: Aggregation at 300 samples 141
Figure 108 - Prediction Error (MPE) Profile: Aggregation at 3000 samples 141
Figure 109 - Prediction Error (MPE) Profile: Aggregation at 6000 samples 141
Figure 110 - Prediction Error (MPE) Profile: Aggregation at 9000 samples 141
Figure 111 - Prediction Error (MAPE) Profile: Aggregation at 300 samples 142
Figure 112 - Prediction Error (MAPE) Profile: Aggregation at 3000 samples 142
Figure 113 - Prediction Error (MAPE) Profile: Aggregation at 6000 samples 142
Figure 114 - Prediction Error (MAPE) Profile: Aggregation at 9000 samples 142
Figure 115- Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – The Firm 143
Figure 116 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – The Firm 143
Figure 117 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – The Firm 144
Figure 118 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – The Firm 144
Figure 119 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – The Firm 144
Figure 120 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – The Firm 144
Figure 121 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Oprah.. 145
Figure 122 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Oprah.. 145
Figure 123 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Oprah.. 145

Figure 124 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Oprah 145

Figure 125 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Oprah 146

Figure 126 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Oprah 146

Figure 127 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Star Wars 146

Figure 128 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Star Wars 146

Figure 129 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Star Wars 147

Figure 130 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Star Wars 147

Figure 131 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Star Wars 147

Figure 132 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Star Wars 147

Figure 133 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Toy Story 148

Figure 134 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Toy Story 148

Figure 135 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Toy Story 148

Figure 136 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Toy Story 148

Figure 137 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Toy Story 148

Figure 138 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Toy Story 148

Figure 139 - Number of Changes in states (200Kbps, frame-based) according to PU Thresholds 150

Figure 140 - BarPlot: Number of Changes in States (200Kbps, frame-based) according to PU Thresholds 151

Figure 141 - Number of Changes in states (logscale, frame-based) according to PU Thresholds 151
Figure 142 - A general causal FIR filter 164
Figure 143 - Typical MPEG Group of Pictures (GoP) - From Reference [198] 167
Figure 144 - Example of Temporal Scalable Encoding (From Reference [198]) 167
Figure 145 - Example of Spatial Scalable Encoding (From Reference [198]) 168
Figure 146 - FGS structure at the encoder 169
Figure 147 - FGS structure at the streaming server 169
Figure 148 - FGS structure at the decoder (expected) 170

List of Tables

Table 1 – List of Scenarios: TFRC parameterization 47

Table 2 - CADPC parameterization and labels 58

Table 3 - Adaptive Step-Size Rules Available at the DLPF 80

Table 4 - Nomenclature for the DLPF algorithms 102

Abstract

The steadily growth of multimedia demands in the Internet can lead to an impending collapse, since there are little efforts in such applications to control their sending rates. In addition to that, it is well known that providing perceptually good quality video streaming is a complex task, in view of the fact that in today's best effort Internet the available bandwidth can fluctuate strongly and the encoded video can exhibit significant rate variability at several time-scales. On the other hand, one important requirement for streaming multimedia flows is that they must exhibit fairness with competing flows. Therefore, the main research problem addressed in this thesis is to bridge the gap between the available bandwidth variability and the encoded video rate variability, taking into account the requirement of the minimization of the quality variability and the maximization of the overall quality of the video rendered to the user.

The main contribution of this thesis is the definition and realization of a novel architecture for video streaming applications in best effort networking environments. We focus on a scenario where the network provides explicit feedback information throughout the network path, which implies that such multimedia streaming must be able to adapt to network conditions efficiently, i.e., it is capable to cope with variations in bandwidth at several time scales. Towards this end, within our scalable architecture we propose several deployable server-side based solutions, which combine most beneficial properties of some innovative congestion control mechanisms, signal processing techniques, and time series analysis. We devise and investigate the mechanisms that implement the proposed solutions, and reveal the efficiency of each approach through simulation.

Specifically, we firstly present a comprehensive investigation of the performance of video streaming in the best-effort Internet when using some selected network friendly protocols. As our experiments show, congestion control mechanisms that rely on precise explicit feedback information from the network provide a significantly better quality (i.e., with low intensity variation in quality) to the end-user than those that rely on rate-based slowly responsive ones. Second, using MPEG-4 Fine Granular Scalable pre-encoded video as our target application, we ensure that we meet the most important requirement from the network point of view that is transporting multimedia flows efficiently while exhibiting fairness with competing flows. Our architecture extracts the most precise information from the network level and then provides video source application with consistent and stable information.

In summary, we build our solution using appropriate techniques in the networking, signal processing and statistical fields. By merging ideas from several areas, we propose a scalable architecture for video streaming over best effort networks with explicit rate notification. Such novel architecture is flexible to extend, subtract, or change functionalities.

Resumo

O aumento contínuo das demandas por serviços multimídia na Internet, observado nos últimos anos, pode levar a um iminente colapso da rede, uma vez que há pouco esforço no controle das taxas de transmissão nas aplicações multimídia existentes. Além disso, um problema bastante conhecido é que o provimento de fluxos de vídeo com boa qualidade visual é uma tarefa complexa, visto que a capacidade de tráfego disponível na Internet varia intensamente em diversas escalas de tempo, bem como os requisitos de taxa de transmissão de vídeos codificados. Um importante requisito para a transmissão de mídia de fluxo contínuo é que estes devem apresentar comportamento amigável com outros fluxos concorrentes. Desta forma, o principal problema abordado nesta tese é equalizar a variação da capacidade de tráfego disponível com a variação resultante do processo de codificação de vídeo, levando em consideração os requisitos de maximização da qualidade geral junto com a minimização da variação desta qualidade percebida pelo usuário.

A principal contribuição desta tese é a definição e implementação de uma arquitetura para transmissão de fluxos de vídeo em ambientes de rede de melhor esforço. O cenário definido consiste em redes onde são fornecidas informações mais precisas sobre o caminho de rede fim-a-fim, seguindo uma tendência forte na implantação de serviços de gerenciamento ativo de filas na Internet. Isto implica que tais fluxos multimídia devem se adaptar às condições da rede de maneira eficiente, isto é, devem ser capazes de lidar com variações na capacidade de transmissão em diversas escalas de tempo. Com base nestas condições, esta tese descreve uma solução escalável, baseada em soluções do lado do servidor, que combina as propriedades benéficas de mecanismos inovadores de controle de congestionamento, técnicas de processamento de sinal e modelagem estatística de série temporais. Uma extensa investigação destes mecanismos é conduzida e a eficiência da solução proposta é demonstrada por simulação.

Especificamente, uma investigação detalhada do desempenho de transmissão de fluxos de vídeo sobre a Internet é conduzida, baseando-se em alguns protocolos com propriedades amigáveis à rede e com notificação explícita da taxa permitida por fluxo. Como os experimentos mostraram, mecanismos de controle de congestionamento que se baseiam em informações precisas e explícitas oriundas da rede conseguem obter um aumento significativo na qualidade final observada pelo usuário, com menor variação na qualidade percebida, diferentemente dos mecanismos de controle de congestionamento baseado em taxa e menos reativos às condições da rede. Além disso, usando vídeos pré-armazenados e codificados com

MPEG-4 FGS como a aplicação alvo, a arquitetura proposta atende aos importantes requisitos do ponto de vista da rede, i.e. eficiência e justiça com fluxos concorrentes. A arquitetura proposta extrai informações precisas da rede e fornece à aplicação um conjunto de informações de tráfego consistentes e estáveis.

Em linhas gerais, o trabalho desenvolvido nessa tese baseia-se em técnicas apropriadas nas áreas de controle de congestionamento em redes, processamento de sinais e estatística. Com a combinação de diversas técnicas de diferentes áreas, a arquitetura proposta é escalável e adequada para transmissão de fluxos de vídeo na Internet com notificação explícita da rede. Ao mesmo tempo, a arquitetura é flexível para extensões ou mudanças em suas funcionalidades.

Publications

Below is the list of some papers I published during my PhD journey. Publications with topics closely related to this thesis are marked with a star (*).

1. FERNANDES, Stenio F. L. and Sadok, Djamel F. H., “An Adaptive-Predictive Architecture for Streaming Scalable Encoded Video”, submitted to the Elsevier International Journal of Computer Networks. (*)

2. FERNANDES, S., et al., “Clustering Techniques in the Internet Traffic Sampling”, 24th Brazilian Symposium on Computer Networks - SBRC 2006.

3. Barbosa, R., Fernandes, S., et al. “Performance Analyis of P2P Voice Applications”, 24th Brazilian Symposium on Computer Networks - SBRC 2006.

4. KAMIENSKI, Carlos A.; SOUSA, Denio M. T.; SADOK, Djamel F. H.; FERNANDES, Stenio F. L. “Network Architectures for the Next Generation Internet”, Short Course - 23rd Brazilian Symposium on Computer Networks - SBRC 2005 - Fortaleza, CE - Brazil, May 09-13, 2005.

5. FERNANDES, S., et al., “Explicit Feedback Notification for Transporting Multimedia Streaming Flows over the Internet”, 18th IEEE CCECE, Saskatoon, May 2005, Canada, 2005 (*)

6. FERNANDES, S., et. al., “Estimating Properties of Flow Statistics using Bootstrap”, 12th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), 2004

7. FERNANDES, Stenio F. L.; SILVA, GUTHEMBERG S., DIAS, Kelvin L., KAMIENSKI, Carlos A.; SADOK, Djamel F. H. “Peer-to-Peer Traffic Analysis in the RNP Backbone” (in Portuguese), 22nd Brazilian Symposium on Computer Networks - SBRC 2004 - Gramado, RS - Brazil, May 10-14, 2004.

8. FERNANDES, Stenio F. L., SILVA, Wellington. J., SILVA, Mauro. J. C., ROSA, Nelson. S., MACIEL, Paulo R. M., SADOK, Djamel F. H., “Performance Analysis of Message-Oriented Middleware Using Stochastic Petri Nets” (extended version), 22nd Brazilian Symposium on Computer Networks - SBRC 2004 - Gramado, RS - Brazil, May 10-14, 2004.

9. FERNANDES, S., et. al., "On the Generalised Stochastic Petri Net Modeling of Message-Oriented Middleware Systems", 23rd IEEE International Performance, Computing and Communications Conference (IPCCC) - International Workshop on Middleware Performance (IWMP 2004), Phoenix, Arizona, USA, 2004.

10. DIAS, Kelvin L., FERNANDES, Stenio F. L., SADOK, Djamel F. H., "Admission Control and Resource Reservation in Mobile Networks using Time Series Forecasting" (in Portuguese), 22nd Brazilian Symposium on Computer Networks - SBRC 2004 - Gramado, RS - Brazil, May 10-14, 2004. (*)

11. DIAS, Kelvin L., FERNANDES, Stenio F. L., SADOK, Djamel F. H., "Predictive Call Admission Control for All-IP Wireless and Mobile Networks". IFIP/ACM Latin America Networking Conference (LANC 2003), La Paz, Bolivia October 3 - 5, 2003. (*)

12. DIAS, Kelvin L., FERNANDES, Stenio F. L., SADOK, Djamel F. H., "Predictive Call Admission Control for All-IP Wireless and Mobile Networks" (extended version). Telecommunications Magazine, Brazilian Institute of Telecommunications, 2004. (*)

13. DIAS, Kelvin L., FERNANDES, Stenio F. L.; SADOK, Djamel F. H., "Call Admission Control for QoS provisioning in Multimedia All-IP Wireless and Mobile Networks". XX SBT - Brazilian Conference on Telecommunication, Rio de Janeiro, 3-5 October 2003. (*)

14. DIAS, Kelvin L., FERNANDES, Stenio F. L.; SADOK, Djamel F. H., "A Local QoS Control Scheme for IP-Based Wireless Mobile Networks". 5th Workshop on Wireless Communications and Mobile Computing - WCSF 2003 - Brazil, 2003. (*)

15. SOUZA, Denio M. T.; FERNANDES, Stenio F. L.; SILVA, Klebson S., SADOK, Djamel F. H., "Micromobility Protocols Performance in Differentiated Services Networks". CONFERENCE ON INTERNET QUALITY OF SERVICE, part of the SPIE INTERNATIONAL SYMPOSIUM ITCom 2003, 7-11 September 2003 in Orlando, FL USA.

16. FERNANDES, Stenio F. L., et al. "Accurate and Fast Replication on the Generation of Fractal Network Traffic Using Alternative Probability Models" (extended version). CONFERENCE ON PERFORMANCE AND CONTROL OF NEXT GENERATION COMMUNICATION NETWORKS, part of the SPIE INTERNATIONAL SYMPOSIUM ITCom 2003, 7-11 September 2003 in Orlando, FL. (*)

17. FERNANDES, Stenio F. L., et al. “Accuracy and Computational Efficiency on the Fractal Traffic Generation”. In: INTERNACIONAL CONFERENCE ON COMMUNICATIONS SYSTEMS AND APPLICATIONS, 2003, Proc. of the 3rd International Conference on Wireless and Optical Communications - WOC 2003, Banff, CA. (*)

18. FERNANDES, Stenio F. L., et al., “Time Series Analysis Applied to Network Traffic Prediction: A Revisited Approach”, In: INTERNATIONAL CONFERENCE ON APPLIED MODELLING AND SIMULATION - AMS 2002, Cambridge, MA, USA. (*)

Chapter 1. Introduction

1.1 Research Problem

The Internet has been evolving and providing new applications such as video and audio streaming, peer-to-peer systems and IP telephony. Some of these new applications are highly sensitive to delay despite their small bandwidth requirements (e.g., audio streaming and IP telephony). However, video applications have usually much larger objects that need to be streamed over the network. Due to the bandwidth-intensive nature of these objects, deploying scalable video streaming services with satisfactory end-user quality has always been a challenge. Additionally, as end-user Internet access is heading most to high-speed connection (e.g., DSL, Cable)[1], real-time video streaming and video-on-demand services will undoubtedly get higher demands [39]. In fact, according to In-Stat, a technology and market research company, consumer on-line streaming video subscriptions will grow from 2.7 million subscribers in 2004 to 9.85 million in 2007. They also forecast a rapidly growing acceptance by consumers of high-speed Internet services. In-Stat expects a growing rate of more than 500% in the number of high speed Internet subscribers, raising from 30 million households in 2005 to 130 million households by the end of 2007 [89]. In a report analysis, Kaufhold [111] shows that the growth of digital video demands for end-user has finally begun. In such analysis, he provides the most important factors concerning video content distribution that could lead video service providers to a successful market share increase. He called these factors as the "the golden rules". First, he forecasts that digital video delivery resources will grow quickly (Figure 1 reproduces In-Stat's forecast for worldwide growth of Digital Electronic entertainment delivery services). Second, he argues that each delivery services needs to span a number of formats of video, including video over IP, since each TV programs or movie should adapt to different delivery technologies (i.e., different encoders). Since the worldwide market potential for Digital Electronic entertainment content is huge, Kaufhold emphasizes the need for multiple formats along with concerns about multiple technologies required for content delivery. From High Definition TV (HDTV), mobile cell phones, cable and satellite, to broadband Internet connections, the major identified issue is the level of complexity for building out next-generation IP video services.

[1] For example, in Oct 2005 the High-Speed DSL service provided by Bell Canada has 3 to 4 Mbps of bandwidth capacity (downstream).

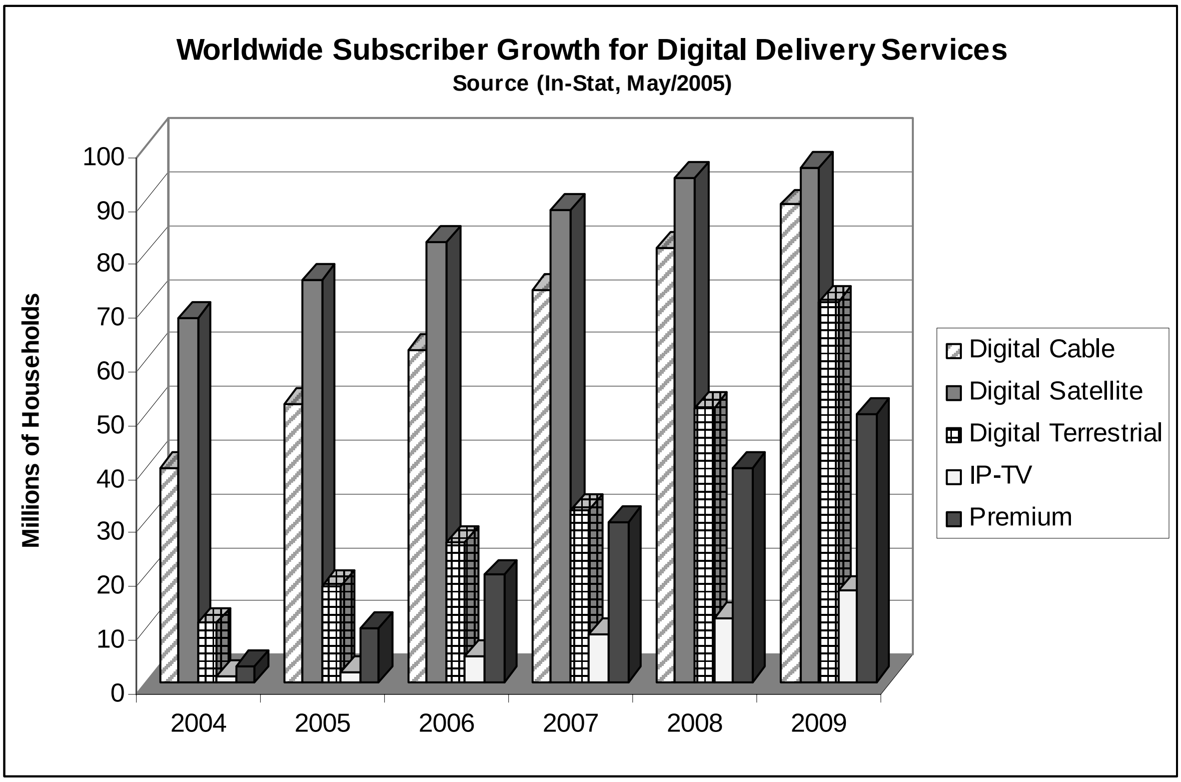

Figure 1 - Forecasts for Video delivery services

Furthermore, as the problem of safely delivering video content becomes more complex, it turned out to be a crucial requirement for delivery service providers (Figure 2 shows the required infrastructure for entertainment services providers in order to offer several formats to a number of technologies). Sripanidkulchai et al [210] point out that live streaming will become an important traffic class. In such scenario, it is clear that the Internet provides an attractive way to reach global audiences ranging from small to large sizes. They argue that as people become more mobile, the demand for staying connected to its local content will increase. In addition, from a technology perspective, as both wired and wireless broadband access becomes ubiquitous, the technology obstacle to live multimedia streaming will eventually vanish.

However, there are many technical challenges for a wide deployment of video services over the Internet, which include the fields of video encoding and networking protocols [175]. In both cases, scalability is the crucial factor that can allow video services providers to meet the requirements associated with Internet video. According to Radha et al. [175] from the encoding point-of-view, scalability is crucial for delivering the best possible video quality over unpredictable highly dynamic networks, where bandwidth variation in several time-scales is a reality. In other words, video scalability enables applications to adapt the video quality to changing network conditions. From the network point-of-view, scalability enables the service provider to scale with the number of users.

In this context, recent streaming traffic measurement data [210] [228] stress the need for careful deployment of multimedia services in the Internet. The main reason for that relies on the fact that most media players automatically query the network to determine which transport protocol it should use. The fact of the matter is that *middleboxes* (i.e., Network Address Translators, Firewalls etc) will eventually block certain protocols. Therefore, media players will try to discover which protocols they may use. For example, the data collected in [210], which consist of live streaming workloads from Akamai Content Distribution Network over a 3-month period (more than 70 million requests for 5,000 distinct URLs), reveal that QuickTime and Real Player have 60% of sessions using UDP traffic and the remaining 40% of sessions using TCP. On the other hand, for Windows Media Player, TCP sessions are the majority, at 80% of all sessions. In addition, they found that roughly 40-50% of the AS domains are TCP-dominant. They raised a hypothesis that hosts from such domains are behind middleboxes that could limit the use of UDP-based applications.

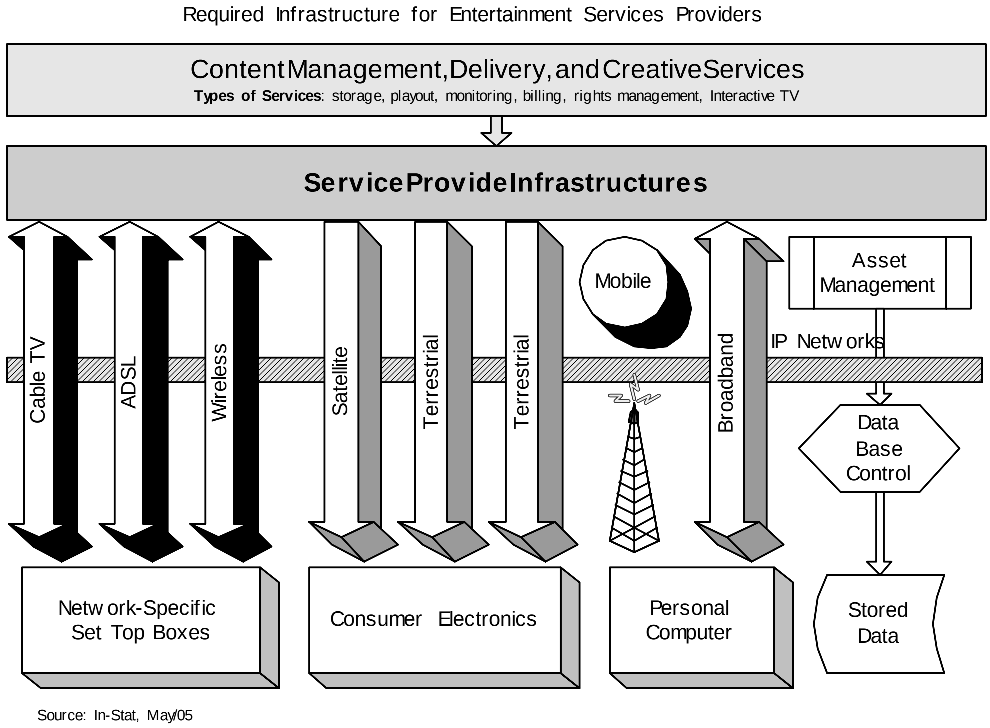
Required Infrastructure for Entertainment Services Providers
Content Management, Delivery, and Creative Services
Types of Services: storage, playout, monitoring, billing, rights management, Interactive TV
Service Provide Infrastructures
Cable TV
ADSL
Wireless
Satellite
Terrestrial
Terrestrial
Mobile
Broadband
Asset Management
IP Networks
Data Base Control
Network-Specific Set Top Boxes
Consumer Electronics
Personal Computer
Stored Data
Source: In-Stat, May/05

Figure 2 - Infrastructure for delivering video services in different formats

Such factors imply in a higher requirement for bandwidth in the network while preserving their requirements for small delays. In order to prevail over this issue in the short term, content providers of multimedia services have been relied on offering restricted options to the end-user. For instance, one can observe that in most cases users are required to choose from a few available bitrates video streams (e.g. normally for high and low speed connections), which should match to the capacity of their access to the Internet approximately. After that, the content provider supplies them with a stream at a constant bitrate (CBR). Recent measurements show that most streaming video encoded with either Windows Media or Real Media on the Internet have bit rates around 300 Kbps [228], thus providing Internet video services with below DVD-quality images and sound.

In such scenario, one could see an impending collapse if the number of users increases faster than the overall network capacity's growing rate, since there is a little effort in applications to control their sending rates [155]. Moreover, the quality perceived by users is highly dependent on the network traffic fluctuations. For instance, if the available bandwidth for the user is much less than the previously chosen option, such video streaming will become a sequence of cyclical playouts and interruptions (e.g. buffering). One possible solution to avoid such shortcoming is forcing video source sending rate to adapt to the network conditions in a fine-grain manner. Such alternative is achievable in two ways. First, it is possible to applications rely on coding standards that provide flexible and low-overhead properties to be used in rate controllers (within applications) to match variable network capacity in an end-to-end approach [44][45][85][183][184][235][249]. Second, in order to surmount difficulties that arises naturally in this approach, applications could take advantage of receiving notifications about network conditions in a regular basis, provided by an underneath congestion control architecture with explicit feedback information. As stated by Lakshman [131], an adaptive encoder that satisfies network traffic constraints will eventually achieve the best possible decoded video quality, since such observance of network state minimizes losses. By combining these alternatives, we do believe that is possible to achieve a video streaming with low variation in quality (e.g., in the user's point of view), as long as the network fluctuate within some lower and upper available bandwidth limits.

As a crucial factor for providing stability to today's Internet, congestion control has been actively studied in the area of networking. In such field of study, two general approaches emerged, namely the end-system based approach and the network-based approach. The end-system based solution consists of source or receiver based congestion control schemes. These schemes try to avoid congestion in the Internet by adjusting down their transmission rate, on every occasion that a network congestion event is detected. Van Jacobson proposed the congestion avoidance and control features in TCP, which is the *de facto* transport layer protocol for the current Internet [91]. We remind that such schemes are able to work without explicit network support. It is clear that end-system based congestion control mechanisms are mandatory in today’s Internet. However, they are not good enough to provide high-quality service under all circumstances since there is an upper limit to how much control that can be achieved from end-systems alone [59] [98] [102] [129] [179] [245]. As a result, in order to complement the end-systems congestion control mechanisms, network-based approaches rely on Active Queue Management (AQM). The network-based approach requires network elements (e.g. routers, switches) to interfere actively in the traffic in order to guarantee fairness between competing flows as well as to prevent large delays and packet loss in the network. By proposing AQM in network elements as a pro-active method for managing queue lengths, the Internet research community took a step forward to a more precise support from the network to the end-systems. After more than a decade that the first AQM approach appeared [59], the Internet research community has now focused on scalable AQM solutions with explicit feedback notification, as they can provide better performance results than the previous ones [129] [134] [245]. Some researches have recently pointed out the trends towards explicit signaling in a stable and scalable fashion ([237] [238] [251]). Even though there are a number of AQM propositions most Internet routers still operates with the traditional Drop Tail queues. This is mainly due to configuration, implementation and deployment problems inherent to all AQM solutions. We will discuss this issue in details later in Section 2.2.

An analysis over all of these factors reveals that providing perceptually good quality streaming video is a complex task. This is because in today’s best effort Internet the available bandwidth can fluctuate strongly as well as encoded video can exhibit significant rate variability at several time-scales. Bear in mind that one important requirement for streaming multimedia flows is that they must exhibit fairness with competing flows. We strongly agree with Vieron and Guillemot [224] that point out that key issues for designing congestion control mechanisms dedicated to multimedia streams are smooth rate indications as well as fairness with competing flows. Additionally, in the application level, the multimedia source should rely on adaptive control techniques to maximize the video quality. We also argue that with the steadily growth of Multimedia demands in the Internet such applications should take advantage of either explicit or pro-active network information in order to use fine-grained coding and rate control procedures precisely.

Therefore, the main research problem addressed in this thesis is how to accommodate the mismatch caused by available bandwidth variability and the encoded video rate variability, taking into account the requirement of the minimization of the quality variability and the maximization of the overall quality of the video rendered to the user. To do that, in this work we shed some light on the performance of explicit feedback-based congestion control protocols and study their suitability for streaming scalable encoded video. Additionally, our work develops an architecture for streaming scalable encoded video, which includes adaptive and predictive units for efficient transmission.

This section defined the key problems associated with streaming scalable video in today's Internet, the special characteristics of streaming media content along with the problem quality adaptation when facing network traffic fluctuations that motivated the development of our novel architecture.

We organized the remainder of this chapter as follows. We describe the research scope of this thesis in Sections 1.2 along with an overview of the proposed solution in Section 1.3. Finally, we present the organization of the remainder of the thesis in Section 1.4.

1.2 Research Scope

In the context of the MPEG-21 Multimedia Framework (MF) [20][24][206], which main goal is to provide improved utilization of multimedia resources within a wide range of networks and devices, adaptation is crucial for wide deployment of multimedia contents. Moreover, Universal Multimedia Access (UMA) requires seamless access to multimedia content. In such contexts, it is necessary some kind of selection or adaptation of content based on the user's dynamic profile, which may include rate or quality reduction, sampling etc [220]. The MPEG-21 MF provides descriptions of the Digital Item Adaptation (DIA), which is of particular relevance for UMA. Universal Multimedia Access (UMA) refers to the ability for any type of terminals to access and consume a rich set of multimedia content. As stated by Vetro [222], ideally UMA should work seamlessly over dynamic and heterogeneous networks and devices. Scalable coding techniques are the first steps toward this goal of universal accessibility, although the challenges for transmitting video over highly dynamic networks (i.e., wireless networks, best effort Internet) remain. Such approach offers ways to adapt to network conditions, i.e. by adapting coding parameters in real-time, while increases the computational overhead (e.g., in servers with high request rate). When there are few limitations on the terminal (e.g., buffer memory) or access network (e.g., available bandwidth), real-time adaptation is viable. On the other hand, the case of streaming pre-stored FGS coded video is an alternative to such real-time coding, since content providers can stream content ranging multiple bit-rates matching the requirements in the underlying network. Burnett et al [24] and Pereira [167] pointed out the eventual need to focus on problems of adaptation for a number of multimedia content and environment conditions (i.e., network state). In other words, applications in conformance to the MPEG-21 MF will face the problem of adapting multimedia resources and maximizing the user experience in a variety of contexts. Therefore, in order to achieve interoperable transparent access to multimedia content, MPEG-21 MF requires the adaptation of Digital Items, as shown in Figure 3. In the context of this conceptual architecture, our novel architecture can provide reliable information for the resource adaptation engine. One should notice that a Digital Item may be also subject to a description adaptation engine or a DID adaptation engine. In the context of UMA, part 7 of the MPEG-21 standard, entitled Digital Item Adaptation – DIA, describes tools to guide the multimedia adaptation engine. The adaptation engines themselves are open to different implementations [90]. For example, such engines should adapt to usage environment. It requires descriptions of terminal capabilities and coarse-grained network characteristics, as well as user and natural environment characteristics [222].

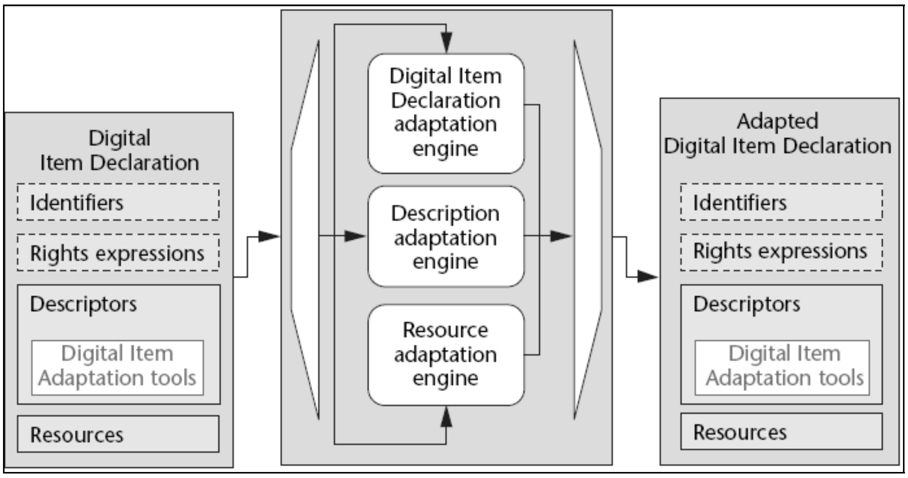

Figure 3 - Digital Item Adaptation Architecture (from the reference [24])

In the scope of this thesis, i.e., the networking issues for DIA, the network characteristics are the most important. In DIA [90], attributes, such as capabilities and conditions, describe the network characteristics. Capabilities define static characteristics of a given network (e.g. maximum capacity, minimum guaranteed bandwidth) whereas conditions describe its dynamic behavior (e.g. available bandwidth, error, delay). Please note that the standard does not tight to any inference technique for gathering network condition information. It only emphasizes that applications should perform such measurements.

1.3 Overview of the Proposed Solution

In a few words, we propose an architecture that takes into account the huge impact of network conditions on continuous media streaming applications and vice-versa. Such architecture relies on fine-grained information at the network and transport level, in order to provide options to multimedia applications to either adjust their sending rates, or even possibly chose better options for coding procedures (e.g. MPEG-4 FGS in the context of the standards MPEG-7 and MPEG-21 Multimedia Framework).

First, in our proposal, we ensure that we meet the most important requirement from the network point of view that is transporting multimedia flows efficiently while exhibiting fairness with competing flows. Therefore, the main issue is to accommodate the mismatch caused by available bandwidth variability and the encoded video rate variability, keeping in mind that we should minimize quality variability and maximize overall video quality rendered to the end user. Second, another objective is extracting from the network level the most precise information as possible and then providing video source application with consistent and stable information.

Therefore, we look for the best tools in the networking and statistical fields in order to build our solution. By merging ideas from several areas, we propose a scalable architecture for video streaming over best effort networks with explicit rate notification. Such novel architecture is flexible to extend, subtract, or change functionalities.

1.4 Outline of the Thesis Proposal

We organize the rest of the thesis as follows. In Chapter 2, we present an overview of several research efforts in the field of streaming multimedia over the Internet. We also provide the necessary background to a good understanding of this thesis by describing some congestion control mechanisms that rely on explicit feedback notification.

In Chapter 3, we perform a quantitative evaluation of the end-user perceived media quality of video streaming under network friendly protocols in the best-effort Internet. To do that, we evaluate the effect of throughput variation of such protocols on the quality of scalable encoded stored video.

In Chapters 4 and 5, we take a step further on the analysis and solution of the problem of providing minimal quality variability and maximum quality of the video rendered to the end-user by accommodating the mismatch caused by available bandwidth variability and the encoded video rate variability.

We summarise our work, discuss our main contributions, and illustrate some possibilities for future work in Chapter 6.

Chapter 2. Background and Related Work: Congestion Control and Multimedia Streaming in the Internet

It has been challenging times for the Internet research community in the last few years. One of the fields that have been receiving great attention and research effort is congestion control and adaptation for multimedia applications in the Internet. In fact, this is not a single topic, but an assortment of Admission Control, Internet Congestion Control, Content Distribution Networks, Proxy Cache Systems, Scalable Coding, Error Concealment, and Forward Error Correction. In less than two years that this issue has become a noteworthy part of research in network traffic control, it has become evident that the preceding approaches to maximizing client perceived QoS are insufficient since most proposals deal either with the end-system based (application) control or network-centric based schemes uniquely. As we presented earlier in this thesis, there are many technical challenges for a wide deployment of video services over the Internet, including the fields of video encoding and networking protocols [214]. In [176], Radha et al pointed out that in both coding and networking issues scalability is the main factor that could allow video services providers to meet the requirements associated with Internet video. According to them, from the encoding point-of-view, scalability is crucial for delivering the best possible video quality over unpredictable highly dynamic networks, where bandwidth variation in several time-scales is a reality. In other words, video scalability enables applications to adapt the video quality to changing network conditions. From the network point-of-view, scalability enables the service provider to scale with the number of users with minimal losses, maximum network utilization, and fairness with competing flows. Such issues lead to the requirement that servers and transport protocols should cope with large number of video streaming flows while maintaining low variation in quality rendered to the end-user.

In the description of the MPEG-4 Fine-Grained Scalable video coding method for multimedia streaming over IP networks [176], Radha et al argue that scalable coding solution in the Internet must meet some important requirements. Firstly, the solution must enable a video streaming server to perform minimal real-time processing and rate control. This is especially useful when dealing with a large number of simultaneous streams. Secondly, as shown before, it should be highly adaptable to unpredictable bandwidth variations due to heterogeneous access-technologies of the receivers and due to dynamic changes in network conditions. Finally, the video encoding method should enable low-complexity decoding and low-memory requirements to support a wide range of receivers. Furthermore, the scalable bit stream must be resilient to packet loss events, which are quite common over the Internet. At this point, we stress the importance of a proper choice for transport protocols suitable for multimedia streaming. Using the existing widespread transport protocols, such as TCP and UDP, most server-side multimedia applications must cope with packet losses and provide their own adaptation policies. Please note that most adaptation policies in literature rely on estimation of the available bandwidth. As inference about available bandwidth is not very accurate, applications suffer with packet losses and must overcome this issue with additional error recovery techniques, such as forward error correction and error concealment. Therefore, if developers of multimedia applications could rely on suitable transport protocols with minimal or zero packet loss, while maintaining fairness with competing flows, they would safely focus their work on minimize quality variation in the streaming session.

In order to provide the necessary background to a good understanding of this thesis, we present an overview of several recent research work related to congestion control mechanisms and streaming multimedia over the Internet. First, we discuss a number of end-to-end congestion control approaches as well as the current research development of router-based congestion control schemes. Second, we give an overview of several recently proposed solutions to transport multimedia streaming. Specifically, we discuss rate and quality adaptation mechanisms. Finally, we conclude this chapter drawing some final remarks and point out few directions toward a novel approach in this field. Section 2.1 presents an overview of current congestion control mechanisms deployed in the Internet. Section 2.2 presents some new proposals for congestion control with emphasis in protocols that rely on explicit signaling information from the network. Section 2.3 describes several researches concerned with only one part of the problem (end-system or network). A number of recent works have shown tendencies to combine these approaches in order to achieve better performance (in both end-systems and network), as we will present in Section 2.4.

2.1. Current Congestion Control Mechanisms in the Internet

Broadly speaking, the main goal of any congestion control scheme is to control the end system's sending rate in order to achieve fairness and high network utilization. In addition to that, it is highly desirable that such schemes result in small queue sizes as well as low packet drops. In the current Internet infrastructure, such control is achieved by a cooperative work of both routers and end systems. In access and core networks, routers do a simple task (from the point of view of any congestion control mechanism) that is to drop packets when their queues become full. For end systems, this action implicitly conveys a sign of congestion. In other words, senders that run congestion control mechanisms interpret such loss as a sign of congestion and their common response to this notification is to decrease their sending rate. As pointed out in [107], although the Internet research community has encouraged the deployment of new mechanisms in routers, such as a more active role in anticipating congestion and explicitly signaling it to the senders [10] [59] [127], the above described model has not changed for years.

As we are going to describe later in this chapter, active controllers inside routers, commonly referred to as Active Queue Management mechanisms (AQMs), have received great attention recently. At this point, it is important to emphasize that until the proposal of the Explicit Control Protocol (XCP) [53] [107] and the Congestion Avoidance with Distributed Proportional Control (CADPC) [238], most of the previous research papers related to Internet congestion control were concerned with either the design of the controllers at the network, or at the end systems. In other words, XCP and CADPC present new frameworks with a joint congestion control design both at the end systems and at the routers.

2.1.1. End-System Congestion Control Protocols

Fundamentally, senders must probe the network to infer the network state, i.e., to detect congestion. Most end-system congestion control protocols make use of packet losses as an implicit signal of network congestion. As long as there are no losses, the sender can increase its sending rate. On the other hand, as soon as it discovers any packet drop, it should decrease its sending rate.

From a control theory point of view, end-systems congestion control protocols are closed-loop systems. The main difference between them is related to their behavior either when facing a congestion signaling (i.e., the decreasing mechanism) or how to probe the network for the available bandwidth (i.e., the increase rules) [128].

2.1.1.1. Window-Based Congestion Control

The basic operation of TCP is specified in RFC 793 [169]. Briefly, the evolution of TCP is based on what follows. TCP is a window based transmission protocol, meaning that it controls the window of outstanding packets in the network. In 1988, Jacobson [91] proposed its first congestion control mechanism. In this proposal (known as Tahoe TCP), some important modifications were included, such as slow start, exponential timer backoff, congestion avoidance (AIMD phase), fast retransmit, and Round Trip Time (RTT) estimator. In 1990, the fast recovery algorithm was added to TCP's congestion control [6][211] (the Reno TCP). In [152], Mathis et al. proposed a new version of TCP, usually referred to as SACK, which allowed the receiver to use selective ACKs (called SACKs) to request retransmission of specific lost packets. Floyd and Henderson et al. [58] proposed a modification to TCP Reno, called NewReno that did not suffer from the same performance problems as the original. Brakmo and Peterson [23] proposed a new TCP congestion control scheme called TCP Vegas, which incorporated a bandwidth estimation scheme into the sender. We refer the interested reader to the reference [81] for a complete description of the TCP design evolution path.

TCP performs poorly in high-speed networks because of its slow response with large congestion windows. New implementations of TCP for dealing with high-speed networks issues alter the parameters of the AIMD-based congestion control mechanism. Most proposals improve performance in high-speed networks with no modification to standard TCP receivers. Currently, there are three main proposals in this field, namely High-Speed TCP (HSTCP) [57], Scalable TCP [112] [114] and FAST TCP [100] [236].

The window adjustment mechanism of FAST TCP [100] reacts to both queuing delay (RTT information) and packet loss. In essence, it is an improved version of TCP Vegas attuned for high-speed networks. HSTCP [57] uses a modified TCP response function for different network environments. For environments with gentle or heavy congestion events, it uses the regular TCP AIMD parameterization (i.e., $\alpha = 1$ and $\beta = 0.5$). For networks with very low packet loss rates, HSTCP presents a more aggressive response function, by introducing a new relation between the average congestion window and the steady-state packet drop rate. For a more detailed description of HSTCP and its advantages, please refer to [57]. Scalable TCP [114] builds on the HSTCP proposal. It modifies the AIMD additive parameter to $\alpha = 0.01$ and the decrease parameter to $\beta = 0.125$. It is worth noting that Scalable TCP algorithm utilizes constants to AIMD parameters whereas HSTCP's parameterization depends on current window size and packet loss rate.

Although the authors of the above protocols claim that their proposals achieve high performance while maintaining fairness with concurrent flows, recent experimental research study on these protocols indicate that those claims do not hold [126]. For instance, in the case of long-lived data transfer flows running concurrently with short-term web browsing flows or long-term video streaming flows, there are a noticeable degradation in all concurrent applications. There are strong indications that such unacceptable performance it is due to their inherent loss-based mechanism.

This evidence makes a persuasive argument against the use of these high-speed versions of TCP for streaming scalable encoded video. From the point of view of video streaming, smooth rate variations are extremely necessary to maintain acceptable quality transmission at the application level [44] [224], since excessive oscillations in throughput induce undesirable cost on the visual quality of the received video signal. In other words, an appropriate congestion control mechanism for video streaming must provide smooth throughput variability to applications in order to reduce playout buffering at the receiver. On the other hand, such smoothing processes should not prevent the congestion control mechanism from responding promptly to changes in network conditions. One should observe that all AIMD-based congestion control algorithms, i.e. most TCP flavors, have strong limitations for video applications due to the sawtooth rate pattern they exhibit.

2.1.1.2. Equation-Based Congestion Control

As we argued in the last Section, TCP end-to-end congestion control is not appropriate for most continuous media streaming applications under a variety of network conditions [228]. The main reason is that halving the sending rate in response to congestion events can be harsh and eventually will reduce the user-perceived quality. In the past few years, Equation-Based Congestion Control (EBCC) mechanisms have received great attention as an alternative answer to provide smooth traffic in the transport level. Broadly speaking, equation-based congestion control relies on a throughput equation that explicitly estimates the maximum allowed sending rate in a RTT. EBCC mechanisms take into consideration the loss event rate mainly. Providing the source with such information, it will be able to adapt its sending rate accordingly. In most EBCC proposals (e.g. [78], [159], [185], and [187]), the throughput equation is based on a model of the steady-state TCP rate, as a function of the RTT and the steady state loss event rate.

TCP-Friendly Rate Control (TFRC)

TFRC seems to be the preferred choice by the Internet “bureaucratic” community for transporting multimedia flows [78]. It is based on an EBCC mechanism and designed for unicast flows in the Internet [60] [157]. It is worth stressing that RFC 3448 does not specify the protocol completely and only describes a congestion control mechanism that could be integrated in any transport protocol, or in any application-level rate control. The main design goal of TFRC is to serve as a base for building applications whose first requirement is a smooth throughput while keeping fairness. As its authors comment in RFC 3448, one should be aware that the price of having a smooth and fair sending rate is that TFRC reacts slower than TCP to abrupt changes in available bandwidth. Another important difference between TFRC and TCP is that the former is a receiver-based mechanism. This means that the estimate of the necessary information, to calculate the fair sending rate, is done at the receiver rather at the sender. A common caveat is related to the label “receiver-based”. Specifically to TFRC, this term does not mean that the receiver does all the calculations compulsorily (e.g., the fair throughput) and sends this back to the sender. It could just collect some basic information, such as the loss rate, and feed them back. We refer the interest reader to RFC 3448 [78] (Section 3.2) for details about packet format and content.

Protocol Description and Parameterization

The rationale in the congestion control mechanism for TFRC is tightly coupled to the throughput model of TCP’s sending rate as a function of three main parameters, namely loss event rate, packet size, and round-trip time [157]. The main reason here is to maintain fairness with TCP. In addition to that, in order to achieve a smooth sending rate, TFRC's congestion control mechanism follows few steps, as described in the RFC 3448 [78]:

a) The receiver measures the loss event rate and feeds it back to the sender;

b) The sender uses the feedback messages to measure the round-trip time (RTT);

c) Those parameters are fed into the throughput model, giving the allowed sending rate;

d) The sender regulates its sending rate to match the allowed rate.

Although any throughput model for TCP that takes into account loss events and RTT as its main parameters is suitable for use in TFRC, the current proposal uses a modified version of the throughput equation for the TCP-Reno [158]. The throughput equation is the following [60] [157]:

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}\left(3\sqrt{\frac{3p}{8}}\right)p\left(1+32p^2\right)} \tag{1}$$

Where:

- T is the allowed rate (bytes/second),
- s is the average packet size (bytes),

- R is the round trip time (seconds),
- p is the loss rate as a fraction of the number of packets transmitted (between 0 and 1.0),
- t_{RTO} is the TCP retransmission timeout value (seconds),
- b is the number of packets acknowledged by a single TCP acknowledgement.

Recall that the parameters packet size, loss event rate and RTT must be measured or inferred by a TFRC end-system.

Sender Functionality

As we briefly described before, a sender adjusts its controlled sending rate every time a feedback report arrives from the receiver. Moreover, observing a specific timer, the sender must halve its sending rate if it does not receive feedback information for two successive round trip times.

The sender-side protocol has the following steps:

1) Measurement of the packet size: As stated in the RFC 3448, this parameter may not be known a priori since the packet size depends on the data. The authors suggest the use of the mean as an estimate. We consider this as a point of failure of the TFRC, since its throughput equation deeply relies on the parameter packet size. We argue that first-order statistics could not be an accurate measurement, if the packet size follows for instance, a heavy tailed probability distribution function. This is still an open issue and the authors suggest a discussion in a separate document. Hence, they assume that the sender can precisely estimate the packet size, and that congestion control is performed by adjusting the number of packets sent per second.

2) Adjustments when a feedback report arrives: Assume that the sender knows its current sending rate (T), the current round trip time (R), and the timeout interval (t_{RTO}). When a feedback report is received, the following actions should be performed:

    a) Calculate a new RTT estimate based on the new sample and previous estimate. It uses an Exponential Weighted Moving Average (EWMA) method;

    b) Update the timeout interval using the standard TCP equation;

    c) Update the sending rate, taking into account whether there are packet losses or not. In general, if the actual sending rate is less than the evaluated one (T), the sender may increase its rate. On the other hand, if the actual sending rate is greater than the estimated, the sender should decrease its rate to the target, T.

3) Adjustments when there is no feedback in a 2-RTT period;
    a) It should halve its sending rate;
    b) Recalculate the new allowed sending rate;
    c) Restart the *nofeedback* timer.
4) Oscillation prevention (optional): To prevent oscillatory behaviour in environments with a low degree of statistical multiplexing, the authors proposed a slight modification to the sender's transmit rate. Hence, in order to integrate a congestion avoidance behaviour, the sender tries to reduce the transmit rate as the queuing delay (and hence RTT) increases.

Each data packet sent by the sender must contain a sequence number, a timestamp, and its current estimate of the RTT.

Receiver Functionality

The receiver-side protocol has the following steps:

1) Provide feedback to the sender: An important role at the receiver is to provide feedback to the sender for every data packet received or whenever a new loss event is detected. This lets the sender estimate the round-trip time (RTT) periodically. The authors mention the possibility of sending periodic feedback messages more than once per RTT when the sender is transmitting at a high rate, but there is no clear advantage in this approach. Finally, it should add the received packet to the packet history.

2) Expiration of feedback timer: When the feedback timer expires, the action to be taken depends on whether data packets have been received since the last feedback was sent.

The receiver also feeds back to the sender the loss event rate, p, which is one of the vital parameters of the protocol. Obtaining an accurate and stable measurement of the loss event rate is critical for TFRC. Such measurement must be based on the detection of lost or marked packets from the sequence numbers of arriving packets. The authors consider that there is a trade-off between measuring the loss event rate over a short period (with a fast response to changes in the available bandwidth), versus measuring over a longer period (with a slow response). There are two long sections in the references [60] and [78] with extensive discussion about measurements and calculations related to loss events. Although we will provide details about this process throughout the thesis, we refer the interested reader to those references for a more precise explanation of how it works.

Finally, each feedback packet sent by the receiver must contain the timestamp of the last data packet received, the current estimate of the data rate, the current estimate of the loss event rate, and the time delay between the last received data packet and the generation of the feedback report.

General Remarks

We conclude this Section by shedding some light on some drawbacks to the current TFRC proposal that could cause difficulties for a widespread deployment. First, the proposal for handling varying packet size is mentioned, namely TFRC-PacketSize (TFRC-PS), but not specified yet at the time of writing this thesis. We consider that as a point of failure of the TFRC, since its throughput equation relies very much on the parameter packet size. As stated in the RFC 3448, this parameter may not be known a priori since the packet size depends on the data. The authors suggest the use of the mean as an estimate. We argue that first-order statistics could not be an accurate estimate, if the packet size follows, for instance a heavy tailed probability distribution function. This is still an open issue and the authors suggest a discussion in a separate document. Hence, they assume that the sender can precisely estimate the packet size, and that congestion control is performed by adjusting the number of packets sent per second. Second, as stated by the authors, the dynamics of TFRC are sensitive to how the measurements are performed and applied. In addition to that, we emphasize that such dynamics are also highly sensitive to the protocol's parameterization.

Finally, we agree with the authors that state that TFRC considers it as a viable alternative for unicast multimedia flows only in a situation where application developers avoid any end-to-end congestion control mechanism. However, its dependency on the configuration parameters could impair further developments. Although it is open for changes and its authors advocate that different TCP throughput models can substitute the current TFRC's throughput equation, we keep the argument concerned with parameterization. From the work of Vojnovic and Le Boudec [225] [226], we can draw some arguments against the deployment of TFRC as a transport protocol for streaming multimedia flows. TFRC is a particular case of unicast equation-based rate control, where a source adjusts its rate to a certain level based on a TCP throughput formula, which depends mainly on estimates of the loss-event rate, the mean round-trip time. Their goal was to identify whether sources that rely on equation-based rate control is indeed TCP-friendly. From ns-2 experiments for TFRC, they found out that in some of the experiments, TFRC is non-TCP-friendly despite the fact that they observed its conservativeness. In addition to the ns-2 simulation experiments, they also performed Internet experiments to verify whether TFRC is TCP-friendly. Some results showed that for small loss-event rates TFRC is significantly non-TCP-friendly. Furthermore, there are many scenarios where excess of conservativeness is present.

2.2. New Approaches for Congestion Control Mechanisms in the Internet

TCP and its extensions have been proved to be stable and efficient as the principal basis for Internet operation. Most TCP flavors have been very triumphant over many network characteristics, such as over a variety of capacity, propagation delay, and loss patterns. However, with the rise of high-speed networks (e.g., in the order of gigabits-per-second), wireless links with high lossy patterns or latency, some authors have been arguing that TCP has little room for evolving its performance in such environments [56] [109] [98] [99] [100]. Additionally, there are strong arguments for providing end-systems with more information (congestion-related state) in order to aid TCP algorithm's to infer impending network congestion in advance [22] [129] [179] [245]. Such network-centric mechanisms are known as Active Queue Management (AQM) systems, which main objectives are trying to keep network router's queues small and conveying end-systems congestion through in-band signaling (i.e., inside IP packets). One of the first ideas of AQM was the proposal to add Explicit Congestion Notification (ECN) to IP (RFC3168) [179], which uses two bits of the ToS field in the IP header pointing out the following state information: not ECN enabled, ECN enabled without congestion experienced, and ECN enabled with congestion experienced. Such in-band signaling, along with packet losses, conveys information to end-systems conduct a more precise congestion control.

As a general concept, AQM mechanisms allow routers to use any suitable technique to detect congestion [59]. In general, a receiver signals any congestion experienced on the forward network path back to the sender in *ACK* packets, and senders can react to congestion similarly as if packet loss were detected, by simply updating their congestion window. There are many additional proposals for AQM schemes with different motivations and features. Particularly, RED was motivated by the desire to avoid synchronization of TCP flows that could potentially lead to poor throughput and unfairness between competing flows. RED uses as its main parameter an Exponentially Weighted Moving Average (EWMA) of the current queue length to infer congestion. It also drops packets randomly in order to signal congestion. Although there are strong arguments against RED [153] [181], its authors advocate that RED can reduce congestion signal latency and can desynchronize flows effectively. As far as we concern to multimedia streaming, following the movement towards a more proactive network, Chan et al [27] proposes an AQM scheme, namely Jitter Detection (JD), for gateway-based congestion control to stream multimedia traffic in IP networks. Arguing that a high jitter level is the main factor leading to multimedia synchronization problems and performance degradation of the streaming buffer in the client, the proposed AQM scheme tries to detect and discard packets that accumulated enough jitter. They also reveal that the proposed scheme can maintain the same TCP friendliness when compared to that of RED and DropTail. The simulation results show that JD scheme reduces the average delay jitter of the multimedia packets and maintains a high throughput for the multimedia flows when compared to that of the traditional naïve AQM schemes.

All AQM proposals were just small steps towards a more scalable and robust solution to the problem of improving TCP performance with explicit signaling information. As pointed out by Falk and Katabi [110], such measures do improve performance, but there is a clear limitation in these solutions, as they do not require explicit and precise information from the network routers. Recently, Katabi proposed the Explicit Control Protocol (XCP) [107] [108] [109] [110], which can be seen as the utmost solution to the Internet congestion control problem. Its main characteristic is to extract in-band congestion state information directly from routers, without any per-flow state. Almost at the same time, Welzl proposed the Performance Transparency Protocol (PTP) in conjunction with the Congestion Avoidance with Distributed Proportional Control (CADPC) [238] [240], which relies on rare explicit out-of-band signaling to build an altered congestion control model. According to some performance evaluations [109] [240], both XCP and CADPC give excellent performance over a broad range of network characteristics, including high speed and high latency links. XCP and CADPC have also been proven to achieve fairness and maximum link utilization (network efficiency). In this thesis, we take a further look at the performance analysis of both XCP and CADPC, by verifying their suitability in transporting multimedia streaming over best effort networks. In our point of view, as such protocols do provide fairness against TCP flows, maximum link utilization, and efficiency, we advocate that the problem of streaming multimedia flows could be focused only on the application level, concerning with the minimization of quality volatility. In other words, we propose decoupling the transport level issues from the application level ones, since all related work had hitherto been concerned about solutions that meet both transport and application level requirements. Hence, in the next two Sections we describe both XCP and CADPC, since these protocols will form the starting point for our proposed architecture.

2.2.1. The Explicit Control Protocol - XCP

The main contribution of the XCP proposal is untangling efficiency from fairness policies of congestion control mechanisms. It is a scalable solution since it relies on carrying per-flow congestion state in packets. In other words, there is no need for routers to keep any per-flow state. Although there are some deployment concerns, since XCP requires changes in both routers and end systems, it has received great attention from the networking research community lately [146] [250].

The design rationale of XCP relies on the observation that packet loss is not a suitable signal of network congestion. Katabi et al [109] argue that as an implicit and binary signal of congestion, loss only signals whether there is congestion or not, forcing senders to probe the network until its limits before backing off. Additionally, since the feedback is imprecise (binary), it is a common sense that in order to avoid congestion the increase policy at the sender must be as conservative as possible, whereas the decrease policy must be aggressive. The Additive Increase Multiplicative Decrease (AIMD) policy in TCP congestion avoidance phase is a clear example of such behavior. From the point of view of the Control Theory, a stable control requires explicit and precise feedback. Hence, to overcome this issue, the proposal of putting the network state in the congestion headers tries to reflect the congestion level on the network path. As presented in [107], Katabi proposed using precise and explicit in-band congestion signaling, where the network explicitly tells the sender the state of congestion on the network path. The senders should react to such precise information adequately, resulting in a protocol that is both more responsive and less oscillatory.

Protocol Description and Parameterization

XCP framework's main characteristic is providing precise feedback from the network to the sender on the maximum allowed sending rate. In order to provide such precise and explicit feedback, XCP relies on the congestion header in each packet. Routers play a significant role in the XCP framework, as they scrutinize and may update each congestion header as packets travel from the sender to the receiver. The receiver roles are copying the updated congestion header into acknowledgment packets and sending them back to the sender in the same flow.

The congestion header contains four data fields for providing precise information.

- *RTT*: current estimate of the round-trip time;

- *Throughput*: current sending rate;
- *Delta_Throughput*: amount that the sender should use to either increase or decrease its sending rate. In order to reflect the network's allocated change in throughput and provide fairness, routers along the network path may update this value. One should notice that this value can be a negative number if there is an impending congestion;
- Reverse_Feedback: at the receiver, Delta_Throughput value is copied in the Reverse_Feedback field and sent back to the sender in a returning packet;

Router Functionality

Broadly speaking, an XCP router calculates the fair share for a given flow and distributes it for each packet in such flow. In other words, an XCP router tries to compute a feedback that forces the closed-loop system to converge to efficiency and fairness. One should notice that a flow only receives this adjustment in the desired throughput from a particular router when that router is the bottleneck for that flow. It also generates and compares the calculated feedback to the packet's *Delta_Throughput* field, which is reduced if the current value exceeds the fair capacity allocation. Therefore, the receiver collects the minimal feedback allocation from the bottleneck router.

We emphasize that this functionality is the main contribution of the XCP framework, since an XCP router conveys an explicit notification of the bottleneck capacity allocation for each flow passing through the bottleneck router. Although further investigation on XCP performance it is necessary, apparently such contribution solved an old question to the Internet community that is how to calculate flow's fair share without keeping any per-flow state. To do that, an XCP router must execute periodically, at a controlled interval, two control algorithms: the efficiency controller, which is mainly responsible for the maximization of the outbound link, and the fairness controller, which is mainly responsible for fairly allocating bandwidth to flows [110]. In addition, as a side effect, an XCP router prevents its queue from building up to its limits.

The most difficult task of an XCP router is to compute the average RTT of the flows, since it needs to take into account the RTT of each flow and divide by the number of flows. Recall that an XCP router does not keep any per-flow state. The elegantly adopted solution was to take the average RTT over the packets normalized by the number of packets that a flow transmits during a control interval. As described in [107], Section 4.5, the average RTT is described by the following formula:

$$RTT_{avg} = \frac{\sum rtt_i \times \frac{s_i}{r_i}}{\sum \frac{s_i}{r_i}} \qquad \textbf{(2)}$$

where r_i is the flow's throughput, rtt_i is the flow's estimated RTT, and s_i is the packet size in bytes.

First, the Efficiency Controller (EC) observes the aggregate inbound traffic. As it does not need to worry about fairness issues, the EC only computes a desired increase or decrease in the aggregated traffic rate. To do that, the EC uses the following formula, which is computed at each control interval:

$$\phi = \alpha S - \beta \frac{Q}{d} \qquad (3)$$

where ***S*** is the difference between the input traffic and the output capacity, ***Q*** is the persistent queue size, and ***d*** is the average RTT, α and β are constant parameters. We refer the interest reader to [107], in order to understand the stability analysis that finds the value for α and β. One should interpret the above equation as a main signal related to the feedback, which is mainly proportional to the spare bandwidth and the queue level. In such situation, when S > 0, the link is underutilized and the EC should signal a positive feedback. Otherwise, the EC should signal a negative feedback. The second term in the equation is necessary to drain the persistent queue. One should observe that such feedback is not sufficient to achieve fairness yet, since the EC works only with the aggregate traffic. In other words, the EC is unaware of flows and has no function for distributing the aggregate feedback into them. It is not its job determining how flows should change their sending rates at each control interval. This work has to be done by the Fairness Controller (FC). Therefore, the FC distributes the feedback to individual packets in order to achieve fairness. The proposal policy to compute the per-packet feedback relies on the following rules firstly [107][109][110]:

If $\phi > 0$, distribute it equally to all flows.

If $\phi < 0$, distribute it to flows proportionally to their current throughputs.

We refer the interest reader to [110] to get an in-depth understanding about the derivation of the per-packet feedback distribution at the FC. It is worth stressing that all of the necessary parameters to compute the per-packet feedback are obtained at any XCP router straightforwardly, namely flow’s throughput information in the congestion header (CH), packet sizes in the IP header, aggregated traffic rates, and average RTT.

Sender Functionality

This Section describes the sender-side XCP mechanisms. The sender keeps four parameters:

- a requested throughput value,
- an updated throughput estimate,
- the maximum throughput allowed by XCP, and
- a current estimate of the round-trip time

Before sending a packet, the sender must fill the congestion header (CH). First, the sender fills the Throughput field in the CH with the current throughput estimate. Second, the sender sets the RTT field to a weighted average of the RTT estimate. Third, it calculates a desired change in throughput, which reflects the difference between the estimated and the desired rate. One should notice that if the sender does not have sufficient data to send to use up the available throughput, it must set the desired change to zero. Fourth, the sender then calculates the value of the *Delta_Throughput* field, by dividing the desired throughput change by the number of packets in one RTT. This last step is an important contribution of the XCP proposal, since it allows a XCP router not to retain any per-flow state. In other words, this per-packet distribution of the throughput change allows a XCP router treating each packet independently of others in the same flow [110].

As we described earlier, when acknowledge packets arrive at the sender carrying reverse feedback, the sender should react adequately. For instance, in a case of a TCP-like behavior, the rate adjustment could be made in the congestion window. Katabi suggests the use of the following formula:

$$cwnd = \max\left(cwnd + feedback \times RTT, MSS\right) \tag{4}$$

where

cwnd is the current congestion window,

feedback is the *Reverse_Feedback* field from the acknowledge received packet,

RTT is the Sender's current round-trip time estimate, and

MSS is the maximum segment size.

Katabi et al [107][110] also argue that when a sending application does not send data fast enough to fully utilize the allowed throughput, XCP should immediately reduce such parameter, to both reflect the actual rate and to avoid bursts of packets into the network. Although the proper behavior is still an ongoing discussion, to this point there are two possibilities for the algorithm used for aging the allowed throughput. The original one has the following formulation [107]:

For every RTT in such situation,

$$cwnd = 0.5 \times (cwnd - K) \tag{5}$$

where K is the number of outstanding packets.

A new proposal has the following formulation [110]: For every RTT in which the actual throughput is less than the allowed throughput, the allowed throughput must be reduced by a Moving Average (MA) smoothing technique:

$$allowed\_throughput = allowed\_throughput \times (1 - \alpha) + actual\_throughput \times \alpha \tag{6}$$

For $0 \leq \alpha \leq 1$, where α controls the speed of aging.

Finally, there is also an ongoing discussion about the best way for the RTT estimate, since this parameter plays an important role in the XCP framework. Additionally, the XCP sender should be ready to respond to occasional packet losses. The authors decided that the sender should react as the same way as TCP, since they assumed that a packet drop points out a congested non-XCP router in the network path of the flow.

Receiver Functionality

A XCP receiver end-system is just in charge of copying the congestion feedback notification from the network (i.e., the *Delta_Throughput* field value) that it finds in arriving packets into the *Reverse_Feedback* field of the Congestion Headers in the outgoing acknowledgment packets.

2.2.2. Scalable Signaling and Congestion Control with PTP and CADPC

The Congestion Avoidance with Distributed Proportional Control (CADPC) [238] mechanism relies on Performance Transparency Protocol (PTP) feedback packets. PTP is both a protocol and a framework for explicit performance signaling in packet networks [240]. Therefore, as CADPC depends on PTP packets, we will first describe the latter before giving an overview of the former.

Performance Transparency Protocol - PTP

The source of inspiration for developing PTP was the signaling scheme presented in the ATM ABR Explicit Rate Feedback. Welzl [238] first designed the PTP architecture keeping in mind that extra signaling from the network could improve congestion control. Additionally, scalability is an extra characteristic of PTP, since network routers do not need to perform any computation. In such architecture, routers just answer queries made by end-systems. In the PTP framework, end-systems query routers for Performance Parameters (PP), which can represent a network path property for a given flow. For instance, a straightforward property is the currently (or average) available bandwidth. However, there are three additional options for PP, such as Path Maximum Transfer Unit (Path MTU), Bit Error Ratio (BER), and Bottleneck Bandwidth.

There is a variety of ways in which PTP could be used. For example, the available bandwidth determination mechanism can possibly enhance the congestion avoidance or the slow start phase in TCP. One should notice that PTP is a network layer protocol. Hence, it is worth stressing that although PTP is an out-of-band signaling protocol, applications can completely control the amount of additional packets flooded into the network. In other words, application developers should follow a recommended restriction on the frequency of PTP packet submissions. Moreover, in our point of view, there is still no clear indication whether in-band (e.g., RED, ECN, and XCP) or out-of-band (e.g., PTP) signaling would be more appropriate to enhance congestion control in the Internet.

PTP packets can give extra support to transport protocols. In order to retrieve performance specific information from the PTP network routers, applications or transport protocols can make use of two basic methods:

- **Forwarding Packet Stamping**: PTP-compliant routers perform some inspection and manipulation on PTP packets. After that, they forward them to the receiver. In general, with the *compare flag* set to 0, a PTP router adds the requested Performance Parameter to PTP packets. If the compare flag is set, applications are interested to know whether the network will meet the traffic requirement described in the PP or not. In such case, PTP routers compare the sender's PP value with its own. When the PTP router has the worst value compared with the PP value, it should update it in order to indicate the presence of the network bottleneck.

- **Direct Reply**: in this mode, there is no need to add information into PTP packets. When a PTP packet encounters a PTP router that does not meet the requirement described by the PP, it should update its value and return the packet to the sender. Consequently, the *direct reply* mode abbreviates the feedback delay as the same way as the Backward ECN [193] and the ICMP Source Quench message.

Congestion Avoidance with Distributed Proportional Control - CADPC

Next we focus on the description of the CADPC, which is a congestion control mechanism based on PTP. The PTP framework is instantiated by using only the Available Bandwidth Performance Parameter and the Forwarding Packet Stamping method. As we mentioned earlier, Welzl [238] designed CADPC based on the assumption that the more information end-systems receive from the network, the better congestion control they can achieve. CADPC came fundamentally from the Congestion Avoidance with Proportional Control (CAPC) in the ATM ABR mechanisms, which describes a scalable switching function with no per-flow state.

CAPC is used to achieve convergence to efficiency by increasing the sending rate proportional to the amount by which the traffic is less than the target throughput. For the CADPC case, the main difference and contribution is the observation that instead of applications adapting their sending rate to the total current load, they consider the correlation between their current rate and the available bandwidth. The proposed CADPC control law has the following fluid model formulation:

$$x_i(t+1) = x_i(t)\left(2 - \frac{x_i(t)}{\beta(t)} - \frac{\lambda(t)}{\beta(t)}\right) \tag{7}$$

where

$x_i(t)$ is the sending rate of user *i* at time *t*;

$\lambda(t)$ is the measured traffic at time *t*;

$\beta(t)$ is the bottleneck bandwidth;

The above control law leads to convergence at the equilibrium point given by:

$$\lambda = \frac{n}{1+n} \tag{8}$$

The above formula states that link utilization converges to its maximum level, as the number of user (*n*) increases. Figure 4 shows the convergence behavior of CADPC efficiency versus the number of users.

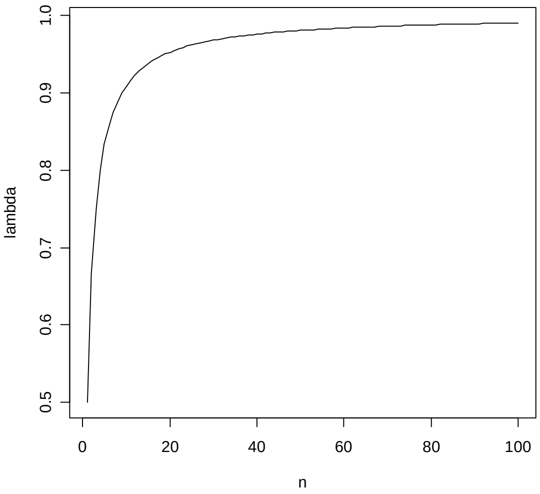

Figure 4 - Convergence of CADPC

We refer the interested reader to [238] and [240] in order to follow the derivation of convergence and stability analysis for the CADPC control law. Additionally, one can find the discrete model derivation along with performance analysis using the Network Simulator – ns-2. In this performance evaluation, the author presents CADPC behavior when facing a number of network scenarios, such as heterogeneous RTT, changes in routing paths, links with a high noise ratio or a large bandwidth delay product, different feedback delays, varying number of flows, impact of web traffic and AQM support.

Based on CADPC performance analysis, we now make some general remarks about its suitability for transporting multimedia objects in the Internet. First, CADPC with PTP has a low packet loss rate similar to XCP. In fact, XCP ensures to have no packet loss in a pure XCP network. CADPC also presents a smooth rate and a comparable throughput to several transport mechanisms in a number of scenarios. Such behavior seems to be highly advantageous for our objective. However, it is still necessary to understand its behavior under highly dynamic networks, e.g. with self-similar background traffic. Second, CADPC supports max-min fairness, which is a highly desirable property for our scenarios. It allows to get rid of all concerns about fairness at the application level when streaming multimedia objects (mainly video objects) over best efforts networks. It is worth stressing that by using the CADPC/PTP framework one can control the frequency of signaling messages, which in highly dynamic environments is an asset. However, it seems that it will be necessary to smooth the available bandwidth information before handing it over to the application level. In [144], Loguinov and Radha depicted important concluding remarks from this analysis, which some of them support the definition of our thesis scope. They analyzed the dynamics of a video streaming experiment, conducted between a number of unicast dialup clients, connecting to the Internet through access points in several U.S. cities, and a backbone video server. In that experiment, the clients streamed low-bitrate MPEG-4 video sequences from the server passing through distinct Internet routers. They unsurprisingly argue that it is extremely important to develop congestion control suitable designed for real-time multimedia streams that scales to a large number of concurrent users and can be employed incrementally with the existing TCP flows, presenting TCP-friendliness. At the time being, we advocate that XCP and CADPC are the best candidates for such task. Additionally, we strongly agree with the authors when they point out that future research should first address congestion control issues before real-time streaming becomes widely available in the Internet.

2.3. Schemes for Multimedia Streaming over the Internet

2.3.1. End-System based Approaches

Weber and Veciana [235] have recently provided a system level analysis of performance and design issues concerning rate adaptive networks. The main issue addressed in that work concerns how to allocate network bandwidth among the several active streams in order to maximize overall QoS. Starting from the premise that rate adaptive multimedia streams offers end-users the additional advantage of being robust to network traffic fluctuations, their work defines QoS in terms of the mean rate seen by the end-user, which is not a precise metric. Thereafter, they undertake the problem of identifying an optimal adaptation policy that maximizes QoS. They suggested an appropriate scaling regime for rate adaptive streams and identified asymptotic QoS for large capacity networks under the optimal adaptation policy. One should observe that such adaptation policy refers to the classical resource allocation problem [208]. That is the allocation of network capacity to the set of active multimedia streams, subject to the constraint that the aggregate allocation to all streams arriving on a given bottleneck link not exceed the link capacity. However, they argued that due to the infeasibility of implementation of such optimal adaptation policy, a potential proposed multi-class admission control policy could asymptotically achieve QoS levels similar to the optimal adaptation policy, but with no need for dynamic adaptation. We would like to emphasize that this multi-class admission control policy requires precise knowledge and tuning of system parameters (e.g. the rate adaptive scaling and the other parameters, the duration and maximum subscription level probability distribution functions). Hence, the main contribution of Weber and Veciana's work [235] is the analysis of QoS under the rate adaptive scaling and the multi-class admission control policy. On the other hand, some assumptions made by the authors limit the usefulness for deploying their proposal in real networks. The main drawback is that they considered the aggregated traffic to be approximately constant. Such assumption does not correspond to recent network traffic measurements made both in large backbone links and in access networks [95] [162] [163]. On the other hand, Seeling and Reisslein [197] undertook an interesting research study concerning video quality, traffic variability, and revenue issues for service providers. They opened up a new research topic in the Internet video area by considering that in future networks, the number of simultaneous video streams on a server will depend on both the mean bit rate as well as bit rate variability of each video stream. At the same time, they consider that the revenue in such scenario will also depend on both the number of supported video streams as well as their quality level. They examined the relationships between video quality, bit rate variability, and the utility from a streaming service provider with statistical multiplexing for open-loop encoded video i.e., with a fixed quantization scale. They focused on single layer encoded videos and described a methodology for the analysis of two goals. First, the maximum number of video streams supported on a link subject to a statistical quality of service criterion. Second, the obtained revenue when statistically multiplexing video of different quality levels over the link. Our work can extend their results by taking into account the dynamic behavior in the links and the use of scalable encoded video.

Cuetos and Ross [45] presented a framework that combines scheduling, FEC error protection, and decoder error concealment to address the issue of how to deliver layered video streams over a lossy packet network in order to optimize the video quality realized by the end-system. They analyzed scenarios related to channel with both perfect and deficient state information. We should mention that some elements in this framework are not mandatory in real environments, although they can be combined to achieve better performance. Moreover, one should observe that in a layered-encoded video scheme, the video is encoded into a Base Layer (BL) and several enhancement layers (EL). This method provides minimal rendered quality and additional decoded EL can progressively improve the rendered quality. At the sender side, the scheduler is responsible for delivering media packets and it may choose not to transmit some media packets, hence not sending some layers in some frames. Additionally, scheduling can be combined with some error correction technique, e.g. retransmission of lost packets or transmission of redundant forward error correction (FEC), to minimize the undesirable effects of packet loss. Therefore, the authors advocate that scheduling mechanisms and error correction should work “in cooperation” in order to adapt to the network conditions. At the receiver side, the decoder usually applies some error concealment (EC) technique to best conceal the missing packets [232]. Roughly speaking, EC consists in interpolating missing packets from the adjacent available ones. Since scheduling and error correction are optimized without taking into account the presence of error concealment at the receiver, the authors argued all these elements should be combined in a unified end-to-end manner. Therefore, the main contribution of this work is that optimizing together scheduling, forward error correction, and error concealment improves performance significantly. They also found that quality deterioration for a channel with imperfect state information was small within their framework.

Guo and Ammar [75] considered the problem of scalability in the distribution of live streaming of video content from a single server to a large number of clients. They focused their solution on a time-shifting video server, and a video patching scheme. The time-shifting video server sends multiple video streams with different time shifting values. Therefore, during stable network conditions, client receives the time-shifted video stream along as the original stream. The authors show that multicast clients can receive the complete video program even in an unreliable network infrastructure. They provide some indications that their solution has interesting properties for multicast streaming such as lossless video reception, stable video quality, continuous video streaming, and low complexity.

Arguing that UDP and TCP do not suit the real-time nature of video transmission, since UDP could lead to congestion collapse and TCP steady state throughput oscillates under normal conditions due to the AIMD algorithm, Balk et al. [13] presented the development of an end-to-end transport protocol called Video Transport Protocol (VTP). Its main goal was streaming video according to the characteristics of the network path. Following the same approach as we do in this thesis, Balk et al. focused only on video streaming in best effort networks. A major difference between VTP and most AIMD protocols (e.g., TCP, RAP etc.) is that a VTP sender performs the decreasing phase by adjusting its rate to the receiving rate observed at the receiver. The core of VTP’s rate adjustment relies on a new bandwidth estimation technique, which in turn applies an exponentially average of the bandwidth information samples. With this estimate in hand, the VTP sender can determine the speed of the outgoing data packets by using an algorithm. At this point, VTP proposal starts to show some drawbacks. First, its authors suggest that the weighting factor parameter should be close to one. We emphasize that with such parameterization it is not necessary to apply any smoothing technique, since the most important sample is always the last one. Moreover, a constant parameter cannot adapt to dynamic network conditions. Such limitation can compromise VTP deployment in the Internet. Second, the proposed algorithm cannot guarantee a TCP-friendly behavior whatsoever, although the authors showed VTP friendliness through some simulation experiments. Finally, we advocate that developing a new transport protocol with only one kind of application in mind will make its deployment highly infeasible. It is better to get an in-depth understanding of the existing protocols and identify their major pitfalls. By doing this, one can be more confident when proposing a proper solution.

2.3.2. Network based Approaches

Departing from similar motivation to ours, Kang et al. [105] elected Kelly's Congestion Control framework or Proportional Fairness [113] for achieving high-quality video streaming, since it provides stability, high link utilization, and fairness under various network conditions. They also introduced a novel framework for video streaming called Partitioned Enhancement Layer Streaming (PELS). In the PELS framework, applications mark their packets using different priority classes, allowing the network routers to prioritize packets based on such marks. Its authors argue that PELS provides an effective and low-overhead basis for multimedia streaming in the future Internet. Apparently, PELS is scalable since it does not require any per-flow management. However, it must operate in conjunction with priority-queuing AQM-based routers in network paths. We advocate that such tight restriction can impair further development or deployment, since it requires large changes in current Internet routers.

2.3.3. Adaptive Approaches

During the past few years, research on congestion control mechanisms and adaptive schemes for transporting multimedia in the Internet has grown significantly [18] [21] [44] [63] [65] [72] [120] [124] [140] [195] [196] [224] [227]. The main motivation in most proposals is that the high variability in the available bandwidth does not provide a fair environment for video delivery. In other words, at the receiver side, in order to provide the best quality video, a video stream requires relatively steady and predictable throughput. Hence, an appropriate mechanism should provide smoothed rates to applications.

Some of these research papers focus on developing a brand new protocol, whereas others build their solutions on the top of a well-known transport protocol, e.g. TCP or TFRC. For instance, the solutions proposed in [72], namely Adaptive Rate-based Control - RC and [13]

Video Transport Protocol (VTP) present a new protocol design and implementation that attempt to maximize application-level quality or improve fairness with TCP. On the other hand, the solutions presented in [21][44][120][196] rely solely on TCP in order to provide an adaptive scheme, whereas in [63][140][224][227] all algorithms take TFRC as the point of departure.

De Cuetos and Ross [44] investigate a solution for adaptive streaming to adapt to the short- and long-term variations in available bandwidth over a TCP-friendly connection. The framework applies to stored fine-grained scalable video and its main contribution is the proposal of an optimization formulation to solve an optimal streaming problem. Although they determine an optimal streaming policy under an unrealistically ideal knowledge of future bandwidth information, the authors argue that such policy provides upper bounds on the performance of real-time policies. Thereafter they suggest a real-time heuristic policy. Surprisingly, their experiments show that video quality fluctuations are in the same range for both TCP and TCP-friendly algorithms, which disagree with several recent related work [72][120][145][224]. We consider that by simplifying the original problem, in order to make the problem more tractable, the proposed solutions leaded to a number of pitfalls. First, they considered both Base Layer (BL) and Enhancement Layer (EL) CBR-encoded. In our opinion, such assumption does make the problem easily to deal with, but it is not realistic whatsoever, since a CBR-encoded video implies in high variability in quality. Second, they assume that the server knows how much seconds of pre-stored video is recorded at the client buffer, which is clearly a non-scalable approach. In addition to that, they suppose that losses only occur due to missing play out deadline at the client. In other words, they assume a highly reliable connection with no losses, which is almost impossible in today's transport protocols over best effort networks. We decide to work with XCP and CADPC, since they can virtually guarantee no packet loss. The real-time algorithm in De Cuetos and Ross' work [44] discards the assumption that the sender knows the available bandwidth a priori. However, it has two major drawbacks. It continues to depend on the knowledge of the client buffer level. Additionally, its efficiency depends on the fixed parameterization of the EWMA for the sending rate, which in turn is not at the same time scale of the available bandwidth estimation.

Kim et al [120] propose a rate adaptation mechanism on top of UDP, called Smooth and Fast Rate Adaptation Mechanism (SFRAM), which also uses RTP as an application-layer control mechanism. The authors argue that SFRAM provides suitable environment for video streaming. The adaptation mechanism of SFRAM uses RTT and packet loss measurements as main parameters. Finally, an error-control scheme based on SFRAM, called Network-Aware Error Control (NAEC), is able to get information about the network status and use it to help the encoder to select a proper error-control scheme. In other words, NAEC reacts to dynamic network status while reducing degradation of video quality. Although the authors advocate that SFRAM plays the same role as TCP's congestion control, there is no analytical evaluation or even simulation-based performance analysis that supports that such mechanism has max-min fairness property, as AIMD protocols do. Therefore, they cannot guarantee any inter or intra-protocol fairness as already proved in TCP, XCP, and CADPC proposals.

Several adaptation mechanisms for video streaming rely on TFRC [63] [140] [224] [227]. Although some of them present fair simulation results, Loguinov [145] demonstrated that it is not a proper choice for this kind of application. However, we decided to present some of these TFRC-based schemes, since part of the solutions inspired us when building our architecture.

In [224], Vieron and Guillemot designed a new TFRC-based protocol along with RTP/RTCP signaling. In order to estimate the available bandwidth, the novel protocol takes into account the multimedia characteristics in the estimation of its model parameters, e.g. RTT, timeout and congest event rate, which deals with variable packet size. Some experiments have shown that this protocol provides a fair estimation of the available bandwidth. The novel transport protocol predicts the available bandwidth periodically and feeds it to the video source. Thereafter, they designed a global rate control model that encompasses the source buffer model as well as the end-to-end delay constraints of real-time streams. They made another important contribution addressing source rate control issues, by considering different approaches from the "direct" translation of bandwidth predicted values into encoder rate constraints to the global model taking into account buffers occupancy and end-to-end transmission delay. The choice of the transport protocol is the main difference between this approach and ours. One should clearly observe that by choosing TFRC for streaming video, the authors had to develop several *patches* in order to overcome its defective behavior. To do that, the authors added an extra signaling based on RTP, to extract the main parameters for the bandwidth estimation. In our approach, by relying on protocols with explicit feedback notification, as XCP or CADPC, one should only focus on delivering such information to the application level. Therefore, the RTP/TFRC architecture did not consider any problem with TFRC parameterization (e.g., the history size parameter) which could lead to unexpected and undesirable throughput oscillation behavior. In a similar approach to Vieron and Guillemot [224], Grieco and Mascolo [72] designed an end-to-end rate-based congestion control mechanism for multimedia streaming.

2.4. Integrated Design for Congestion and Rate Control of Multimedia Streams

The Constant Quality Video Rate Control (CQVRC) scheme for MPEG-4 FGS tries to alleviate quality variations among consecutive frames in highly volatile scenarios, i.e. when the video source or transmission bandwidth varies greatly [204] [253] [254]. Designed to tackle the problem of MPEG-4 FGS video quality variability, the CQVRC utilizes an approach that exploits a larger decoder buffer, future frame information, and temporal scene segmentation for the Base Layer (BL). For the Enhancement Layer (EL), CQVRC inserts a small amount of rate-distortion (R-D) information for each bitplane and use the embedded R-D samples to interpolate the R-D curve linearly. After that, CQVRC applies an adaptive rate allocation. The resulting video stream is prioritized before being flooded into the network. CQVRC assumes there is a network with QoS features (e.g., IP DiffServ) for the prioritized stream. The authors argue that all system components, namely the FGS encoder, the CQVRC-based rate adaptation and packetization unit, along with error resilient decoding and differentiated forwarding, can be seamlessly integrated into a unique system.

Chapter 3. Performance Analysis of Streaming Multimedia Flows with Explicit Feedback Notification

As the capacity of Internet Service Providers (ISP) on both access and core network increases, it encourages a massive deployment of multimedia applications over the Internet. Moreover, the increasing demand for video streaming (e.g., video on demand) poses additional challenges to congestion management and quality of service, since the current best effort Internet cannot offer high-quality environment to end users. Hence, this scenario brings some challenges for the implementation of mechanisms that assure smooth rate variation (media friendliness) for applications as well as maintaining the TCP-friendly behavior. Although TCP is the dominant congestion control protocol in the current Internet, it is not suitable for transporting multimedia traffic, mainly because its window-based control mechanism leads to undesired reactions when congestion occurs, which implies in a high rate variation. Additionally, TCP-Friendly Rate Control Protocol (TFRC) [60] [61], which is a rate-based congestion control mechanism, offers a smooth throughput variation and it seems to be the preferred choice by the Internet community for transporting multimedia flows. However, recent research works have shown that although TFRC has a smoother rate profile, it is highly dependent on the configuration parameters (e.g., self-clock mechanism or loss history size parameter) [16] [234]. There are also some research papers indicating significant divergence between the throughput achieved by TFRC and that by TCP [16] [60]. Recently, Rhee and Xu [186] studied the limitations of TFRC, by examining how the main factors that determine its steady-state throughput (e.g. throughput equation, loss rate estimation, and RTO estimation) compel such discrepancy between TFRC and TCP sending rate.

In this chapter, we perform a quantitative evaluation of the end-user perceived media quality of video streaming under network friendly protocols in the best-effort Internet. First, we evaluate the effect of throughput variation of the TFRC Protocol on the Peak Signal to Noise Ratio (PSNR) of MPEG-4 video files. We show that although TFRC has a less aggressive behavior, it still exhibits (causes) oscillations under dynamic network conditions (e.g., under self-similar background traffic [55]). Furthermore, we advocate that any adaptive rate control for multimedia streaming flows should rely on precise explicit feedback notification from the network, in order to provide the streaming media with both uninterrupted transport services and low intensity variation in quality. Hence, we carried out a set of experiments under the Congestion Avoidance and Distributed Proportional Control framework (CADPC). Recall that CADPC is a transport and network-level approach that relies on Performance Transparency Protocol (PTP) packets in order to infer network conditions periodically.

Our results show that in a number of network conditions, such adaptive mechanism with explicit feedback provides low variation in the PSNR. Recall that the fundamental problem addressed in this thesis is how to stream video in the Internet with low quality variation while keeping max-min fairness on the network. The results presented in this chapter allow us to focus on only one part of the initial problem, namely quality variation (in Chapter 4).

Section 3.1 presents related work. Section 3.2 develops the basic idea behind explicit network feedback notification and describes its utilization under the CADPC framework. The following sections (Sections 3.3 and 3.4) show the performance evaluation of the TFRC and CAPDC protocols when streaming MPEG-4 video files. Finally, we draw some conclusions and present our published papers in Section 3.5.

3.1. Related Work

In Chapter 2, we presented a number of recent research papers related to the problem of transporting multimedia content over the Internet [15] [16] [60] [61] [72] [104] [119] [224] [234] [251]. We described the most important ones and showed that some of them focused only on the problem of transcoding (compression) techniques for end-systems [149], whereas others are more interested in application and/or transport level protocols for transmitting multimedia objects [15] [36] [61] [72] [119] [224] [234]. For example, in [224], Vieron and Guillemot described a new RTP-based TCP-compatible congestion control protocol that takes into account the multimedia flows characteristics, such as variable packet size and delays. Built upon TFRC the new protocol tries to get a more accurate estimation of the bandwidth model parameters. Kim et al. [119] proposed a rate adaptation mechanism called the smooth and fast rate adaptation mechanism (SFRAM), which is based on the transmission control protocol (TCP) throughput equation. The authors argued that by adaptively averaging measurements, SFRAM alleviates the undesirable throughput variation for video transmission. Moreover, they proposed an adaptive network-aware error control to alleviate error propagation due to packet loss. Grieco and Mascolo [72] also proposed an end-to-end rate-based congestion control algorithm for streaming flows over the Internet. By using control theoretic analysis, the proposed algorithm tries to predict accurately both the used bandwidth and the queue backlog in an end-to-end manner.

There are two main disadvantages on the above approaches. First, all proposed algorithms try to estimate (predict) the end-to-end available bandwidth accurately, in order to take some action in the application layer. We argue that such estimation (prediction) could not be precise in highly dynamic networks. Second, using TFRC as the basis for the construction of new protocols could have serious implications on the multimedia streaming quality [234] or even on the network stability [16] [251] and fairness [72].

In an important work presented by Zhang and Loguinov [251], it was demonstrated that window-based protocols have less packet loss under delayed feedback than their rate-based counterparts, but their performance deteriorates (e.g. with amplified oscillations) as buffering delay becomes large. This is an undesirable behavior for video streaming. They argue that multimedia in the future Internet will not benefit from oscillation-free congestion control unless the network deploys some form of Active Queue Management (AQM).

The study in [16] shows that the TFRC protocol is more stable than the other slowly responsive congestion control protocols, but it is highly dependent on the packet loss patterns. In [234] Wang et al. also evaluated TFRC's media-friendliness, which is an attribute of any congestion control protocol that takes into account the characteristics of streaming media and provide it with uninterrupted transport services. They concluded that TFRC fails to prevent abrupt sending rate reduction during transient workload increases and causes fairness issues when competing with TCP traffic. In addition, they argue that a common approach to overcome this issue, such as increasing loss event history size or removing self-clocking in TFRC, can only give TFRC slight resistance to transient changes.

The idea of getting precise information (e.g., available bandwidth or congestion indication) from the network is not new. Several approaches related to explicit feedback notification have been discussed in the Internet research community for years. However, apart from the fact that traditional approaches (e.g., Available Bit Rate (ABR) and Resource Management (RM) cells in ATM networks [88], Explicit Congestion Notification (ECN), Random Early Dropping (RED), and IP Source Quench Messages (SQM)) could enable an end-system to adapt its rate more efficiently, some research studies have recently pointed out the necessity and trends towards explicit signaling in a stable and scalable fashion [237] [238] [251].

There are studies related to the performance of window-based congestion control protocols over an explicit bottleneck rate feedback networks [102] [106] [201]. The motivation behind the work in [201] is to understand if TCP over an explicit rate control (ATM/ABR) could enhance its end-to-end throughput performance. In that work, Shakkottai et al studied two explicit rate feedback schemes. With INSTCAP (*instantaneous capacity feedback*) method, the short-term average capacity of the bottleneck link is fed back to the end-systems, whereas with EFFCAP (*effective service capacity*) only the long-term history is fed back. The most important conclusion from the analysis and simulation results is that the throughput of TCP over ABR depends on the relative rate of capacity variation with respect to the RTT of the network path. They found out that for slow variations of the link capacity the improvement with INSTCAP achieves 25%–30%. Otherwise, the throughput can be slightly worse than with TCP alone. On the other hand, EFFCAP rate feedback achieves higher throughputs than INSTCAP, always beating the throughput of TCP alone. The main drawback is that EFFCAP computation involves two parameters, which must be tuned properly, namely the M block of samples and N sliding blocks. In a continuation of Shakkottai's work, Karnik et al [106] propose Rate Adaptive TCP (RATCP). They study and compare the performance of RATCP and TCP with the same goals as Shakkottai's work that is to understand the dynamics of rate feedback and window control. In other words, their work tries to understand the performance limits of providing precise feedback directly to TCP sources other than implicit feedback (i.e. packet losses). RATCP changes TCP's behavior in order to utilize rate feedback effectively. For the performance evaluation, Karnik made an important assumption. They assumed that the network is by some means capable to calculate and feedback fair rates to end-systems. One should notice that such behavior is intrinsic to XCP. Although those are reasonable assumptions in the context of XCP and CADPC, Shakkottai et al did not offer any new solution on how to provide such calculations. However, they provided some important concluding remarks that we consider in the design of our novel architecture. First, there is an important effect of time scales of rate variations compared to the RTT of the network path. When the rate variations are slow compared to the RTT, precise feedback information is effective and enhances end-system performance. On the contrary, when the RTT is large and rate variations are fast, the end-system performance does not improve. We refer the interested reader to [106] for a precise definition of the terms slow and fast in this context. Second, RATCP deals effectively with random losses on the link, since it differentiates between congestion and corruption losses leading to higher throughputs. XCP and CADPC also claim the same behavior. In addition to that, RATCP ensures fairness among competing flows even if they have different RTT. This statement supports our design decision that it is viable to look for transport protocols that can give available bandwidth information that is suitable for multimedia streaming. In other words, our rationale changes the focus to the maximization of quality instead of transport issues (i.e., max-min fairness and efficiency).

In [130] Lakshman et al propose a scheme for transporting compressed VBR video traffic using explicit-rate congestion-control mechanisms proposed for the ABR service in ATM networks. In their approach, the video sender tries to match the encoder rate to the available bandwidth by modifying the quantization level during compression. They also propose a new rate-allocation scheme in the network based on a weighted max–min fairness criterion. As a general concluding remark, they state that transporting video using the enhanced explicit-rate-based feedback control has the potential to combine the best features of VBR, CBR, and RCBR video.

In such a case, we advocate that some algorithms, including either CADPC [237] [238] or XCP [53] [107], could form a reliable framework to build application-level rate-based protocols to supply multimedia flows with reduced oscillation in throughput. We only would like to emphasize that a new research should focus on today's Internet technology, but we can learn important lessons from the past. In this work, many research papers from ATM technology gave us good guidelines for developing new ideas.

Recall that CADPC receives periodic performance feedback information by using explicit out-of-band signaling. In fact, it is a distributed variant of the Congestion Avoidance with Proportional Control (CAPC) used in ATM networks. It has some desirable characteristics for streaming multimedia content in best-efforts networks, such as smooth rate and small queue length. Moreover, it quickly reaches a stable state and requires only sporadic signaling packets. In general, a CADPC sender adjusts its sending rate by increasing or decreasing it proportionally to the relationship between the rate of the sender and the available bandwidth in the network path. Signaling is carried out using the Performance Transparency Protocol (PTP), which was designed to retrieve performance related information or performance parameters (e.g., the average bottleneck queue length or the maximum expected bit error ratio) from the network. Although PTP could be seen as an IP layer service, it is actually layered on top of IP. A header and several datasets form a PTP packet. There are two available querying methods, namely Forward Packet Stamping and Direct Reply. An Echo flag in the header supports both methods. In Forward Packet Stamping mode, PTP packets carrying information requests are sent from the source to the destination and updated (if the "Compare flag" in the header is set to 1) or added (if the "Compare flag" is set to 0) by all intermediate PTP-compliant routers. Hence, the receiver can assemble the relevant information from the network and feed it back to the sender. In Direct Reply mode, a PTP packet contains only one dataset with values set by the sender according to its information requirements. Each PTP-compliant router along the network path compares such values with its own recent measurements. The first router that does not meet the requirements updates the dataset and immediately returns the packet to the sender.

3.2. Performance Evaluation Scenario

Following the guidelines in [199], this Section describes the scenarios used during our simulations. In next Section, we will present a numerical evaluation of the effect of throughput variation of both TFRC and CADPC protocols on the PSNR when transmitting MPEG-4 video files over a best-effort network. To do that, we assessed the video quality based on the perceived quality by users at the end-system.

One should note that there are two approaches to measure video quality, namely subjective quality measures and objective quality measures. The former usually takes the impression of the user watching the video, which is prohibitively expensive. The latter tries to mimic the quality impression of the human visual system using quantitative metric. Such an approach is highly suitable for emulation and simulation experiments. It is worth stressing that a straightforward evaluation of network metrics (e.g., mainly throughput, but also delay and jitter) does not ensure a fair media-friendliness analysis, since packets that contain key video frames (e.g., I frames in MPEG-4) could be discarded (dropped). In this situation, the decoder becomes unable to reconstruct such key frames despite having received some additional ones (e.g., P and B frames) hence leading to a higher throughput at the application level, but a lower user perceived video quality. Therefore, to overcome this issue, we use the most widespread objective metric in this context – PSNR - to evaluate the video quality at the receiver. The computation of the objective video quality PSNR is performed on an image-by-image basis.

The objective video frame quality is the difference between the unencoded original video frame and the encoded video frame. The PSNR uses as the main parameter the Root Mean Squared Error (RMSE) between the pixels of the unencoded and encoded video frame. Let $F_n(x,y)$ be the luminance value of an individual pixel in the nth original video frame at position *(x, y)*. Also, let $f_n(x,y)$ be its encoded value for the same pixel, where X and Y represents the resolution in pixels in each the video frame, which is constant for all frames in the video sequence [46]. Therefore, the RMSE for an individual frame has the following formulation:

$$RMSE_n = \sqrt{\frac{1}{XY}\sum_{x=0}^{X-1}\sum_{y=0}^{Y-1}\left[F_n(x,y) - f_n(x,y)\right]^2} \tag{9}$$

We calculate the PSNR as follows:

$$q_n = 10\log\frac{(255)^2}{RMSE_n} \tag{10}$$

We are aware that there are some critics to objective metrics, such as the PSNR, mostly due to lack of good correlation with perceived quality measurement. However, this is still a topic of ongoing research and there is no consensus for better objective metrics [233].

We relied on EvalVid for our simulation experiments. EvalVid is a trustworthy framework and a toolkit for evaluation of the quality of stored videos transmitted over either real experimental networks or simulation environments [121]. We also carried out our simulations using the network simulator ns-2 [216].

Figure 5 depicts the network topology used in our simulations. It consists of a video server and a correspondent client. It also has a number of traffic sources aggregated in one node (Sender), in order to generate highly dynamic background traffic [55]. They are connected to a router that in turn sends traffic to each destination through the bottleneck link. All access links have a fixed capacity of 10 Mbps and delay of 1ms. The capacity of the bottleneck link is 1.5Mbps and its delay is 10ms.

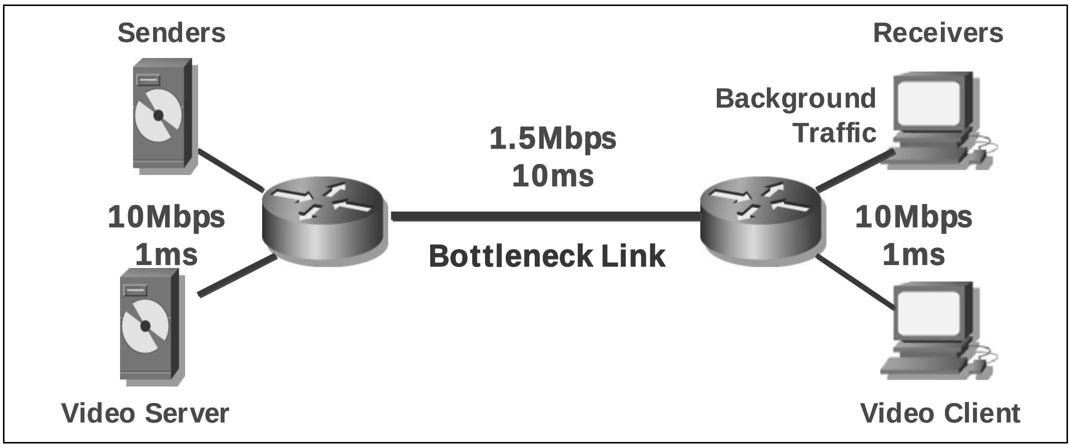

Figure 5 – Simulated Network: a Dumbbell Topology

For each simulated experiment, we collect video frames at the destination node, labeled "Video Client". For each scenario, results represent the mean measurement for a number of replications (at least 100 of simulation runs). For each mean value, we determined the 99% asymptotic confidence limits. Due to the narrow width of the confidence intervals, indicating a precise estimate of the mean value, there will be no need to show them in later figures in this chapter. We follow the guidelines in [92] for simulation in dynamic environments involving stochastic processes (i.e., Monte Carlo's simulation). With the combination of the selected parameters variation and replications, the number of simulations achieved 4000 runs.

Background Traffic

We also varied the background traffic characteristics, such as the amount of the aggregated fractal traffic, i.e., the background target rate. Recall that our objective is to understand whether explicit congestion control protocols are suitable for multimedia streaming in highly dynamic environments. In this chapter, we narrow our focus on the evaluation of quality variability, since max-min fairness and efficiency issues have been already addressed by the original proposals.

However, before exploring protocol behaviors when facing highly dynamic background traffic, we shed some light on the generation of proper self-similar traffic in network simulations. We consider this topic very important since many papers do not consider such behavior.

Recently, researchers identified some evidences of self-similar (or fractal) behavior in computer network traffic, as well as its severe implications in network performance [41] [133] [188] [191] [221] [244]. Under such condition, router's queue works at a high level of occupancy, mainly due to presence of burst traffic in several time-scales leading to a higher end-to-end delay and packet losses [101]. Consequently, this phenomenon could lead to a low-level utilization of the communication links. Therefore, an in-depth knowledge of the self-similar nature in network traffic and the identification of its characteristics or implications in different scenarios and network topologies are vital for carrying out network management activities, keeping QoS assurances in suitable levels, making traffic engineering decisions work and designing networks efficiently. Additionally, considering a simulation environment, the performance evaluation of network protocols and mechanisms under proper conditions is vital for obtaining reliable results [103]. The selection of representative scenarios in computer networks must include the exploitation of fractal traffic.

There are some known analytical methods for the generation of synthetic self-similar traffic. However, due to the complexity of a physical interpretation, an alternative construction closer to real traffic models in computer networks is based on the aggregation and superposition of On/Off sources [3], which activity and/or inactivity periods follow a heavy tailed probability distribution function (PDF). There are related studies concerning the aggregation of heavy tailed sources and self-similar traffic. Some of them present results and analysis of traffic measurement in real networks [244], whereas others focus on purely statistical perspectives [94] [142]. In a recent work [55], we analyzed the trade-off between accuracy and computational efficiency on the generation of fractal traffic. The precision was determined by evaluating the error between a target Hurst parameter (usually used to measure the self-similarity level) and its actually estimated value from the traffic sample collected during a simulation experiment. The computational efficiency concerns to the processing time needed to obtain a previous chosen precision. An important feature on the generation of self-similar traffic is associated to the relation between the form (shape) parameter of the heavy tailed PDF and the Hurst parameter H. For instance, for the Pareto PDF, analytical and empirical procedures show the relation $\alpha = 3 - 2H$ [41], where α is its form parameter.

The main concept related to self-similarity or general fractal behavior consists of the phenomenon of preserving the major characteristics of an entity in nature when observed in distinct time or space scales [19]. Particularly, in the case of stochastic objects such as the time series (e.g., computers network traffic), the self-similar behavior exhibits the same structural properties in several time scales. Without a suitable strong statistical approach, one should now assume if a realization of a stochastic process is aggregated in distinct time scales and keep its most important statistical properties (e.g., first- and second order moments), it is considered a fractal process.

Self-Similar Processes

Let be $X(t)$ a strict-sense stationary time series, with mean μ, variance σ^2 and autocorrelation function $\rho(\tau)$. Additionally, let $X^m(t)$ be a new time series obtained from $X(t)$, through averaging it in non-overlapping blocks of size m. In other words, the aggregated series has the form $X^m(t) = (m^{-1})(X_{tm-m+1} + X_{tm-m+2} + \cdots + X_{tm})$ and $\rho^m(\tau)$ is the autocorrelation function. The process $X(t)$ is considered self-similar if $\rho^m(\tau) = \rho(\tau)$ for any $m = 1,2,3,\ldots$. In particular, if the autocorrelation function has the form $\rho(\tau) \to \tau^{\beta} L(\tau), \tau \to \infty$, where $L(\tau)$ is slowly varying at infinity, one could say that it is a self-similar process with a Hurst parameter H. The relation between the Hurst parameter and the decaying rate of autocorrelation function β is $H = 1 - \beta / 2$. This kind of process exhibits Long-Range Dependence (LRD), which implies the autocorrelation function is not limited, that is $\sum_{\tau} \rho(\tau) \to \infty$. Another important property is related to the variance of the aggregated series that has a slow decrease as the aggregation level increases. Such characteristic could be used to estimate the self-similar level of a stochastic process. There are evidences that the LRD feature is firmly associated to heavy tailed behavior of the generating process. Additionally, the superposition of several independent heavy tailed sources yields self-similarity [244].

Several empirical and analytical studies show evidences related to the phenomenon of self-similar in computer network traffic [41] [51] [133] [166] [191] [221] [244]. Some approaches showed that aspects such as file sizes in Web servers and file transfer times under HTTP caused unfavorable impact in network performance. Such characteristics lead to traffic bursts in several time scales, which make it difficult the determination of efficient algorithms of congestion control, admission control and traffic prediction [62] [132]. For instance, in the presence of LRD traffic, increasing queue lengths does not produce fewer packets loss rates [73], as would be expected for traffic with short-range dependence. In addition, performance is seriously affected due to the high concentration of congestion periods and significant increase in queue delays [166]. Therefore, the traditional traffic source models, such as Poisson and Exponential PDF, which superposition does not exhibit self-similarity, must be replaced for more accurate models in order to obtain reliable simulation results [41]. For this reason, usual performance metrics, such as throughput, delay, jitter, packets loss and queue lengths, must be evaluated taking into account these evidences as support for obtaining coherent results.

As we have shown before, the Hurst parameter determines the self-similarity level of a time series. If *H* is in the [0.5, 1] range, there is a clear indication of the presence of self-similar behavior. In addition, *H* values closer to the unity point out a high self-similarity level. There are a number of methods to estimate the *H* parameter, which could be classified in heuristic and inference-based ones. Heuristic methods are mainly useful as simple diagnostic tools and the best-known one is the analysis of the rescaled range R/S statistic. Other techniques include the log-log correlogram, the log-log plot of the variance of the aggregated processes versus the aggregation level, least squares regression in the spectral domain and inference by maximum likelihood estimation in time and spectral domain (e.g., Whittle's estimator).

In order to exemplify some self-similarity level estimation methods, we briefly describe the R/S statistic and the variance techniques [19]. The R/S statistic is related to the *H* parameter by $E[R(n)/S(n)] \approx cn^{H}$, when $n \rightarrow \infty$ and c is constant and independent of n. It is easy to notice that $\log(E[R(n)/S(n)]) \cong H\log(n) + \log(c)$. This equation has the form $y = a + bx$ and consequently H could be estimated by linear regression, where $\hat{H} = \hat{b}$. Using the variance approach, the relation between the logarithm of the variance of the aggregated process $X^{(m)}$ and the block size m has the form $Var(X^{(m)}) \approx am^{-\beta}, m \rightarrow \infty$. As a result, $\log[Var(X^{(m)})] \approx -\beta \log(m) + \log(a)$ and *H* is estimated by linear regression that determines the negative slope β with $\hat{\beta} = 2 - 2\hat{H}$.

Due to the importance of the fractal behavior in a number of areas (e.g., economy, telecommunications), several formal analytical models have been proposed which most of them are useful for generating such sequences. Some of them rely on Fractional Autoregressive Integrated Moving Average (FARIMA) processes [142], Fractional Gaussian Noise (FGN) and Wavelets [94]. However, using these approaches lead to difficulties in getting some physical meaning for network engineers and computer scientists. In order to address this issue, an alternative proposal that has a meaning close to real networks is based on the aggregation and superposition of Renewal Rewards Process (On/Off) [189], which activity (On) and inactivity (Off) periods follow a heavy tailed PDF.

The M/Pareto process, also known as Poisson Pareto Burst Process – PPBP [3], is an excellent model that we used for precise fractal traffic generation at the same time as maintaining the understanding of the physical process existing in local or wide area networks. The M/Pareto is a process composed of a number of overlapping bursts. Bursts arrive following a Poisson Process with rate λ and have a Pareto distributed duration. Increasing λ implies to an increase in the level of activity of individual sources or in the number of sources. Each burst has a constant rate r and its length has the form $P_r(X > x) = 1 - F(x) = x^{-\alpha}/\delta, x \geq \delta$, with $1 < \alpha < 2, \delta > 0$, where δ is the scale parameter. It is easy to verify that the mean amount of work arriving in the PPBP model is $\mu = (\lambda r \delta \alpha)/(\alpha - 1)$. It also is asymptotically self-similar with H parameter $H = (3 - \alpha)/2$, where α is the form parameter of the Pareto PDF.

In our slightly different model of M/Pareto, we can set the source average aggregate rate. Hence, during the activity periods (On), each source sends data at a rate of $4/n$ Mbps, where n is the number of simultaneous sources. On and Off average duration times follow either the Pareto, Weibull or Lognormal distributions, resulting in an average rate of $r = 2/n$

for each source. This model is comparable to the M/Pareto model. However, in our On/Off model, we fixed the number of sources, which send several bursts with random duration. On the other hand, the M/Pareto model uses a random number of sources (a Poisson process), but each source generates only a single burst with random duration.

Simulation: Configuration and Parameterization

In this chapter, for each protocol (TFRC and CADPC) we set the target rate for the background traffic at 25% or 75% of the bottleneck link. We choose these values since they can represent mild and high impact on the multimedia flows respectively. In all simulations, the Hurst parameter was set to 0.85, which seems to be a common fractal level in access networks [188] [191].

Figure 6 presents the original PSNR video signal in dB that we use in all simulations in this chapter. The video trace file consists of 1061 frames and its average signal level is around 26 dB. One should notice that we need this original video signal information since we will compare it with the resulting video quality at the receiver, when transmitted under different transport protocols.

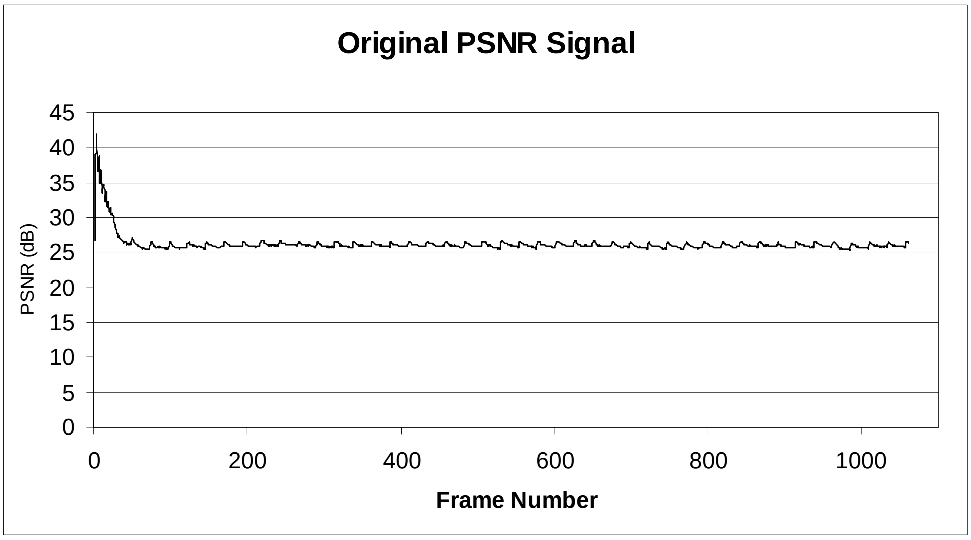

Figure 6 - Original Video Signal

3.3. Video Streaming with TFRC

For TFRC evaluation, we varied some factors (protocol parameters) in order to get an in-depth knowledge of their influence on the PSNR. We selected the Loss History Size and the Self-Clocking parameters, as these ones were pointed out to have some drastic influence on the TFRC sending rate [16] [234]. The Loss History Size parameter affects TFRC's responsiveness to persistent congestion, whereas Self-Clocking helps TFRC to achieve faster response to incipient congestion [61]. We set the value for the Loss History Size as 8 or 128 and turned the Self-Clocking parameter either on or off [234]. Table 1 lists all TFRC parameterization options along with the labels used to identify them.

Table 1 – List of Scenarios: TFRC parameterization

TFRC Label	History Size	Self-clock	Bottleneck Delay	Background Traffic
TFRC#1	8	ON	10 ms	25%
TFRC#2	8	ON	100 ms	25%
TFRC#3	8	ON	10 ms	75%
TFRC#4	8	ON	100 ms	75%
TFRC#5	8	OFF	10 ms	25%
TFRC#6	8	OFF	100 ms	25%
TFRC#7	8	OFF	10 ms	75%
TFRC#8	8	OFF	100 ms	75%
TFRC#9	128	ON	10 ms	25%
TFRC#10	128	ON	100 ms	25%
TFRC#11	128	ON	10 ms	75%
TFRC#12	128	ON	100 ms	75%
TFRC#13	128	OFF	10 ms	25%
TFRC#14	128	OFF	100 ms	25%
TFRC#15	128	OFF	10 ms	75%
TFRC#16	128	OFF	100 ms	75%

The following results present the average PSNR, for the first 300 out of 1000 frames and at least 100 simulation runs, received at the Video Client node using TFRC. In all simulations, we observed that network achieved its steady-state behavior before the 200th frame, but in most cases before the 100th frame. Therefore, in order to make graphics clearer, we decided to show only the first 300 frames. Additionally, in order to get the big picture of the simulation results, we present the obtained Empirical Cumulative Distribution Function (ECDF) for each parameterization. It is worth stressing that we also decided not to show the confidence intervals, since there is no superposition between the original PSNR signal and those from the TFRC simulations. This choice favored the aesthetic and clean presentation of results.

Figure 7 presents the simulation results for TFRC#1 and TFRC#3 parameterization. For TFRC#1 we turned self-clocking ON, set the History Loss Size to 8, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 10ms. For TFRC#3, we adjusted the self-similar background traffic to 75% of the bottleneck capacity. Figure 8 shows the ECDF for the same TFRC parameterization. With a small history size, TFRC has an unacceptable performance under high or low level of background traffic. For both scenarios (parameterization instances), it achieves a mean level around 15dB, which is 10dB less than the original signal.

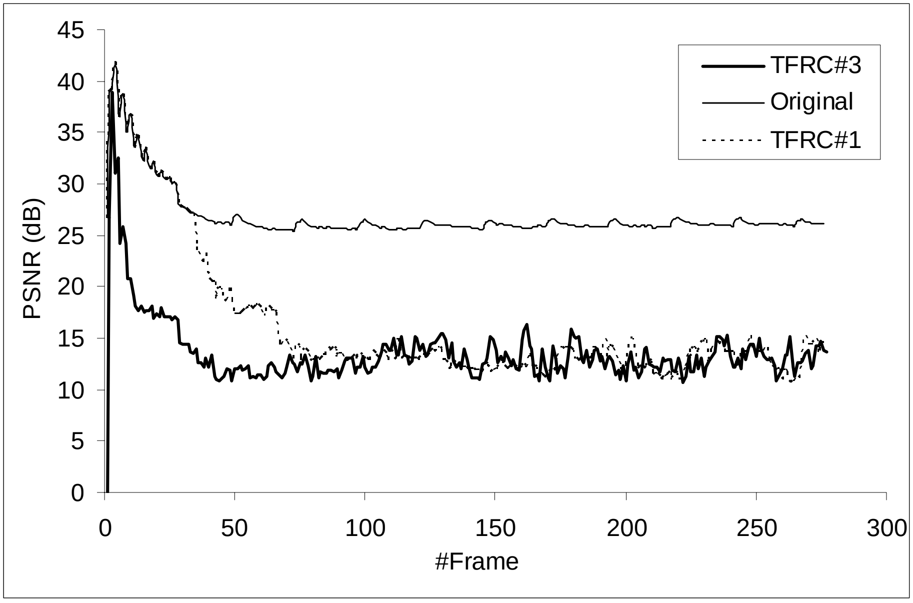

Figure 7 - TFRC#1 and TFRC#3

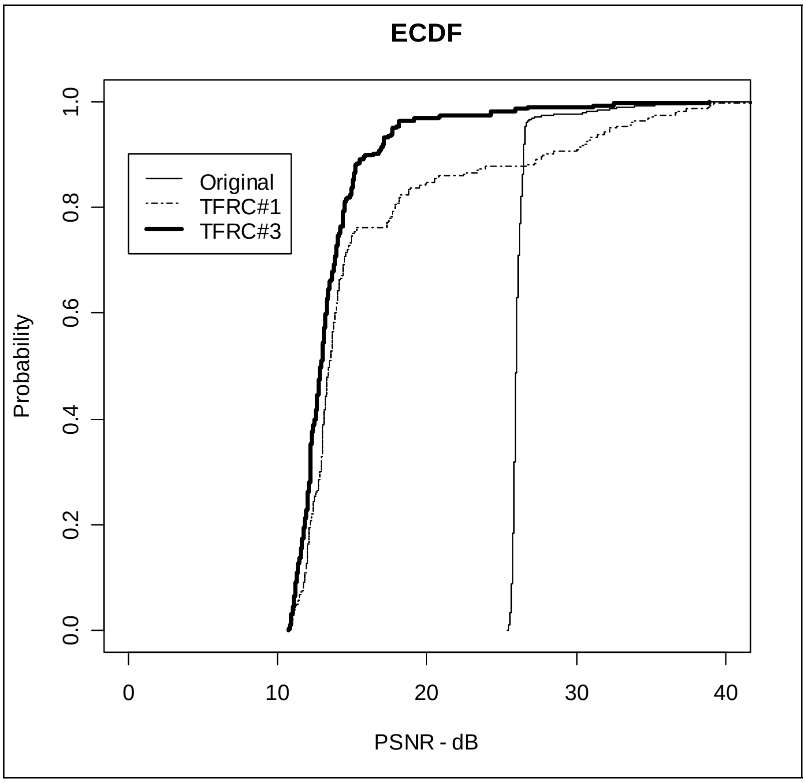

Figure 8 - ECDF - TFRC#1 and TFRC#3

Figure 9 presents the simulation results for TFRC#2 and TFRC#4 parameterization. For TFRC#2 we turned self-clocking ON, set the History Loss Size to 8, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 100ms. For TFRC#4, we adjust the self-similar background traffic to 75% of the bottleneck capacity. Figure 10 shows the ECDF for the same TFRC parameterization. As similar to TFRC#1 and TFRC#3, which has a small history size, TFRC also has an undesirable performance in a steady-state time, under either high or low level of background traffic. The main difference from the previous results is that with the background traffic set to 25% of the bottleneck capacity is that the network delay influences in the responsiveness of the TFRC sender. As it reacts slowly, it will only adjust by reducing its sending rate with a delay, as shown in Figure 9. Once more, for both parameterization instances, it achieves a mean level around 15dB, which is 10dB less than the original signal.

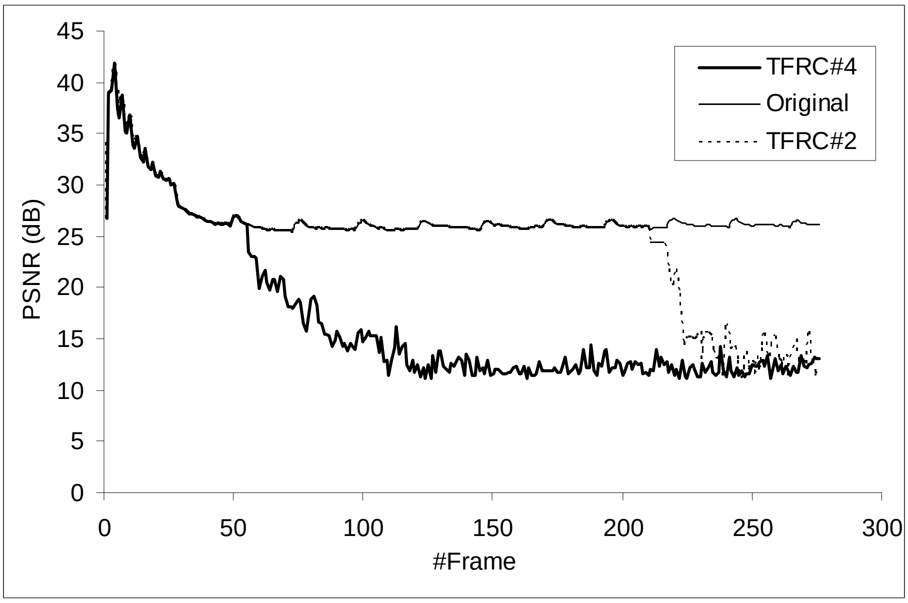

Figure 9 - TFRC#2 and TFRC#4

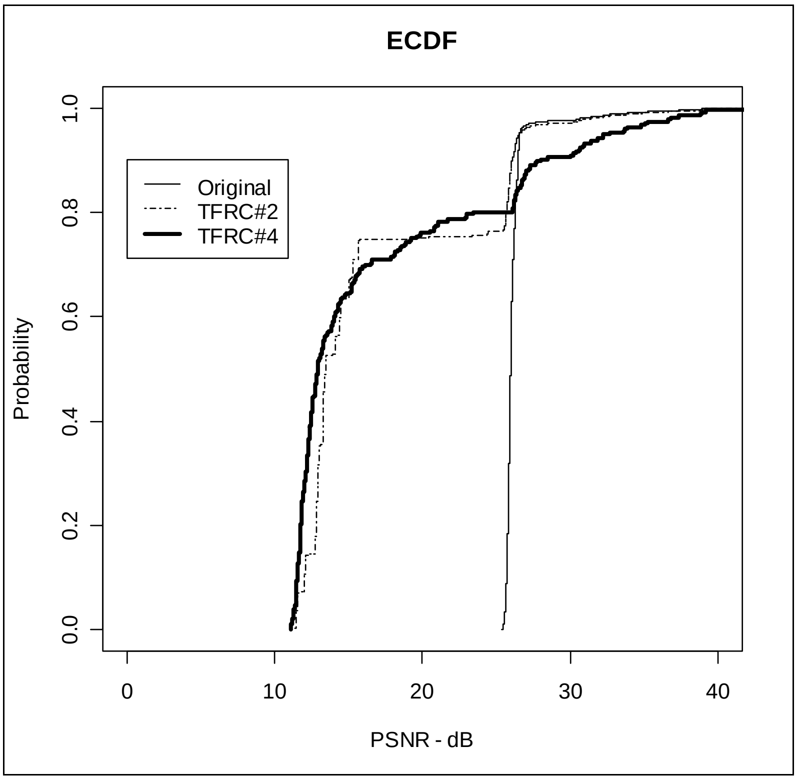

Figure 10 - ECDF - TFRC#2 and TFRC#4

Figure 11 presents the simulation results for TFRC#5 and TFRC#7 parameterization. For TFRC#5 we turned self-clocking OFF, set the History Loss Size to 8, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 10ms. For TFRC#7, we adjusted the self-similar background traffic to 75% of the bottleneck capacity.

Figure 12 shows the ECDF for the same TFRC parameterization. Similarly to TFRC#1 and to TFRC#4, TFRC5 and TFRC#7 achieve an undesirable performance in a steady-state time, under either high or low level of background traffic. The main difference from the previous results is that with the self-clocking parameter turned off, it has slow response to incipient congestion, thus achieving lower sending rate levels.

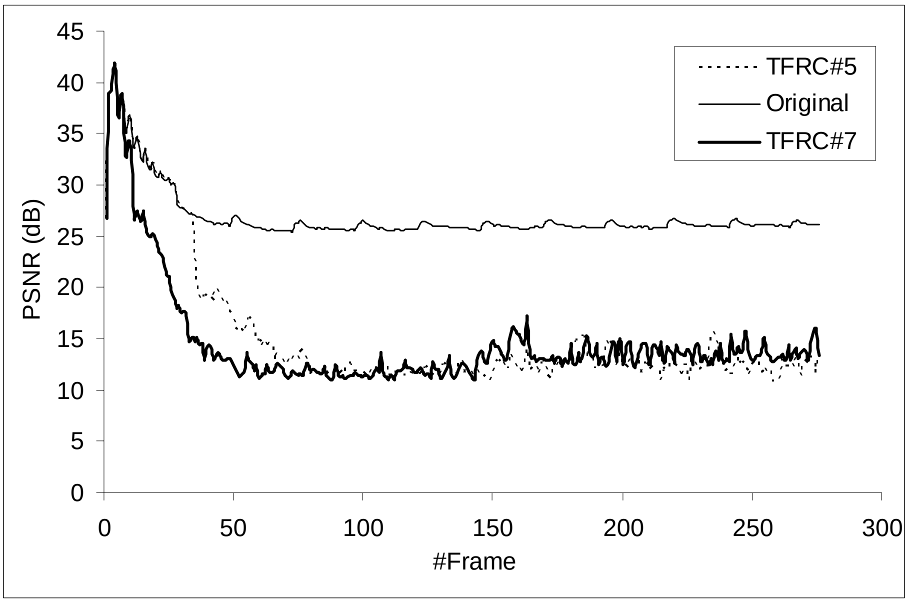

Figure 11 - TFRC#5 and TFRC#7

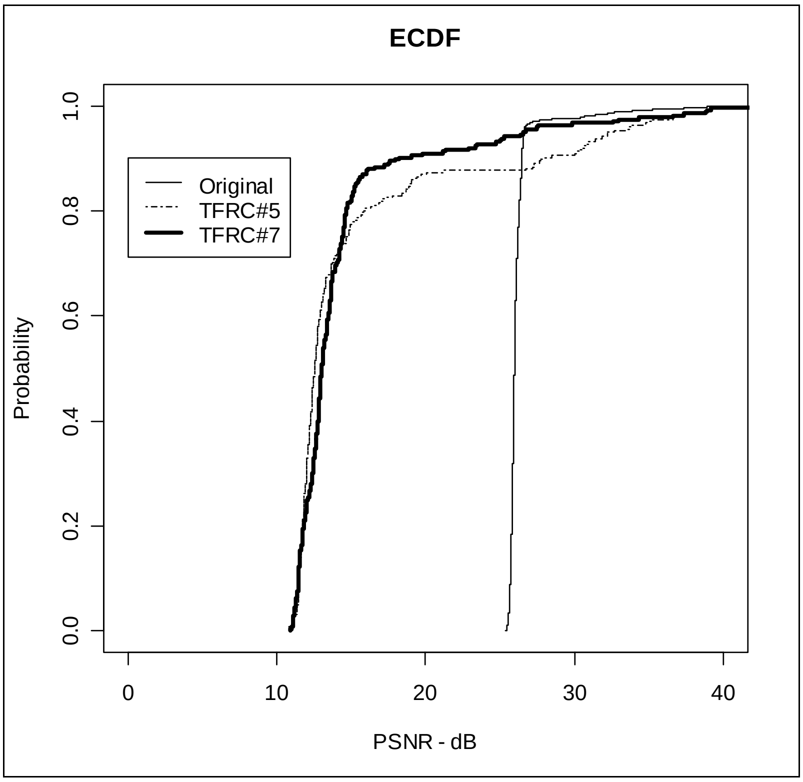

Figure 12 - ECDF - TFRC#5 and TFRC#7

Figure 13 presents the simulation results for TFRC#6 and TFRC#8 parameterization. For TFRC#6 we turned self-clocking OFF, set the History Loss Size to 8, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 100ms. For TFRC#8, we adjusted the self-similar background traffic to 75% of the bottleneck capacity. Figure 14 shows the ECDF for the same TFRC parameterization. As we adjusted the network delay to 100ms, there is no significant difference in the overall performance. From the analysis of all previous results, we observe that the self-clocking and the history size parameters control TFRC performance strongly.

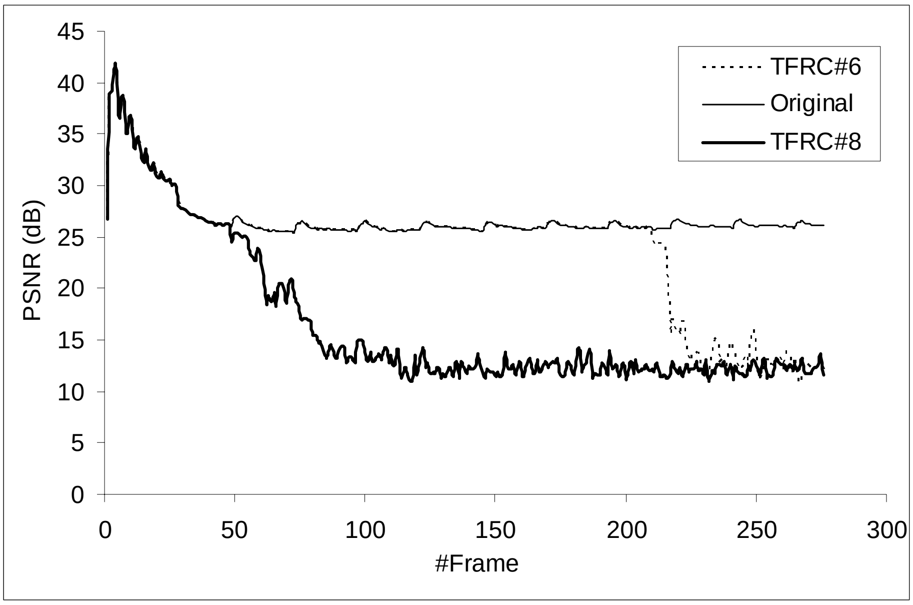

Figure 13 - TFRC#6 and TFRC#8

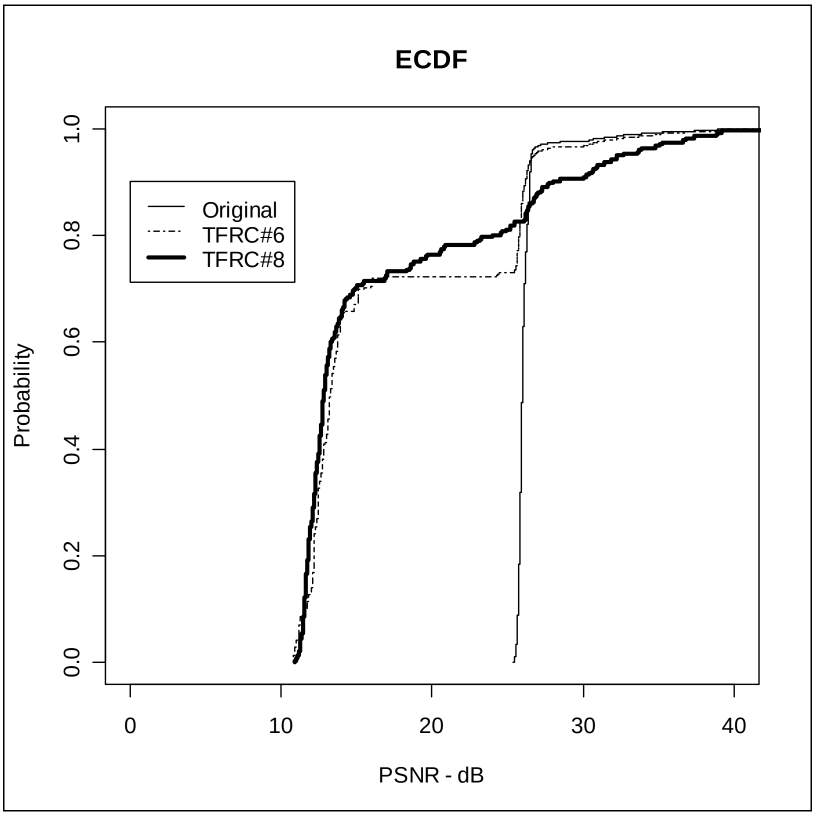

Figure 14 - ECDF - TFRC#6 and TFRC#8

Figure 15 presents the simulation results for scenarios TFRC#9 and TFRC#11 parameterization. For TFRC#9 we turned self-clocking ON, set the History Loss Size to 128, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 10ms. For TFRC#11, we adjust the self-similar background traffic to 75% of the bottleneck capacity. Figure 16 shows the ECDF for the same TFRC parameterization. We expected that with this parameterization, i.e. with Self-Clocking on and the History Size parameter set to 128, TFRC would achieve its best performance. Surprisingly, its performance continues poor as indicate in

Figure 15 to Figure 18. These results corroborate with recent performance evaluation of TFRC under several network loads, as first pointed out in [16] and [234]

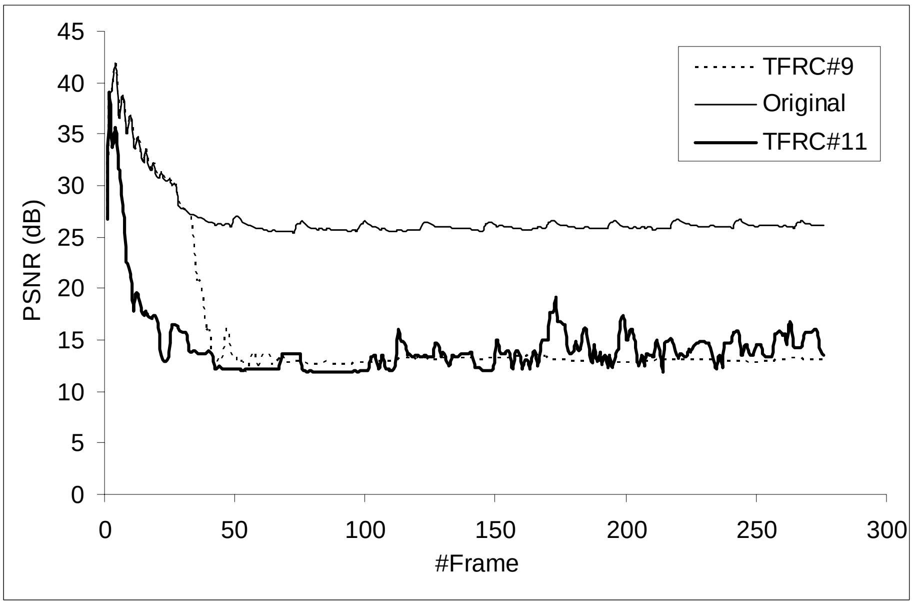

Figure 15 - TFRC#9 and TFRC#11

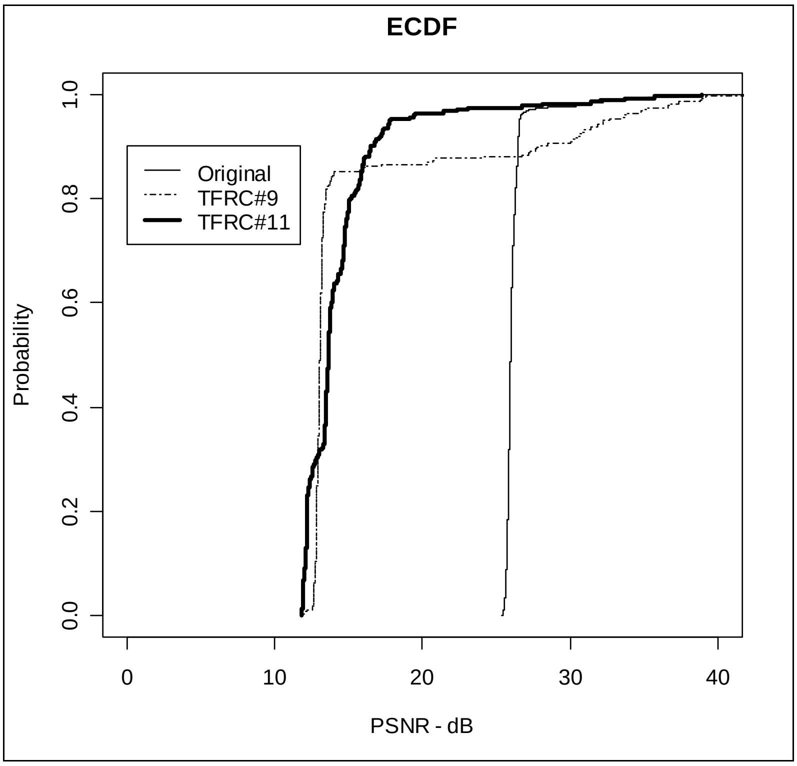

Figure 16 - ECDF - TFRC#9 and TFRC#11

Figure 17 presents the simulation results for TFRC#9 and TFRC#11 parameterization. For TFRC#9 we turned self-clocking ON, set the History Loss Size to 128, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 100ms. For TFRC#11, we adjust the self-similar background traffic to 75% of the bottleneck capacity. Figure 18 shows the ECDF for the same TFRC parameterization.

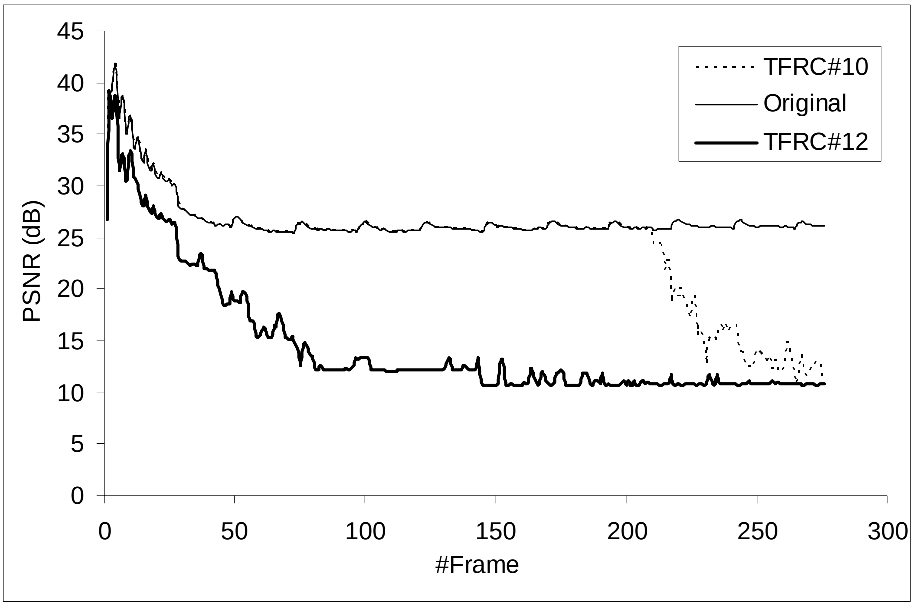

Figure 17 - TFRC#10 and TFRC#12

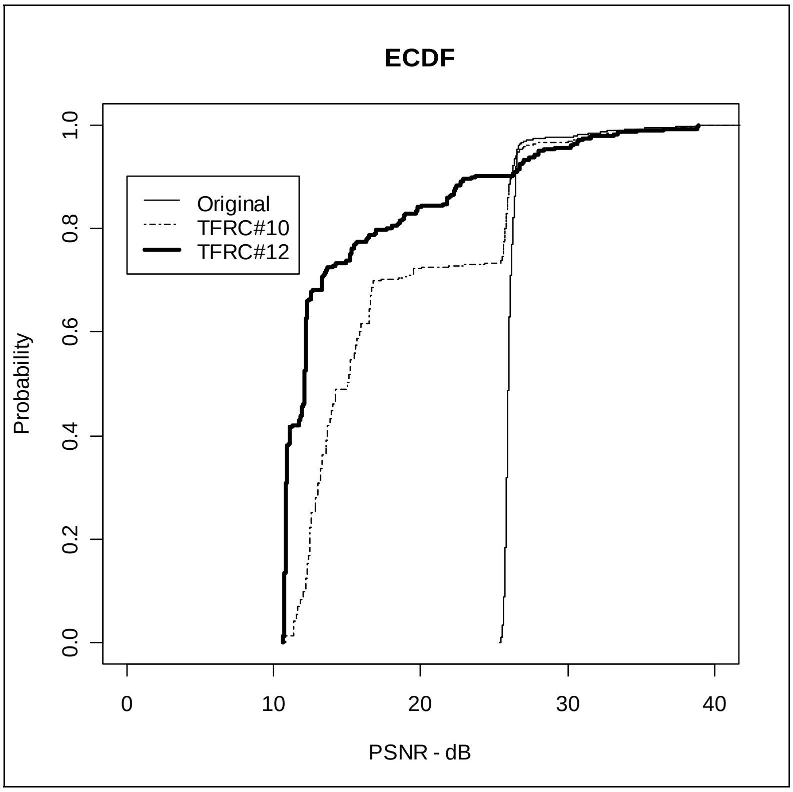

Figure 18 - ECDF - TFRC#10 and TFRC#12

Figure 19 presents the simulation results for TFRC#13 and TFRC#15 parameterization. For TFRC#13 we turned self-clocking OFF, set the History Loss Size to 128, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 100ms.

For TFRC#15, we adjusted the self-similar background traffic to 75% of the bottleneck capacity. Figure 20 shows the ECDF for the same TFRC parameterization.

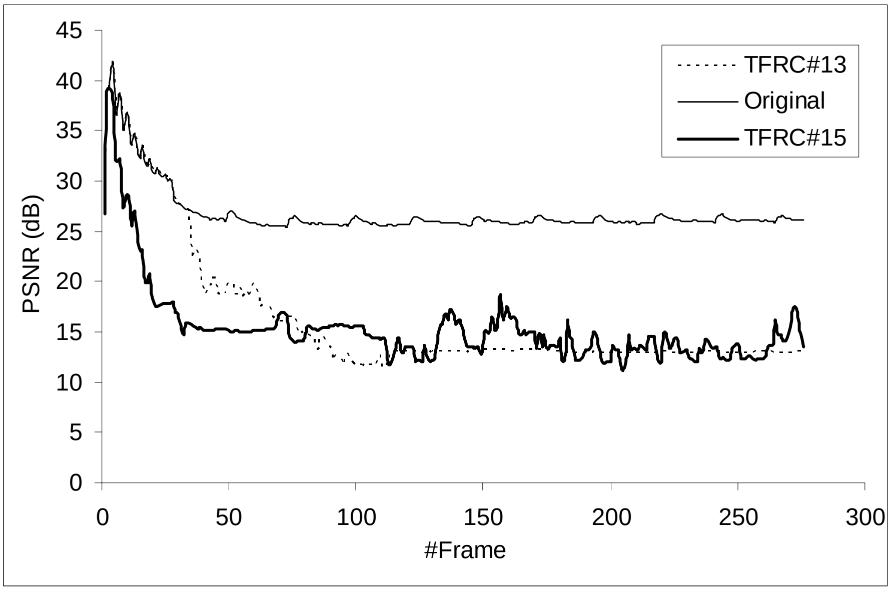

Figure 19 - TFRC#13 and TFRC#15

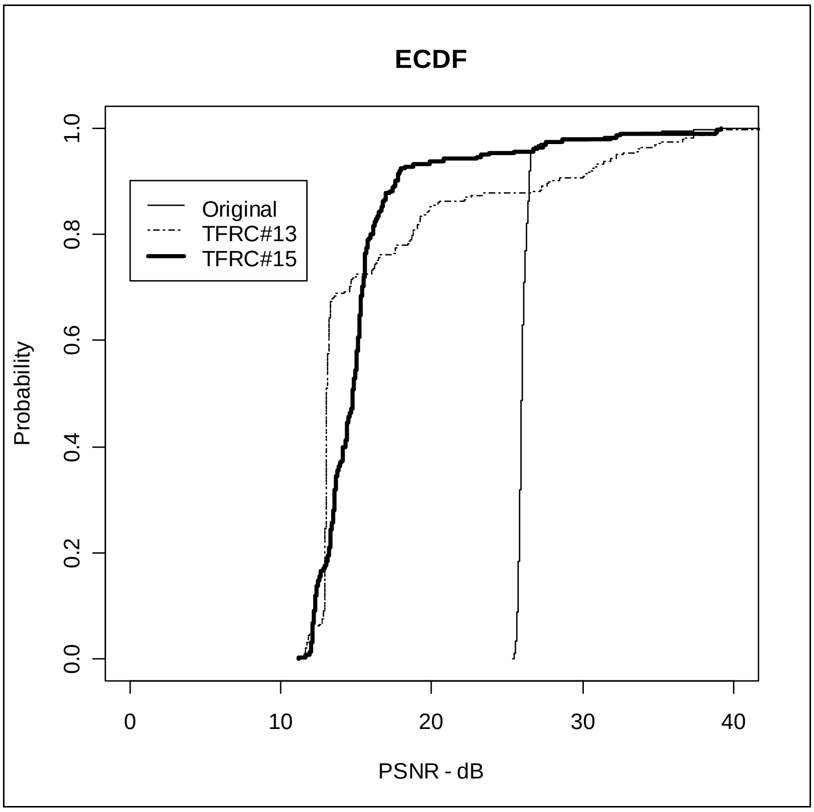

Figure 20 - ECDF - TFRC#13 and TFRC#15

Figure 21 presents the simulation results for TFRC#9 and TFRC#11 parameterization. For TFRC#9 we turned self-clocking OFF, set the History Loss Size to 128, defined the background traffic level to 25% of the bottleneck link, and set the bottleneck delay to 100ms.

For TFRC#11, we adjusted the self-similar background traffic to 75% of the bottleneck capacity. Figure 22 shows the ECDF for the same TFRC parameterization.

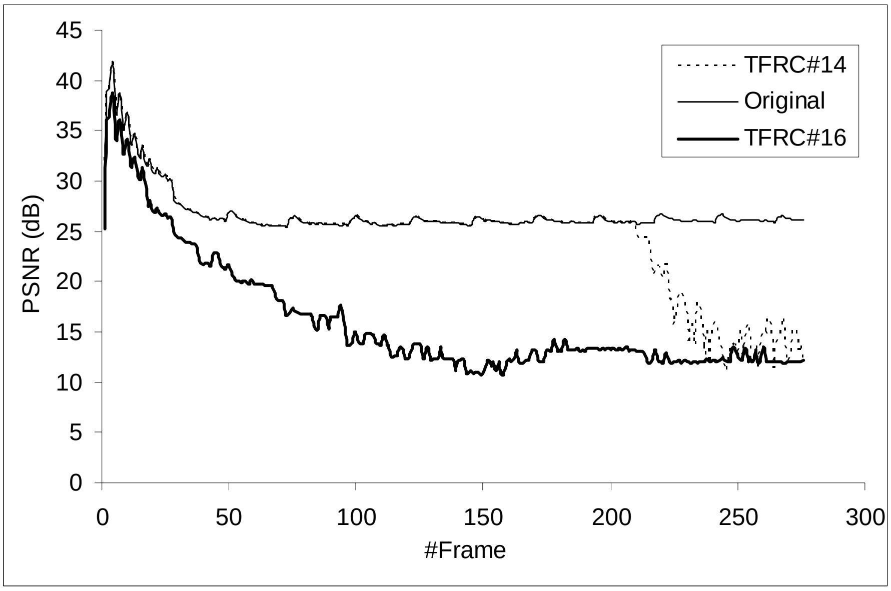

Figure 21 - TFRC#14 and TFRC#16

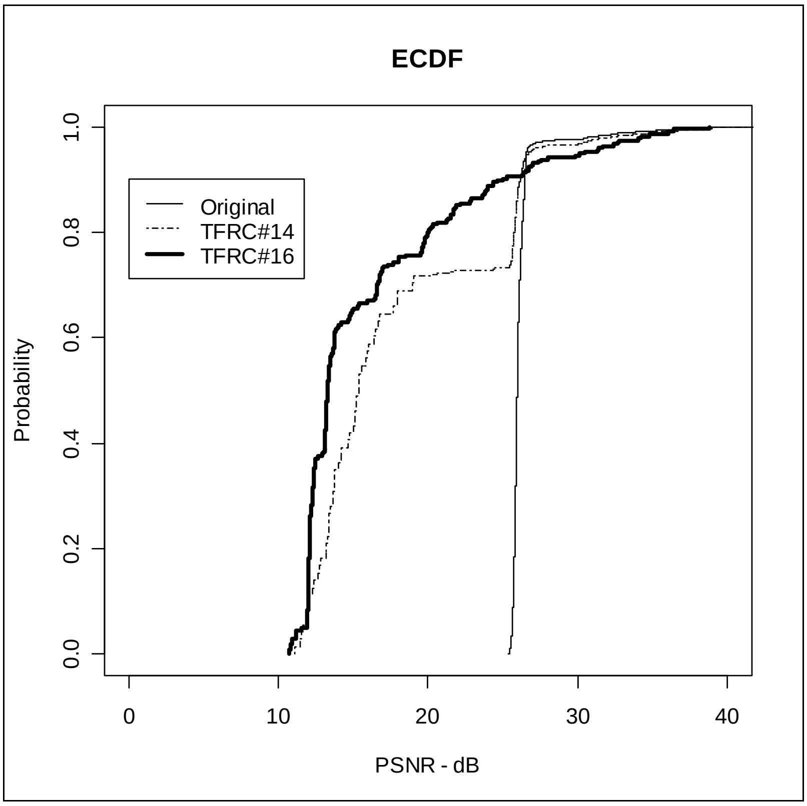

Figure 22 - ECDF - TFRC#14 and TFRC#16

We can observe from these results that TFRC's rate oscillations cause potential performance degradation for video streaming applications. We provide additional comments in Section 3.5.

3.4. Video Streaming with Explicit Feedback Notification

Next, we decided to select one of the protocols that rely on explicit feedback notification from the network, i.e. XCP or CADPC, to validate our hypothesis that we can substantially enhance the user perceived video quality. Since both XCP and CADPC have very similar performance behavior in several network scenarios, we advocate that analyzing only CADPC protocol will be sufficient at this point of the thesis. Later, we will extend the validation by evaluating XCP behavior in our proposal architecture.

For CADPC evaluation, we selected the *rttFactor* and *timeout_* parameters. As CADPC makes strong rate adaptation based on PTP packets, the *rttFactor* helps to prevent oscillations. By reducing its proposed default value of 4xRTT, it will make CADPC more reactive. The *timeout_* attribute determines how long CADPC should wait for a new PTP packet reply before sending another one [238]. We set the value for the *rttFactor* as 1.0, 4.0 or 8.0. We also set the *timeout_* value as 0.5 and 2.0. Table 1 lists all CADPC parameterization options along with the labels used to identify them. One should notice that we insert a new column at the CADPC label's table, to indicate whether this protocol could transmit video packets flawlessly. Therefore, in this Section we present only the results that had video frame losses, which means that the received video signal was different from the original one.

Table 2 - CADPC parameterization and labels

CADPC Label	*timeout_*	*rttFactor*	Bottleneck Delay	Background Traffic	Received Flawlessly
CADPC#1	2.0	1.0	10 ms	25%	YES
CADPC#2	2.0	1.0	100 ms	75%	YES
CADPC#3	2.0	1.0	10 ms	75%	NO
CADPC#4	2.0	1.0	10 ms	75%	YES
CADPC#5	2.0	4.0	10 ms	25%	YES
CADPC#6	2.0	4.0	100 ms	25%	YES
CADPC#7	2.0	4.0	10 ms	75%	NO
CADPC#8	2.0	4.0	100 ms	75%	NO

CADPC#9	2.0	8.0	10 ms	25%	YES
CADPC#10	2.0	8.0	100 ms	25%	YES
CADPC#11	2.0	8.0	10 ms	75%	NO
CADPC#12	2.0	8.0	100 ms	75%	YES
CADPC#13	0.5	1.0	10 ms	25%	YES
CADPC#14	0.5	1.0	100 ms	25%	YES
CADPC#15	0.5	1.0	10 ms	75%	YES
CADPC#16	0.5	1.0	100 ms	75%	NO
CADPC#17	0.5	4.0	10 ms	25%	YES
CADPC#18	0.5	4.0	100 ms	25%	YES
CADPC#19	0.5	4.0	10 ms	75%	NO
CADPC#20	0.5	4.0	100 ms	75%	NO
CADPC#21	0.5	8.0	10 ms	25%	YES
CADPC#22	0.5	8.0	100 ms	25%	YES
CADPC#23	0.5	8.0	100 ms	75%	NO
CADPC#24	0.5	8.0	10 ms	75%	NO

The following results (Figure 23 to Figure 28) present the average PSNR, for the first 300 out of 1000 frames and at least 100 simulation runs, received at the Video Client node for CADPC. In all simulations, as the same way as the TFRC simulations, we observed that network achieved its steady-state behavior before the 200th frame, but in most cases before the 100th frame. Therefore, in order to make graphics clearer, we also decided to show only the first 300 frames. Additionally, in order to get the big picture of the simulation results, we present the Empirical Cumulative Distribution Function (ECDF) for each parameterization. It is worth stressing that we also decided not to show the confidence intervals, since there is no superposition between the original PSNR signal and those from the CADPC simulations. As we argued before, this choice facilitates an aesthetic and clean presentation of results.

In all simulations, one can clearly see that there is a notable improvement in the PSNR level, since transporting streaming media over CADPC could help reducing its oscillations. As it is easy to comment such results, we first present all simulation results and discuss some of them later in this Section.

Figure 23 presents the simulation results for CADPC#7 and CADPC#8 parameterization. For CADPC#7 we turned *timeout_* to 2.0, set the *rttFactor* to 4.0, defined the background traffic level to 75% of the bottleneck link, and set the bottleneck delay to 10ms. For CADPC#8, we only set the bottleneck delay to 100ms. Figure 24 shows the ECDF for the same CADPC parameterization.

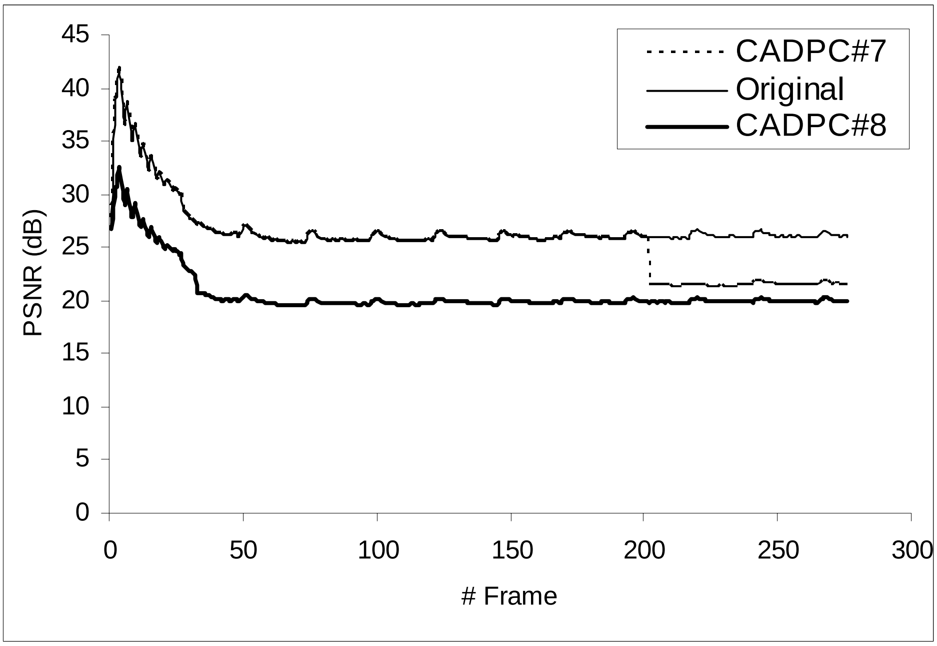

Figure 23 - CADPC#7 and CADPC#8

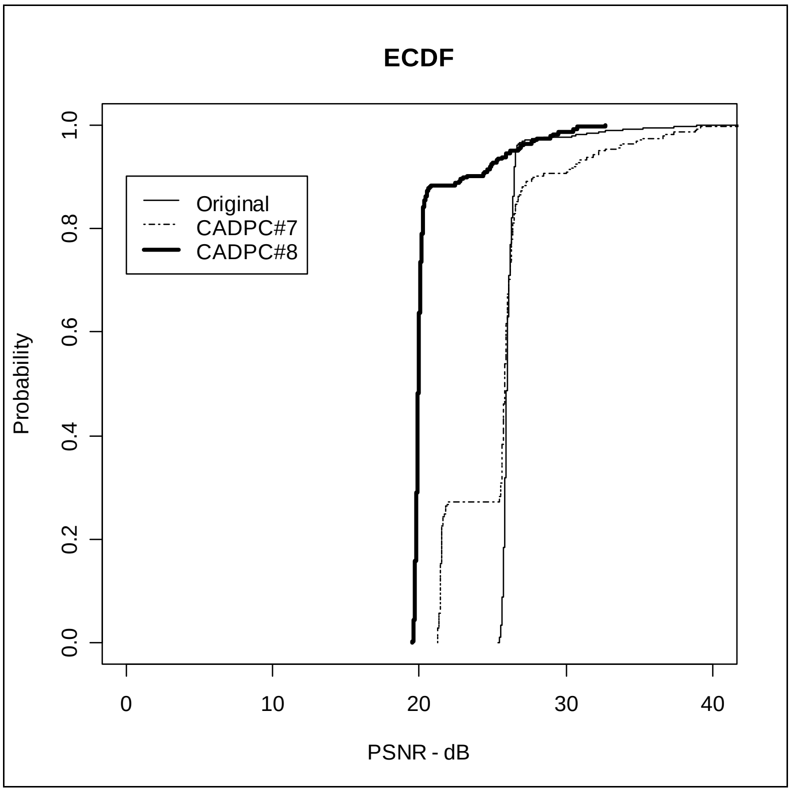

Figure 24 - ECDF - CADPC#7 and CADPC#8

Figure 25 presents the simulation results for CADPC#19 and CADPC#20 parameterization. For CADPC#19 we turned *timeout_* to 0.5, set the *rttFactor* to 4.0, defined the background traffic level to 75% of the bottleneck link, and set the bottleneck delay to 10ms. For CADPC#20, we simply set the bottleneck delay to 100ms. Figure 26 shows the ECDF for the same CADPC parameterization. In such a situation, CADPC becomes less reactive due to the *rttFactor*, although it floods the network with extra PTP packets (small *timeout_* value).

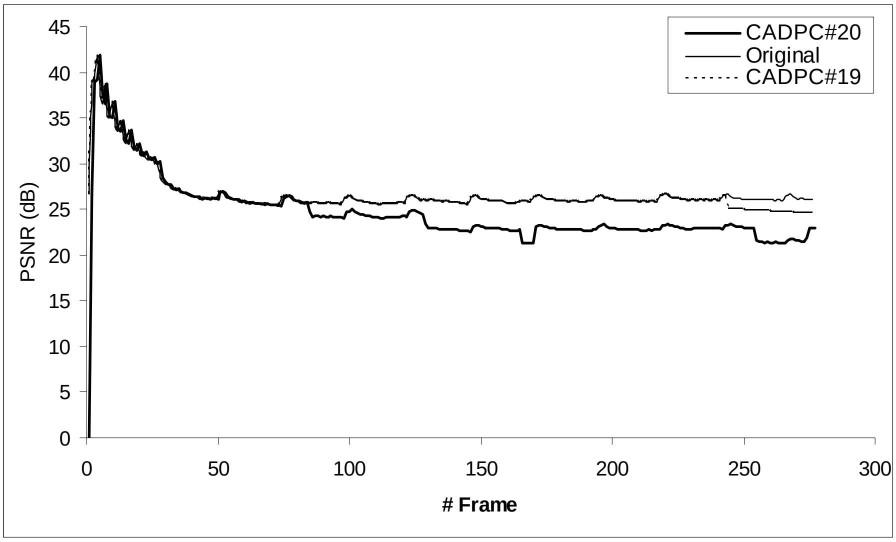

Figure 25 - CADPC#19 and CADPC#20

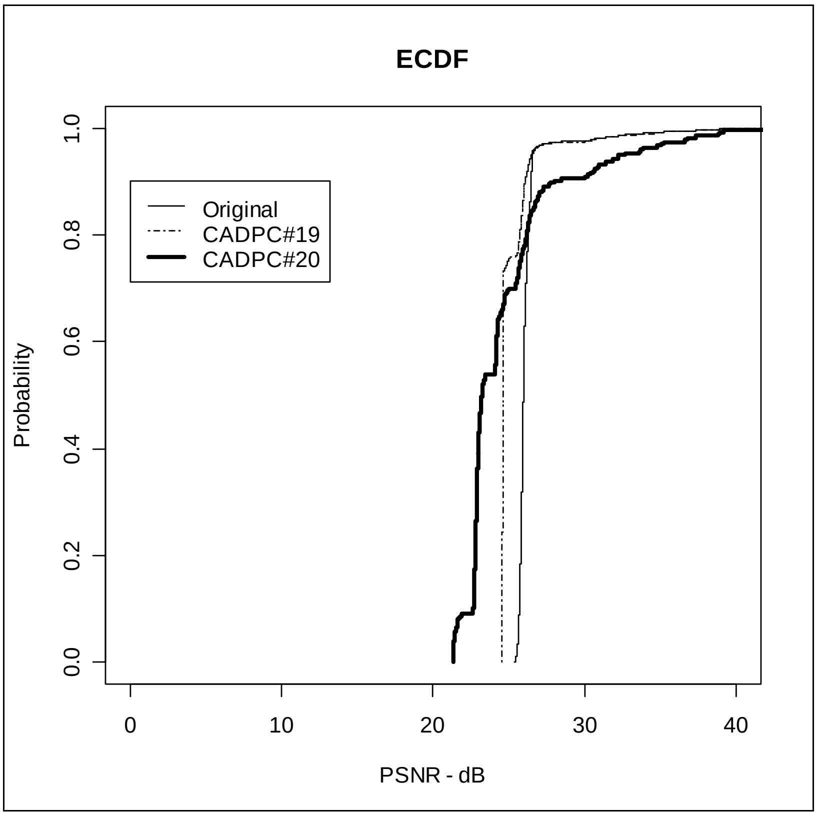

Figure 26 - ECDF - CADPC#19 and CADPC#20

Figure 27 presents the simulation results for CADPC#23 and CADPC#24 parameterization. For CADPC#23 we turned *timeout_* to 0.5, set the *rttFactor* to 8.0, defined the background traffic level to 75% of the bottleneck link, and set the bottleneck delay to 10ms. For CADPC#24, we simply set the bottleneck delay to 100ms. Figure 28 shows the ECDF for the same CADPC parameterization.

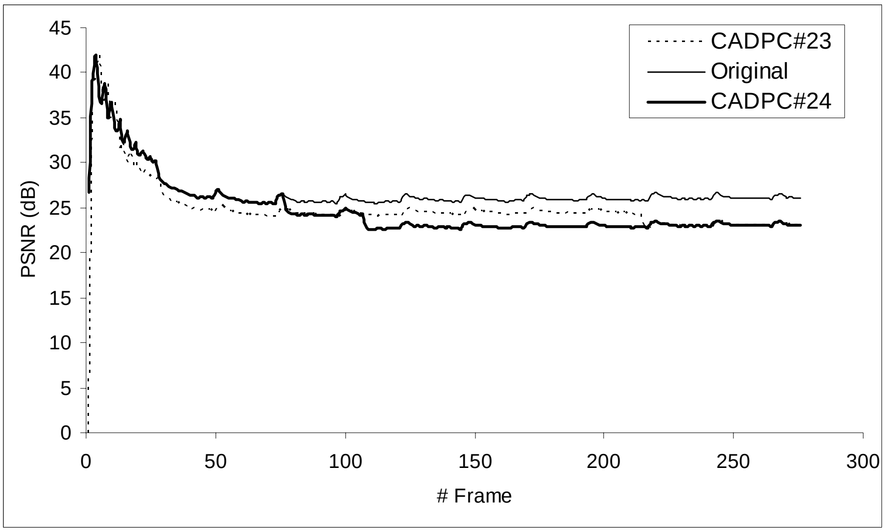

Figure 27 - CADPC#23 and CADPC#24

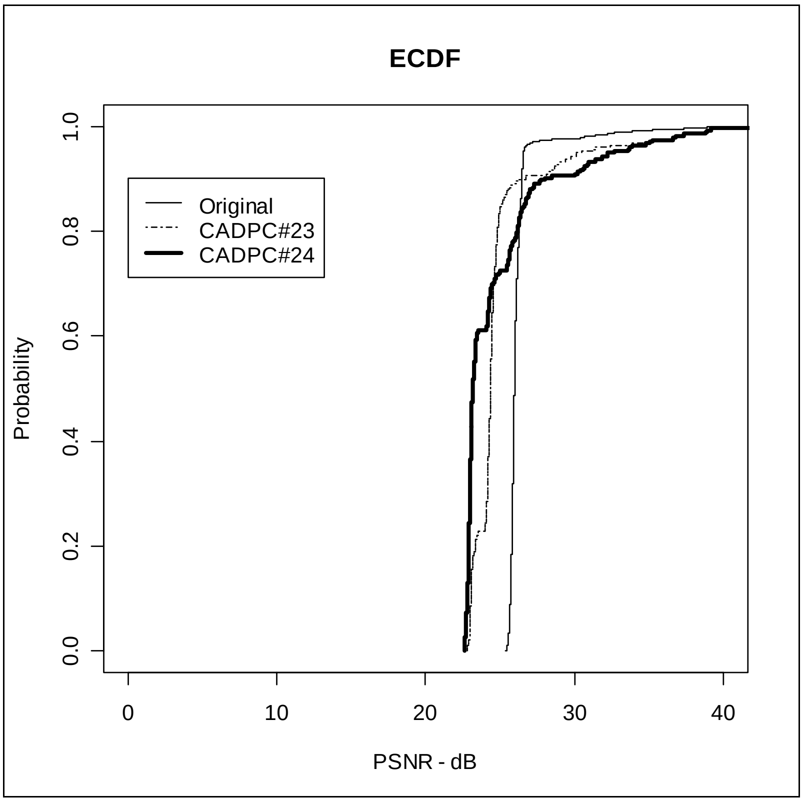

Figure 28 - ECDF - CADPC#23 and CADPC#24

In general, we can observe that as CADPC tries to get more information from the network (e.g., with a small timeout), it provides a better performance for the video streaming. However, the combination of the two main factors, i.e. timeout and RTT, is crucial to the overall performance. Besides, CADPC attempts to smooth the available bandwidth information using other attributes of the CAPC sender agent. Such attributes include *smoothness*, *Inter Packet Gap (IPG)*. One must observe that the author alerts that *Smoothness* parameter is sensitive and values greater than 0.5 increases aggressiveness. On the other hand, IPG controls the rate by varying the time between packets. Therefore, although CADPC attempts to reduce rate oscillations as much as possible, there is no accurate or adaptive parameterization. Consequently, CADPC fails to provide a reliable and steady behavior for all network environments. These facts lead us to seek ways to enhance the perceived quality with no need for proposing new transport protocols.

3.5. General Remarks

In this chapter, we performed a quantitative evaluation of the end-user perceived media quality of video streaming under network friendly protocols in the best-effort Internet. First, we evaluated the effect of throughput variation of the TFRC Protocol on the PSNR of MPEG-4 video trace files. Some recent studies ([16] [234] [246]) have shown that TFRC protocol is more stable than other slowly responsive congestion control protocols, but it is highly dependent on the packet loss patterns. They also found out that TFRC fails to prevent abrupt sending rate reduction during transient workload increases. Additionally, they found that increasing the history size parameter (i.e., for loss event computation) or removing self-clocking could only give it slight resistance to transient changes.

Recently, the Internet Architecture Board (IAB) presented some concerns regarding congestion issues for continuous media traffic in the Internet [190]. In such documents, the authors raise the discussion about congestion control issues for voice traffic. The Internet Draft (I-D) proposed by Phelan [168] follows the guidelines in the RFC 3714 [190], but it has a broader scope. Phelan discusses strategies for using streaming media applications with unreliable congestion-controlled transport protocols for TCP Friendly Rate Control. He also focuses the discussion on how media streaming applications can adapt to the varying transmit rate requirements of congestion control protocols. He suggests that some streaming media applications operate in a non-adaptive fashion, thus never changing its mode of operation. In [12], we analyzed the dynamic behavior of popular VoIP P2P applications when submitted to a variety of network conditions. In such research, we found out that in some scenarios, such applications do not change their mode of operation, i.e. the codec or its encoding parameters.

As far as this discussion concerns to congestion control protocols, there are several characteristics in most current congestion control mechanisms that do not match with common media stream transmission practices. We list some particular considerations presented by Phelan

[168]. Although his discussion concerns to only TFRC, we extend his arguments to other congestion control (CC) mechanisms present in transport protocols. First, the slow start phase implies that the initial transmit rate is often slower than the lowest bit rate encoding of the media, forcing the application to deal with a ramp up period. Second, as it is usual for AIMD-based procedures, CC mechanisms will raise the allowed rate until a packet is lost. Phelan considers that, in many circumstances, packet loss will not be a rare event, thus occurring routinely in the course of probing for more capacity. Finally, current CC mechanisms are essentially greed. If an application asks for transmitting at the maximum allowed rate, they will try to raise that rate. However, if its sending rate requirement is below the maximum allowed rate, the CC mechanism will not increase the maximum allowed rate promptly. In the case of TFRC, the maximum allowed rate will not be increased higher than twice the current transmit rate. Such inherent behavior create bottlenecks for the application server when it attempts to use a higher rate encoding, or simply raise the transmission rate due to a scene variation. As we argued earlier in this thesis, continuous relocation between several quality levels will annoy users. It is a common sense that it is better for the perceptually quality that once the server find a suitable quality level, it should remain there for sometime.

Please note that our approach deal with pre-stored video that is mainly suitable for video on demand services. Coping with live streaming media requires different approaches that we will discuss later in this thesis (Chapter 6). Fundamentally, recorded and live media differ from each other in their greed for bandwidth. While server can send recorded media as fast as the network allows, the encoder limits the sending rate for live media, i.e. the maximum encoding rate. Live media suffers with motion compensation side effect. Motion compensation techniques generate huge variation in the sending rate from the minimum to the maximum rate. With a fine-grained approach for multi-layered pre-stored video, one can control the requirements for bandwidth efficiently. Phelan [168] presents different strategies for streaming media including the cases for one-way pre-recorded media, one-way live media, and two-way interactive media. We now comment the first case, which is also the focus here, and compare with our scope and approach. His approach has as a first assumption that a pre-recorded video file resides on a media server, and the server and its clients are capable of stream switching between only two encoding rates. The receiver playout buffer is sufficient to hold the entire recording. He divides the playout buffer in three thresholds, namely the low-level, medium-level, and high-level. He also assumes that during the connection the server is able to determine the depth of data in the receiver playout buffer. Thereafter, his approach for switching from the high to low rate (and vice-versa) relies on the evaluation of such buffer level. Consequently, the sending rate at the server will oscillate between the two levels until the end of the transmission. Additionally, if the network available bandwidth becomes below the low-bit encoding rate for a long period, the playout buffer will drain completely. Phelan argues that in this scheme, the media server does not need to know the rate that TFRC has determined explicitly, since it can send as fast as it allows. TFRC will shape the stream to the network's bottleneck. Moreover, the playout buffer feedback will allow the server to shape the stream to the application's requirements.

Although there are some interesting ideas in Phelan's Internet Draft (I-D), it has weaknesses that deserve an in-depth discussion in order to overcome them. First, the assumption of a large buffer space at the client does no hold, even for standard personal computers with hundreds of megabytes of memory. Second, with the adoption for scalable video, such as MPEG-4 FGS, one has several quality levels that it could switch back and forth to, thus implying the need for a precise server-side adaptation policy. We consider this when proposing our novel architecture (Chapters 4 and 5). This situation will eventually worse since TFRC probes the network to determine its capacity. As it does not know a priori what is the available bandwidth, it increases the transmit rate until packets are lost, then backs down. In view of the fact that TFRC follows an AIMD policy, the transmit rate will oscillate up and down, with packet loss events always occurring at the rate peaks. In his I-D proposal, Phelan is aware of that packet loss will routinely appear and the extent of noticeable quality problems will depend on the characteristics of the codec in use. In this thesis, we assume that the playback buffer is just sufficient to provide a small relief by absorbing delay and bandwidth variations from the network (Chapter 4). We do believe that the intrinsic AIMD policy in TFRC, lead to the observed perceptually low quality under highly dynamic network scenarios.

With all these arguments in hand, we advocate that any adaptive rate control for multimedia streaming flows should rely on explicit feedback notification from the network, in order to provide the streaming media with both uninterrupted transport services and low quality variation. As the Internet research community increases its interests in the adoption of some AQM-based mechanisms, we find that some recent strategies, such as CADPC and XCP, could be able to supply video streaming with rate-based, oscillation-free virtual channels [251]. Hence, we carried out a set of experiments under the CADPC framework. Simulation results have shown that application-level mechanisms should rely on protocols with explicit feedback notification as they offer smoother rate variation than TFRC, since the received quality seems to be more stable (based on PSNR measurements).

We have some general remarks about TFRC and CADPC performance. In summary, CADPC has overall better performance than TFRC. In the case of background traffic with a mean level of 25% of the bottleneck capacity, CADPC performs flawlessly whereas TFRC shows losses in some scenarios. Additionally, CADPC performs well even when the background traffic reaches 75% of the bottleneck capacity, thus achieving around 1-2dB in losses in the PSNR level. On the other hand, TFRC failed to provide consistent mean quality at the receiver, since it has strong oscillations in the short-term throughput estimation, thus leading to high variations as well as low mean quality level. However, we observe that even with explicit notification, the perceived video quality can still oscillate using single layer videos. It really makes difficult to predict what will be the performance behavior when using scalable temporal and spatial video over explicit rate feedback networks. This observation leads us to go further in this evaluation. We advocate that it is possible to mitigate the effects of short- and long-term variations in video quality by controlling the short-term variations in the available bandwidth or the resource demands in the video server using buffer strategies, and the long-term variation relying on techniques to control portions of video layers streamed into the network.

By testing the effect of the above-mentioned algorithms on the perceived visual quality at the end-system's application-level, we had a clear picture that a proper approach to overcome this problem is to switch the solution to another level. The idea would be a mechanism or framework that takes into account the volatility in the available rate in order to decide when to transmit portion of the enhancement layers (e.g., using scalable encoded video with MPEG-4 FGS). However, before taking any decision such adaptation mechanism should smooth the information received from the transport layer. We think this hypothesis as viable, since investing in a novel protocol or even attuning the existing ones, can cause side effects such as network inefficiency and unfairness with competing flows.

Chapter 4. An Architecture for Streaming Scalable Encoded Video

As alluded to earlier in this thesis, providing perceptually good quality streaming video is a hard task. This is due to the fact that in today's best-effort Internet (e.g., in access networks) the available bandwidth can fluctuate strongly. It is well known that its traffic profile exhibits variability at multiple time-scales [188] [213]. For small time-scale bandwidth fluctuations, i.e. on the order of a few milliseconds, a small play-back buffer at the client side can provide limited relief. For longer time-scale bandwidth fluctuations, i.e. on the order of a few seconds, either the use of multiple versions of the same video or layered-encoded video is a feasible solution. On the other hand, encoded video can also exhibit significant rate variability at several time-scales. Our main argument is that even if the end-system knows the available bandwidth information precisely, server application will need adaptive control policies. Such control policies will decide which additional portion should be streamed, e.g. whole layers in a multi-layer approach or fraction in a Fine-Grained Scalable approach [44]. Additionally, one important requirement for transporting multimedia flows is that such streams must exhibit fairness with competing flows. Therefore, the main issue is to accommodate the mismatch caused by available bandwidth variability and the encoded video rate variability, keeping in mind the following goals:

a) Minimization of the quality variability and

b) Maximization of the overall quality of the video rendered to the user.

Let us take a close look at such goals. When a server application tries to minimize the quality variability, its main task is to add or drop video layers (or fraction of layers). However, frequent layers adding and dropping (or fractions of layers) is annoying and certainly degrades the perceptual quality of video. On the other hand, quality maximization means a received signal as close as possible to the original. The underlying transport protocol performance imposes an upper bound limit on the latter goal, since it has to ensure max-min fairness (e.g. TCP-Friendliness) and network efficiency. It is worth emphasizing some arguments for obtaining smooth video quality. As described in [194], from the human visual system point of view, smoothed video quality is visually better for the human perception than variations in quality, such as flicker, frozen images, and temporal noise, since they are very annoying in video appearance.

In this chapter, we take a further step on the analysis and solution of this problem. In Section 4.1, we present some related work, followed by the definitions and notation used throughout this chapter in Section 4.2. We describe in Section 4.3, some important design decisions for the proposal of our architecture. After these considerations, we give details about the architectural components in Section 4.4, where we describe the Dynamic Low-Pass Filter unit, the Prediction unit, and the Decision unit. Finally, to conclude this chapter, we carry out an extensive simulation-based performance analysis in Section 4.5.

4.1. Related Work

In the past, researchers addressed adaptation, smoothing and prediction techniques as promising approaches to provide better quality or utilization of network resources [52] [74] [118] [130]. In [192], Salehi et al. propose a smoothing technique by work ahead for non-scalable VBR video. Using this technique a video server sends video data ahead of schedule in order to minimize the variability of the transmitted bit rate. As an optimal solution, they compute the transmission schedule which minimizes both the peak rate and variance of the rate at which data is sent to the client. They also evaluate the impact on the network resources required by the video stream, under a realistic network service models, namely the Renegotiated Constant Bit Rate (RCBR). In general, their findings indicate that optimal smoothing can result in a significant reduction in the network resources required for VBR video.

In [74] Grossglauser et al propose a dynamic bandwidth allocation mechanism to support VBR encoded video that uses an adaptive linear prediction in order to predict the bandwidth requirements for upcoming frames. They argue that supporting real-time VBR video traffic using CBR allocation does not achieve good network utilization. Alternatively, using prediction and allocating bandwidth dynamically provides higher utilization and small buffer size requirements. In a few words, Grossglauser et al investigate the performance of linear prediction algorithms to forecast VBR video traffic using adaptive techniques. Furthermore, they study the performance of dynamic bandwidth allocation based on predicted values under RCBR network service model. However, Gan et al. [66] pointed out that renegotiation failure in RCBR might cause buffer underflow and interrupt the playback of video. In order to overcome this issue, they proposed a novel dual-plan bandwidth smoothing (DBS), which takes advantage of the SNR scalability of layer-encoded video. Upon a renegotiation failure event, the proposed scheme adaptively discards selected enhancement layers to maintain the original frame rate.

Unsurprisingly, the authors point three factors where DBS gets better performance that is reducing the renegotiation interval, employing multilayer FGS video encoding, and increasing the playback buffer size. We should interpret the reduction in the renegotiation interval differently on best-effort networks with explicit feedback notification. In our case, the network provides flow's allowed rate information until next RTT whereas in the RCBR case the network guarantees the negotiated rate until the next renegotiation. It is clear when the renegotiation interval gets close to the flow's RTT we have a fair approximation of both solutions. Furthermore, using fine-grained layered encoded video, allow both approaches to rely on scalable coding procedure for precise steaming according to network conditions. It is worth stressing that the DBS proposal is suitable in our architecture completely, since we did not include any transmission plan that roughly will establish preferable transmission rates during the video session. In general, such transmission plan will help avoiding both overflow and underflow of the playback buffer. Some research studies [73] [74] showed that the combination of transmission plans obtained through bandwidth smoothing techniques along with RCBR yield better network utilization through statistical multiplexing. Recall that in the RCBR approach a source renegotiate bandwidth with the network according to its desired transmission rate. In the DBS scheme, a renegotiation failure occurrence triggers a dropping event in the enhancement layers in order to match to both bandwidth and buffer constraints. Given the frame sizes (for both base and enhancement layer in the FGS coding) and the playback buffer size, the proposed DBS scheme compute in advance several transmission plans, namely the trunk plan and the some branch plans. The former establishes the main transmission schedule of the video stream, while the former determine the transmission rates for the lower layers, which servers should follow in case of renegotiation failure. We could interpret such renegotiation failure as a change in the available bandwidth conveyed by the underlying transport protocol (e.g., either XCP or CADPC). In other words, we could fully utilize the DBS proposal within our architecture. Although the authors consider the fine-grained dropping event as an advantage, since it degrades the picture quality, while maintaining the original frame rate without entire frame dropping, we keep arguing that one should also control recurrent quality variation.

Based on the same motivation as ours, Kim and Ammar [118] propose an alternative solution to the problem of accommodating the mismatch between the available bandwidth variability and the encoded video variability. Their focus is on quality adaptation algorithms for scalable encoded variable bit-rate video over the Internet. To this end, they developed a quality adaptation mechanism that maximizes perceptual video quality by minimizing quality variation, while at the same time increasing the usage of available bandwidth. It is worth discussing some drawbacks in Kim's proposal. First, they consider maximization of network utilization as an important feature in its architecture. We argue that maximization of network utilization must be an issue only for transport protocols. As we stated before, if in a design of a novel solution for video streaming there is an extreme cautious about network utilization, it should also worry about fairness. Second, for the optimal algorithm, its efficiency depends on the sender knowledge of the receiver buffer occupation, which could be clearly non-feasible and non-scalable. Third, the real time algorithm depends on the efficiency of the bandwidth estimator, which is not very precise under highly dynamic networks. In fact, coping with bursty traffic is a daunting task. Although XCP and CADPC provide applications with excellent control over the available bandwidth, they still must deal with intrinsic burstiness and high variability in the network traffic profiles. For wireless networks, Atkin and Birman [11] argue that transport protocols should offer more control over communication in order to allow applications to adapt their behavior to bandwidth variability. Furthermore, they argued that an application designed to operate in a wireless network might adjust its sending rate in order to respond to changing bandwidth available to the host. An informal definition of high variability, according to Willinger et al [243], is a phenomenon by which a set of observations takes values that vary over orders of magnitude. For instance, this phenomenon happens when the outcomes of an event take mostly small values, with a few observations arriving at very large values with non-negligible probabilities, and the intermediate-sized observations occurring with significant frequencies. In such a case, the sample standard deviation is in general gigantic, implying that the sample mean fails describe the location of the bulk of the observed values [243]. This observable fact implies the existence of concentrated periods of high activity and low activity at a wide range of time scales [83]. The fundamental problem addressed in the Atkin and Birman research study [11] is somewhat similar to ours. Their main argument is that most existing interfaces to network protocols do not provide much detail about network conditions. In most cases, determining the exact bandwidth available to the application is a cruel mission. At the core, the adaptation process relies on a Network Aware Interface (NAI) and on the Adaptive Transport Protocol (ATP). In order to adjust its behavior based on network conditions, ATP incorporates a bandwidth estimator where an application derives an estimate of how much bandwidth is available. The bandwidth estimator uses an averaging filter with a window size of five to smooth the estimates and to make them less sensitive to transient spikes. In summary, the proposed network-aware API for adaptive applications informs applications about current network state, allowing them to adjust their behavior accordingly. The ATP implementation of NAI indeed adjusts well to changes in bandwidth, indicating that an application using ATP will eventually match its sending rate requirements to the available bandwidth. Therefore, in a search for flexible transport protocols suitable for fine-granular scalable encoded video, proposal such as NAI/ATP give us insight into why and how we should build our solution for our problem scope. To some extent, we agree that under the circumstances of high burstiness level in any network feature (e.g., RTT) applications will certainly face poor performance, unless they are able to adapt to such a degree of variability. For the available bandwidth case, such scenario induces us to avoid both underestimation and overestimation, since it can lead to either underutilization of network resources or application malfunctioning, respectively. However, we deal with the specific case of scalable video over wired networks whereas they cope with general adaptive applications over wireless environments.

Recall that De Cuetos and Ross' work [44] presented a similar approach that investigates a solution for adaptive streaming to adapt to the short- and long-term variations in available bandwidth over a TCP-friendly connection. We discussed that work in Chapter 2. The framework applies to stored fine-grained scalable video and its main contribution is the proposal of an optimization formulation to solve an optimal streaming problem. We consider that by simplifying the original problem, in order to make the problem more tractable, the solutions lead to a number of pitfalls. First, they considered both BL and EL CBR-encoded. Moreover, the real-time algorithm discards the assumption that the sender knows the available bandwidth a priori. However, it has two major drawbacks. It continues to depend on the knowledge of the client buffer level. Additionally, its efficiency depends on the fixed parameterization of the moving average algorithm for the sending rate, which in turn is not at the same time scale of the available bandwidth estimation. Departing from a slightly different perspective, adaptive delivery mechanisms indicate how to adjust the transmission rate of the real-time encoder in response to various network conditions. For example, Cranley et al. [40] propose the use of an optimum adaptation trajectory, which indicates how encoding quality should be adapted with respect to user perceived quality, thus maximizing it. By finding a set of encoding parameters close to optimal, a live video streaming server can use the knowledge of user-perceived quality in cooperation with any existing adaptation mechanism. Therefore, we do believe that we can improve our architecture by placing both bandwidth smoothing and adaptive delivery mechanisms as functional blocks for streaming live-encoded video. However, this evaluation is out of scope of this thesis, since it is possible to validate our proposal with the use of pre-stored encoded video only.

In a similar approach to Lakshman [130], Duffield et al [52] also propose an adaptive smoothing algorithm for compressed video, namely SAVE (Smoothed Adaptive Video over

Explicit rate networks). They show that SAVE maintains the quality of the video within acceptable levels, ensures that the delay is within acceptable bounds, and that there are significant multiplexing gains. Similarly to our approach, it uses the explicit rate based control mechanisms to transport compressed video. The main difference is that SAVE is in the context of ATM networks and it adapts the encoder by modifying the quantization parameter, whereas ours uses stored and scalable video and does not work with adaptation into the encoder. Two important Duffield's paper remarks give us support and confidence to build our architecture. First, the paper states that the combination of the explicit rate mechanism and the smoothing technique lead SAVE to achieve a higher multiplexing gain. Moreover, smoothing also maintains a suitable requested rate of the network, which in our case is viable for matching the average available bandwidth.

4.2. Definitions and Notation

We now present the formal notation that we will use in this chapter. First, we define adaptation or quality adaptation, which is a mechanism that either adds or subtracts information on video frames (in layers or fraction of layers) based on the available network bandwidth. In this thesis we work with pre-stored video, therefore in this context adaptation does not mean any form of dealing with the encoder by modifying the quantization level during compression whatsoever. The stored video is encoded into two layers, namely a Base Layer (BL) and a Fine-grained Enhancement Layer (EL). Both BL and EL follow the MPEG-4 Fine Grain Scalable (FGS) specification [138] [176]. The combination of the BL and the EL has a VBR profile. We denote the BL encoded rate as $X_{BL}(t)$ and the EL encoded rate as $X_{EL}(t)$. We also denote the encoded rate of the combination as $X(t) = X_{BL}(t) + X_{EL}(t)$. The video length is T (seconds).

Thereafter, we make some key assumptions. First, the available bandwidth exhibits multiple time-scale variability. This is due to the highly dynamic background traffic. The end-to-end network path is reliable and provides each flow with its fair share by using transport protocols with explicit feedback notification (e.g., either XCP or CADPC). Second, the bottleneck has enough capacity for transporting the BL flawlessly and losses may only occur due to missed play out deadlines. Buffering capacity at the receiver allows occasional selective retransmissions, but most important it allows it to neglect small transmission delays in the network path. Additionally, such buffer can absorb short time-scale available bandwidth variability. As a requirement for most portable devices, receiver buffer sizes should be kept small. Finally, a server always sends the BL and a portion of the EL. Figure 29 shows how the server merges both BL and EL of the stored video feeding the aggregation into the network.

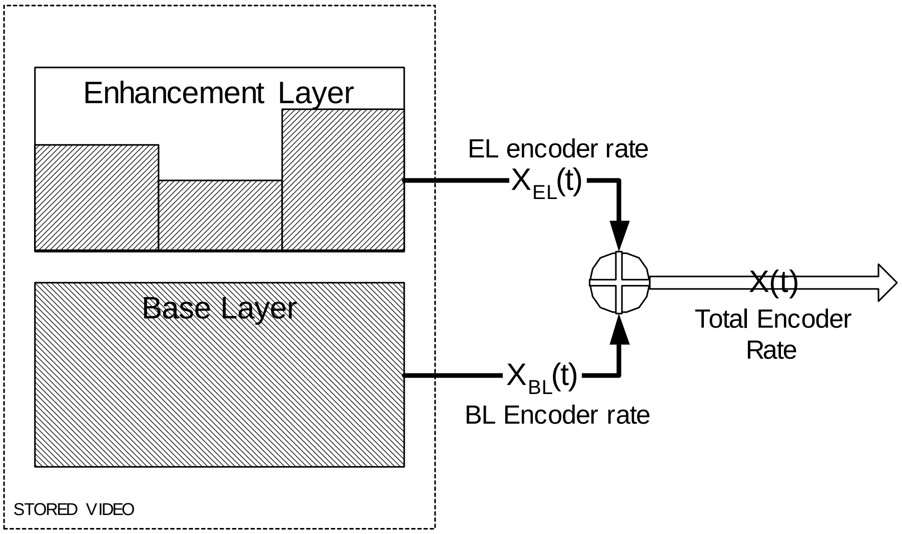

Figure 29 - Combination of BL and EL

As we stated earlier in this thesis, matching VBR video encoded streaming from the video server to VBR available bandwidth information from the network does seem to be an infeasible task. It is clear that at least one element of this system should be smoothed in order to reduce the complexity of the solution. However, instead of simplifying the problem as in many previous researches [52] [118], we propose a solution to smooth the information provided by the transport protocols in order to achieve as close as possible a CBR-like perception level at the user, thus reducing complexity.

4.3. Design Rationale

Our initial objective is extracting from the lower levels (i.e. transport and network levels) the most precise information as possible, and provide upper level (i.e. application) with a correspondent reliable and stable one. Such information flow forms the foundation to build an architecture, which in turn should be flexible to extend, subtract, or change functionalities. Hence, the architecture emphasizes the most important objective that is the minimization of quality variability.

Another crucial design decision concerns the time scale in which the system components will deal with. There is a clear advantage for decoupling the time scale of interest of the encoder and transport. Due to its intrinsic characteristics, an encoder should not switch to different EL levels at the same time scale that transport protocols work. In other words, an application server should only change its quality level when it is safe to do so. By safe, we mean that such change will have a positive impact on the perceived user quality, presumably. With this desirable feature in mind, we argue that one architecture component should accumulate data from the transport levels, make some adjustments before passing it to the application server, thus decoupling the time scale domain for different components. Specifically, in our proposal the application servers work at the Group of Pictures (GOP) time scale domain (i.e. on the order of a few seconds), whereas transport protocols work at the RTT domain (i.e. on the order of hundreds of milliseconds).

As far as our architecture mainly concerns for adaptability of the application server, we focus on how to manage state transitions (adaptability for the EL), in order to provide minimal quality variation. First, we rely on the evaluation of the smoothed available rate, which is the result from a low-pass filter. We develop this idea more in the next Sections. Other possible auxiliary sources of information for the adaptation heuristics are the stochastic volatility and prediction error of the available bandwidth. A low value of these metrics means network stability and that it is apparently safe to increase quality using additional bits from the EL. On the contrary, higher values means instability and the application server should stay on the same level or decrease quality to ensure low variability.

It is worth emphasizing that our architecture relies on the use of a transport level with explicit network feedback information. In chapter 3, we showed that relying on such protocols would provide precise and reliable available bandwidth information, ensure fairness with competing flows, and maximize network efficiency.

Figure 30 presents an overview of our novel architecture for video streaming over best effort networks. We observe several components that highlight the contributions of this thesis. We will give details about all units' functionalities in the next Sections in this chapter. However, one should observe that our proposal architecture is flexible to support live or pre-encoded stored video. For performance evaluation, we will use only stored video, since we are not dealing with either encoding or quantization issues. The streaming server unit is responsible for the aggregation of the BL and EL. In other words, it makes its decision of how much of the EL should be added to the BL based on information that comes from the adaptive-predictive (AP) unit. The AP unit in turn receives information about network status from the transport protocol.

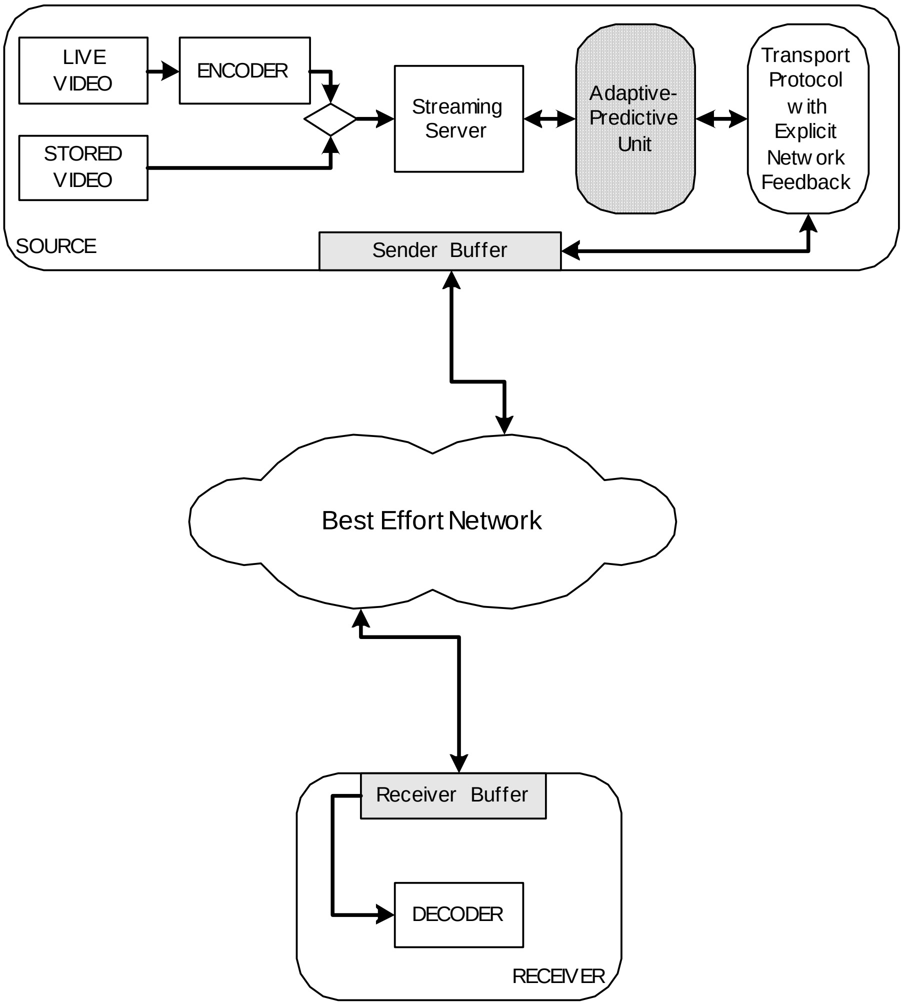

Figure 30 - Adaptive-Predictive Architecture for Video Streaming

4.4. System Components

We now scrutinize all the system components. Figure 31 shows the AP unit in details. Three major components form the AP unit, namely the Dynamic Low-Pass Filter (DLPF), the Prediction Unit (PU) and the Decision Unit (DU). In general, information flows from the transport level, passes through the DLPF, PU and DU before feeding into the streaming server. Please note that there could be a feedback from the DU to the DLPF in order to adjust some parameters. On the other hand, the streaming server simply feeds the network with the aggregated encoder allowed rate, which is the combination of BL and EL content frames.

Figure 31 also emphasizes which time scale each unit is using. The DLPF receives information about the available bandwidth every RTT. It does not perform any aggregation of estimates. Therefore, the DLPF also provides information to the PU every RTT. When necessary, the PU accumulates some estimates before performing any prediction. One consequence is decoupling the time scale in the network and application levels. Therefore, we place the PU unit between the DLPF and the DU strategically. In such location, it will perform two important tasks that are providing information about prediction errors and decoupling the two different time scales.

As the PU plays a major role in our architecture, has a number of procedures, and absorbs different mathematical and statistical knowledge, we decided to describe it in a new chapter.

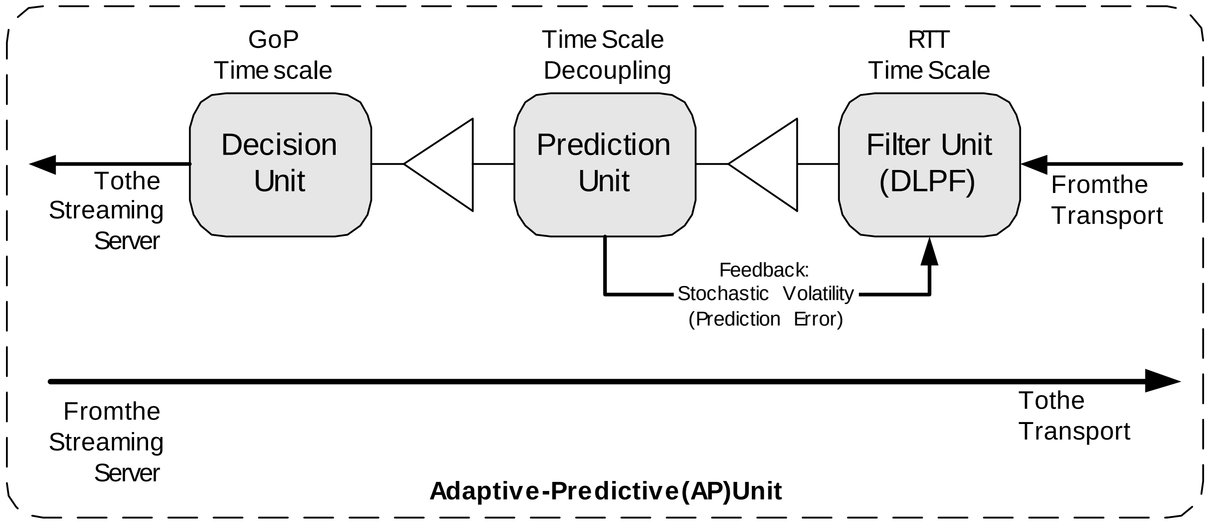

Figure 31 - The Adaptive-Predictive Unit

4.4.1. Filter Unit: Dynamic Low-Pass Filtering (DLPF)

Before delving into several algorithms for smoothing available bandwidth estimates from the network, we discuss our decision for placing the Filter Unit (DLPF) into our architecture. In the area of signal processing, a common approach to obtaining smoothness is to use low-pass filtering. In this work, we apply low-pass filtering techniques to provide the PU unit with a less variable time series. Exponential smoothing methods proved to be optimal for a very general class of state-space models and their correspondent adaptive methods demonstrated to have trustworthy improved forecast accuracy over non-adaptive smoothing [215]. In our scope of application, the main reason for the need of the DLPF in our architecture is that the performance of applications in explicit feedback networks is still dependent on the variations of the background traffic. Its main objective is smoothing the stochastic behavior of the network information throughout time. Recall that Shakkottai [201] and Karnik [106] found out that when the RTT is large and rate variations are fast, the end-system performance does not improve. Also, recall that one design decision is reducing complexity for matching VBR encoding rate to VBR available bandwidth information. Therefore, we decide to filter the high frequency components of the available bandwidth signal thus providing a smooth available rate to the PU. In other words, we remove high rate changes in the available bandwidth information from the transport protocol. One can see DLPF as a cautious decision to prevent abrupt changes coming from the network. Figure 32 shows the DLPF in details.

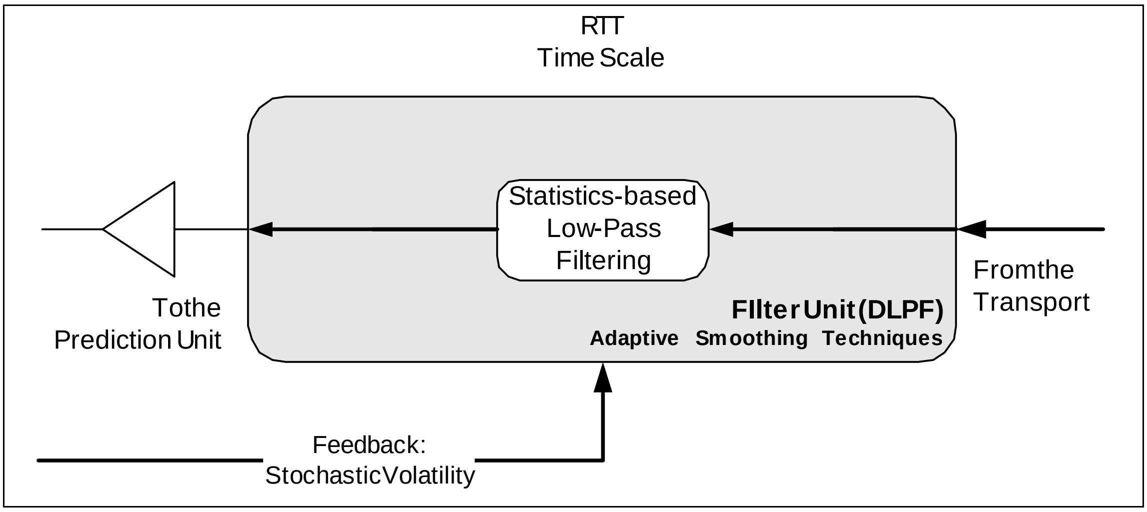

Figure 32 – Filter Unit (DLPF)

Exponential smoothing techniques have long been the methods of choice for both univariate filtering and forecasting due to their accuracy and ease of use. Researchers from a variety of fields are increasingly utilizing them because of their simplicity and overall good performance. They also suggested their use for short-term prediction. For instance, in a previous work, we proposed the use of two time series-based models for predicting handoff load in a call admission control (CAC) scheme for wireless mobile networks [49] [50]. In general, smoothing is filtering. All smoothing techniques accept any highly dynamic time series as input, removes short-term variations or noise, and reveal the essential intrinsic information, such as mean, trend, or seasonality. Among the simplest methods is the ordinary (simple) exponential smoothing, which assumes no trend and no seasonality. In this chapter, we will work with Exponentially Weighted Moving Average (EWMA), also known as exponential smoothing. Thereafter, we will use the terms Low Pass Filter and exponential smoothing interchangeably. As a final argument, we argue that although there are a number of adaptive methods available in the literature, we were not able to find any evidence in favor of any one. Therefore, it leaves space for evaluation in the scope of our applications by undertaking a careful comparison study [215].

Let Y_t denote a univariate time series. EWMA techniques assume that the forecast $\hat{Y}$ for period $t+h$ is given by a variable level $\hat{a}$ at period t, $Y_{t+h} = \hat{a}_t$, which is recursively estimated by a weighted average of the observed and the predicted value for Y_t.

$$\hat{a}_t = \alpha Y_t + (1-\alpha)\hat{Y}_t,$$

$$\hat{a}_t = \alpha Y_t + (1-\alpha)\hat{a}_{t-1}$$

where $0 < \alpha < 1$. There are several terms for the constant α. The most common names are smoothing parameter (constant), stepsize, learning rate, or gain. It governs the rate at which new information is combined with the existing knowledge about the time series. The main drawback of this technique is the choice of the smoothing parameter since setting it close to 1 could give rise to a highly reactive model. On the contrary, choosing the smoothing constant close to 0 could lead to an insensitive model. Researchers argue that the smoothing parameter should vary over time, in order to adjust to the latest characteristics of the data. For instance, this behavior will be certainly useful when there is a level shift in the time series. The model will adjust to data by allowing a greater weight on the most recent observation. In order to assist the selection of α, i.e. to improve awareness capability of the predictor, a number of adaptive methods have been recommended in the literature. In other words, such proposals enable the exponential smoothing parameters to adapt over time according to the characteristics of the time series. Their main advantages rely on the fact that there is no need to specify the filter gain previously. Adaptive procedures regulate the smoothing constant α whenever a change occurs in the time series basic structure. Therefore, α_t will change based on variations in the data pattern. Thereafter we will call these techniques Dynamic Low-Pass Filter (DLPF). We refer the interested reader to the Appendix 1 - Filters in Time Domain, for an in-depth review on filtering in time domain.

For the DLPF unit, we implemented several algorithms for the choice of the smoothing parameter. As we stated before, the architecture is flexible enough to use the most appropriate features in each component. Consider now the following equation

$$\begin{aligned}\hat{\upsilon}^n &= \hat{\upsilon}^{n-1} - \alpha^n\left(\hat{\upsilon}^{n-1} - \hat{X}^n\right)\\ &= \left(1-\alpha^n\right)\hat{\upsilon}^{n-1} + \alpha^n \hat{X}^n\end{aligned}$$

This is the general form of the Low-Pass filter [68], where

$\hat{\upsilon}^n$ is the new estimate at time n,

$\hat{\upsilon}^{n-1}$ is the previous estimate at time *n-1*,

$\hat{X}^n$ is the current traffic sample, and

$\alpha^n \in [0,1]$ is the filter gain, stepsize, or smoothing parameter.

We selected, implemented and tested several stochastic filter gain formulas, which overcome the drawbacks of deterministic rules. As a rule of thumb, stochastic filter gain formulas, also called adaptive stepsize rules, react to the errors in the estimation with respect to the actual sample. Although such selection, implementation and testing are not exhaustive, the chosen algorithms reflect and are representative for the considerable amount of work on the adaptive simple exponential smoothing techniques [68]. They provide a solid base with which it is possible to evaluate other similar methods.

Table 3 presents the adaptive smoothing techniques implemented in our architecture.

We now present implementation details about several adaptive filter gain rules, namely Kesten [115], Mirozahmedov [154], Gaivoronski [68], Trigg & Leach [217] [218], Whybark [241], Dennis [48], Tukey [219], FIR [77], and Smooth Transition Exponential Smoothing (STES) [215]. We also describe some curve-fitting techniques such as Smoothing Spline [82] and a local regression smoothing technique called Locally Weighted Scatter Plot Smooth (LOWESS) [37]. For each algorithm, we will show its basic formulation, along with a simulation result. We think such simulations indispensable, since we need to verify whether all chosen algorithms meet our requirements. Recall that the requirements for a suitable dynamic filter include having parsimonious parameterization and working well with both stationary and non-stationary data. In other words, a good adaptive filter should follow the low frequency components (e.g. the first-order moment). Therefore, in the following experiments, we undertake simulations by generating non-stationary data. We generate data with 1000 samples from a light tail probability distribution function, namely Lognormal. However, we set different first- and second-order moments every 200 samples. This behavior fully characterizes the data as a non-stationary time series.

Table 3 - Adaptive Step-Size Rules Available at the DLPF

Kesten's Rule	General FIR
Mirozahmedov's Rule	Dennis' Rule
Gaivoronski's Rule	Tukey's Smoothing
Trigg & Leach's Rule	Smooth Transition Exponential Smoothing

Whybark's Rule	Smoothing Spline / LOWESS Smoother

Kesten's Rule

Kesten's rule [68][115] has the following formulation for the adaptation of the smoothing parameter.

$$\alpha^n = \alpha_0 \left(\frac{a}{b + K^n} \right) \qquad ,$$

where a, b and α_0 are positive constants.

K^n tries to keep track the number of times that the error changes signal. In other words, Kesten's rule decreases the smooth parameter if the inner product of two successive errors is negative. Otherwise, it leaves it unaltered. One should recursively calculate it as follows:

$$K^n = \begin{cases} n, & if\ n = 1,2 \\ K^{n-1} + 1_{\{\hat{\varepsilon}^n \hat{\varepsilon}^{n-1} < 0\}}, & n > 2 \end{cases} ,$$

$$\text{where } 1_{\{X\}} = \begin{cases} 1, & if\ X\ is\,true \\ 0, & otherwise \end{cases} ,$$

where $\hat{\varepsilon}^n$ is the estimation error at time n and $1_{\{X\}}$ is the indicator function.

Figure 33 shows an example of Kesten's rule use. One can clearly see that the filtered signal keeps track of the mean value of the original traffic. As a recursive process, this algorithm is highly suitable to our purposes, thus meeting system requirements.

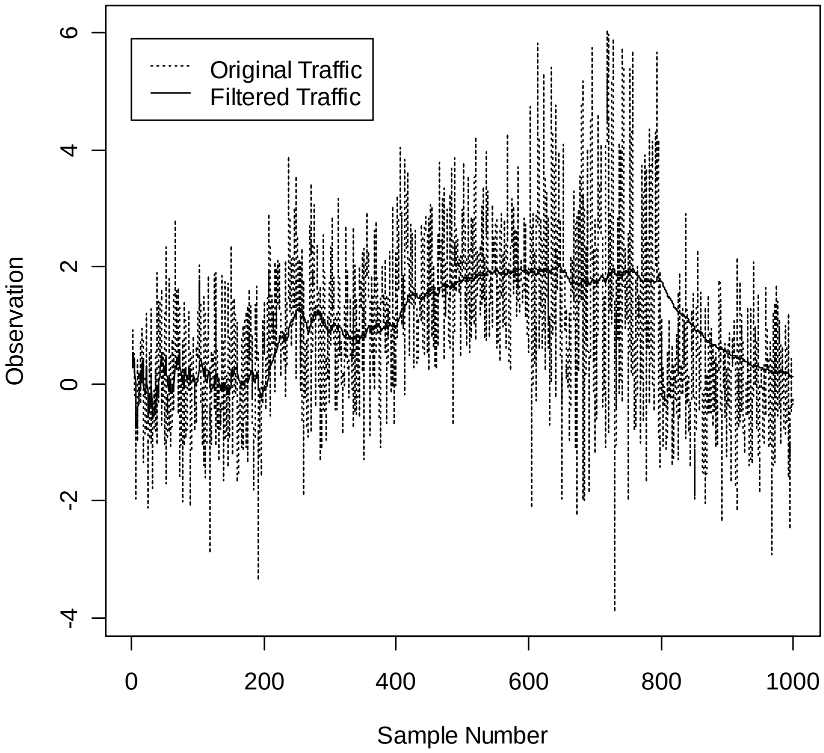

Figure 33 - Kesten's Rule Simulation

Mirozahmedov's Rule

Mirozahmedov's rule [68][154]follows the same principle of Kesten's idea that is changing the filter gain in response to the product of two successive errors. The formulation of Mirozahmedov's rule is simple as follows:

$$\alpha^n = \alpha^{n-1} \exp\left[\left(a\hat{\varepsilon}^n \hat{\varepsilon}^{n-1} - \delta\right)\alpha^{n-1}\right],$$

where a and δ are positive constants, and $\hat{\varepsilon}^n$ is the estimation error at time n.

Figure 34 shows an example of Mirozahmedov's rule simulation. One should observe that the filtered signal is not as smooth as the one from Kesten's rule. However, we still consider this algorithm suitable to our purposes, since such variability is easy to handle.

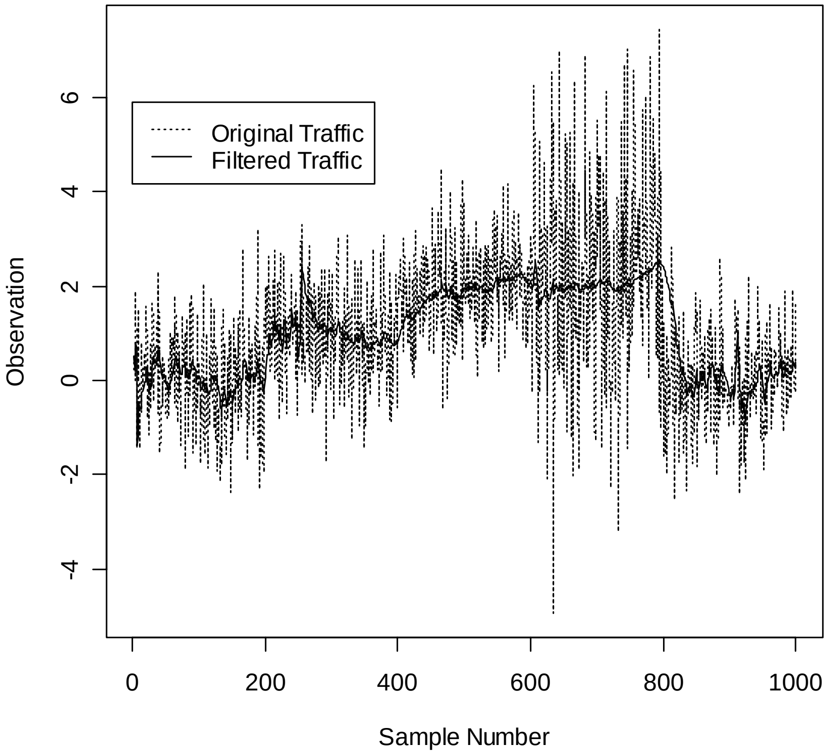

Figure 34 - Mirozahmedov's Rule Simulation

Gaivoronski's Rule

Gaivoronski's rule [64][68] calculates the filter gain as a function of the difference in the values of the smoothed estimate. We refer the interested reader to [68] for details on the motivation behind Gaivoronski's algorithm. The formulation of Gaivoronski's algorithm is:

$$\alpha^n = \begin{cases} \gamma_1 \alpha^{n-1} & if\ \Phi^{n-1} \le \gamma_2 \\ \alpha^{n-1} & otherwise \end{cases}$$

where,

$$\Phi^n = \frac{\left|\hat{\theta}^{n-k} - \hat{\theta}^n\right|}{\sum_{i=n-k}^{n-1} \left|\hat{\theta}^i - \hat{\theta}^{i+1}\right|},$$

γ_1 and γ_2 are positive constants, k is the number of iterations, and $\hat{\theta}^n$ is the estimate at time n.

Figure 35 shows an example of Gaivoronski's rule simulation, which filtered signal is the most smoothed compared to Kesten' and Mirozahmedov's algorithm.

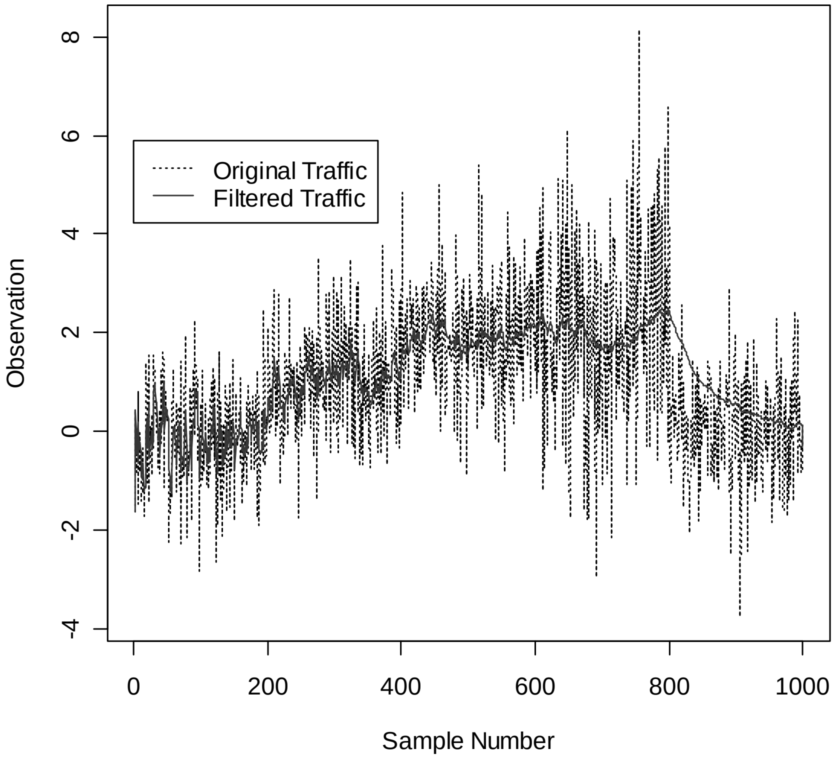

Figure 35 - Gaivoronski's rule Simulation

Trigg & Leach's Rule

As we mentioned earlier in this chapter, we used Trigg & Leach's adaptive algorithm as part of a novel call admission control scheme for wireless and mobile networks [49][50]. We applied Trigg & Leach's rule to predict, instead of filtering, the expected bandwidth of future handoffs. Although there is no consensus about the most useful adaptive approach, Trigg and Leach is indeed the most widely used procedure. The main advantage for using Trigg & Leach's rule is that it is effortless, does not impose computation overhead, and requires only a small amount of saved data to perform one-step ahead forecasting. In Trigg & Leach's rule, the track signal T^n monitors the estimation process. In the original algorithm, called Trigg's rule [218], the adaptive stepsize and the track was the same variable. The following equations define Trigg & Leach algorithm:

$$T^n = \frac{S^n}{M^n},$$

$$\alpha^n = \left|T^n\right|,$$

where

$$S^n = (1-\beta)S^{n-1} + \beta\hat{\varepsilon}^n .$$

S^n represents the smoothed weighted sum of the observed errors and $\hat{\varepsilon}^n$ is the estimation error at time *n*. M^n represents mean absolute deviation and has the following formulation.

$$M^n = (1-\beta)M^{n-1} + \beta\left|\hat{\varepsilon}^n\right|$$

Figure 36 shows an example of Trigg & Leach's rule simulation. As expected, Trigg & Leach's algorithm achieved similar performance as the previous rules.

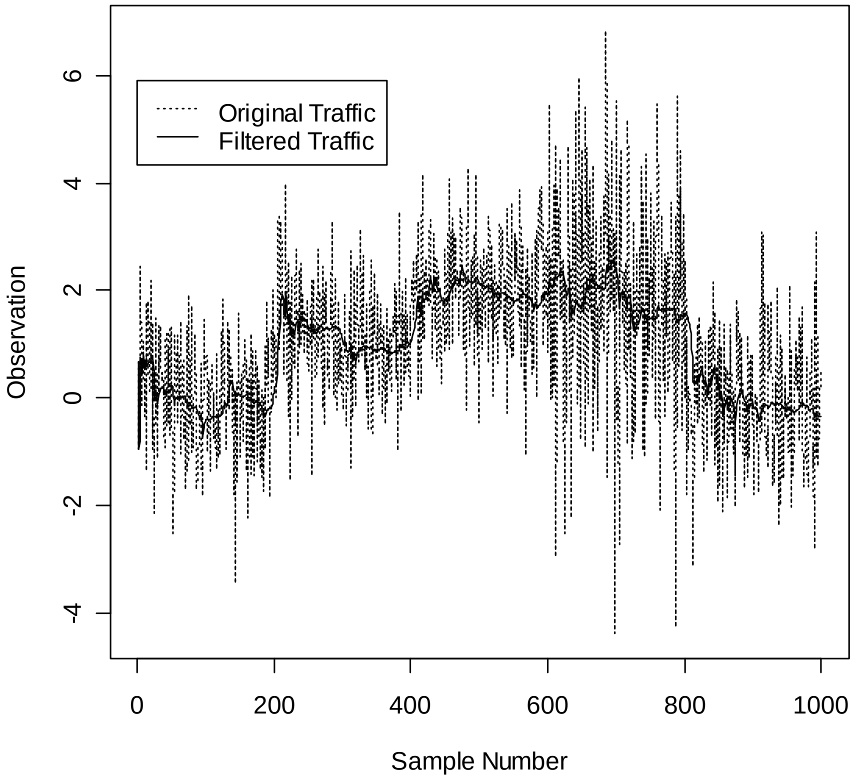

Figure 36 - Trigg & Leach's rule simulation

Whybark's Rule

Whybark's algorithm relies on the assumption that the filter gain can be non-continuous and only a few values will be enough for the adaptation [241]. Whybark imposes three levels for the stochastic stepsize, which in turn depends on a tracking signal δ^n. δ^n has the following formulation:

$$\delta^n = \begin{cases} 1, & if\ \left|\varepsilon^n\right| > 4\sigma \\ 1, & if\ \left|\varepsilon^n\right| > 1.2\sigma\ \&\ \left|\varepsilon^{n-1}\right| > 1.2\sigma\ \&\ \varepsilon^n\varepsilon^{n-1} > 0 \\ 0, & otherwise \end{cases}$$

where $\hat{\varepsilon}^n$ is the estimation error at time *n* and σ is the standard deviation of the estimates. In Whybark's rule, the step size can assume the values 0.2, 0.4 and 0.8. It will depend on the signal δ^n by following the rule below:

$$\alpha^n = \begin{cases} 0.8, & if\ \delta^n = 1 \\ 0.4, & if\ \delta^n = 0\ and\ \delta^{n-1} = 1 \\ 0.2, & otherwise \end{cases}$$

Figure 37 shows a simulation of the Whybark's rule. As expected, Whybark's algorithm did not achieve a good performance as the previous rules. This is due to the fact that the limitation on step sizes values imposes high variability. We argue that Whybark's algorithm does not represent a good choice for implementation in our architecture, since it cannot smooth the original signal at the desired level.

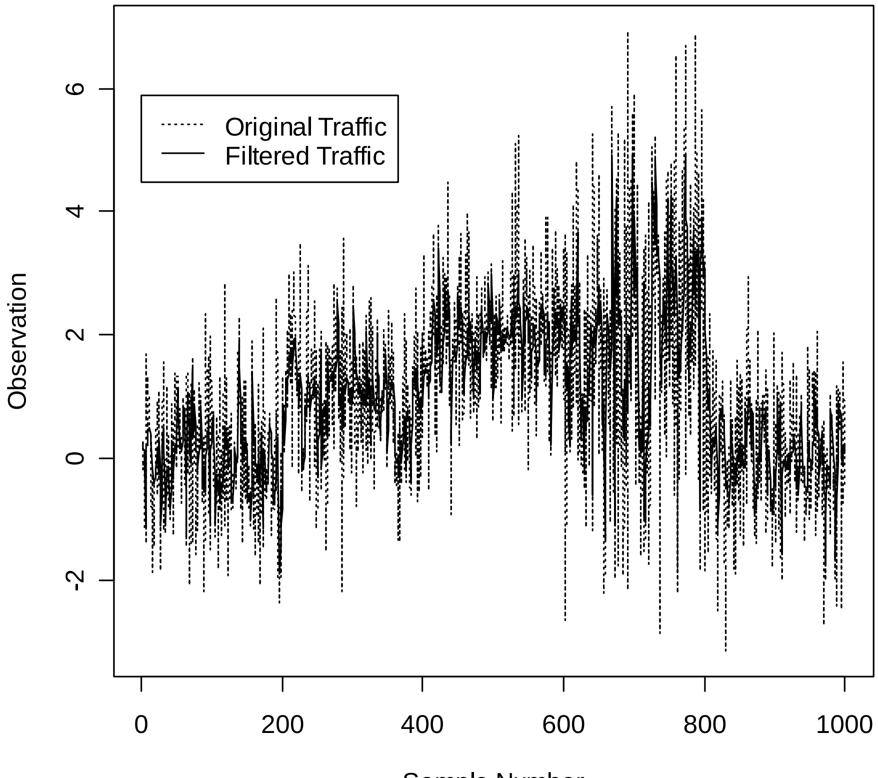

Figure 37 - Whybark's rule simulation

Dennis' Rule

Dennis' rule [48] assumes non-continuous domain for the filter gain in the same way as Whybark's rule. It follows the algorithm below:

$$N^n = \begin{cases} 1, & if\ \varepsilon^n \varepsilon^{n-1} \le 0 \\ N^{n-1} + 1, & otherwise \end{cases},$$

$$\alpha^n = \begin{cases} B, & if\ \ N^n < L \\ \min\{\alpha^{n-1} + \lambda, 1\}, & otherwise \end{cases},$$

where $\hat{\varepsilon}^n$ is the estimation error at time n, $B = 0.2$, $L = 2$, $\lambda = 0.6$ are common parameters. N^n keeps track of the frequency that the error changes its signal. Figure 38 shows a simulation result of the Dennis' rule. Dennis's algorithm did not achieve a good performance as well. As similar to Whybark's rule, the limitation on step sizes values imposes high variability to α^n. We will discard the use of Dennis' algorithm in our architecture.

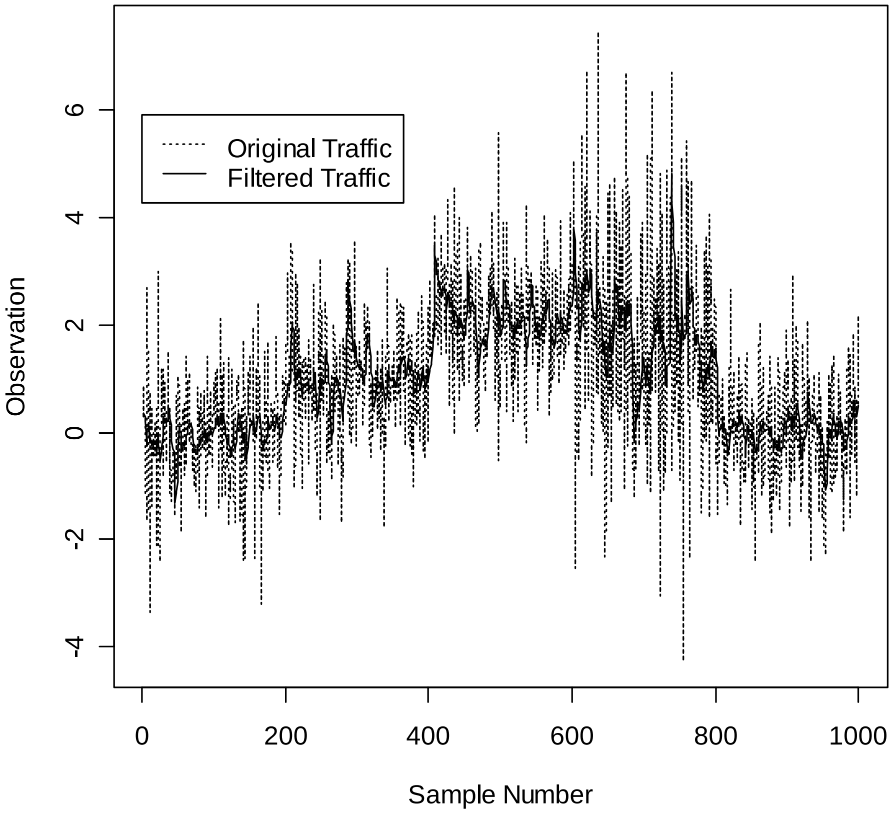

Figure 38 - Dennis' rule simulation

Smoothing Spline

Another way to get a smooth version of the original data is to use curve-fitting techniques. Although they have a close relation to low pass filtering mechanisms, smoothing with curve fitting techniques bears no concern to modeling or extracting parameters from data. The main objective is simply to obtain a smooth curve through data. This type of smoothing is also called nonparametric fitting or interpolation [71][82]. The main idea behind curve fitting is that the close neighbors of a sample contain functional information about the "proper" value of the sample, thus eliminating noise. To this end, every sample in the original series is replaced by a weighted average of itself and its close samples.

Smoothing spline consists of the approximation of a function using a series of polynomials over adjacent intervals with continuous derivatives. In other words, spline is a series of cubic polynomials that fits to a group of consecutive values. Such polynomials must be continuous and must have a continuous first derivative. Fitting a smoothing spline to data means finding a function f that minimizes

$$\frac{1}{n}\sum_{i=1}^{n}\left(\left(y_i - f\left(x_i\right)\right)^2\right) + \lambda\int_{x_1}^{x_m}\left(f^{m}(x)u\right)^2 du\,,$$

where $m \text{ and } n$ define the number of samples for applying f and λ is a weighting factor. We will not give more details about smoothing spline and we refer the interested reader to [71].

There is an important advice about using non-parametric curve fitting. After processing the original data (e.g., smoothing with curve fitting), one should not fit data with a parametric model. As one has the smoothed data only, one should consider using the last sample as the last estimate.

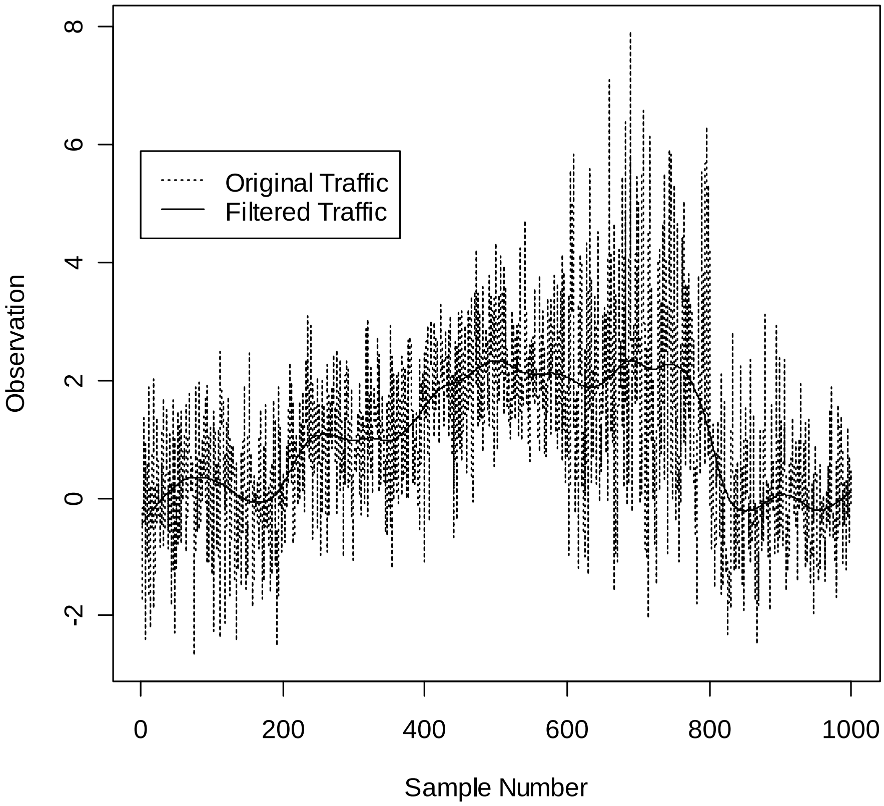

Figure 39 - Smoothing Spline simulation

Figure 39 shows a simulation result when using smoothing spline, which achieved excellent smoothing effect. We would like to emphasize that such technique is not suitable for "online" use that is applying the mechanism for every new sample arrival. We must wait for the predefined number of samples that we use to compute each smoothed value. In other words, smoothing spline and similar techniques (e.g. Loess, Lowess, Tukey, and Savitzky-Golay etc) requires a span of values. In fact, span refers to a window of neighboring points, not only a set of previous samples.

LOWESS Smoother

LOWESS method performs smoothing using linear least squares fitting and a first-order polynomial approximation [37][38]. The term LOWESS means locally weighted scatter plot smoothing. LOWESS smoothing is a local process, since the neighboring samples within the span determine each smoothed value. LOWESS is also a method based on local polynomial fits. It starts with a local polynomial least square fit and then refines it, i.e. re-smoothes many times, until the algorithm reaches a predefined number of iterations. Fitting with LOWESS means finding coefficients $\{\beta_j\}_{j=0}^{p}$ for the polynomial in a neighborhood of x that minimizes

$$\frac{1}{N}\sum_{i=1}^{N} w_{ki}(x)\left(Y_i - \sum_{j=0}^{p}\beta_j x^j\right)^2 ,$$

where N is the number of points in the neighborhood, p is the smoothing parameter, and w_{ki} are weighting factors.

We will not give more details about LOWESS and we refer the interested reader to [37]

Figure 40 presents a simulation result when using Lowess smoothing method, which achieved excellent smoothing effect as well.

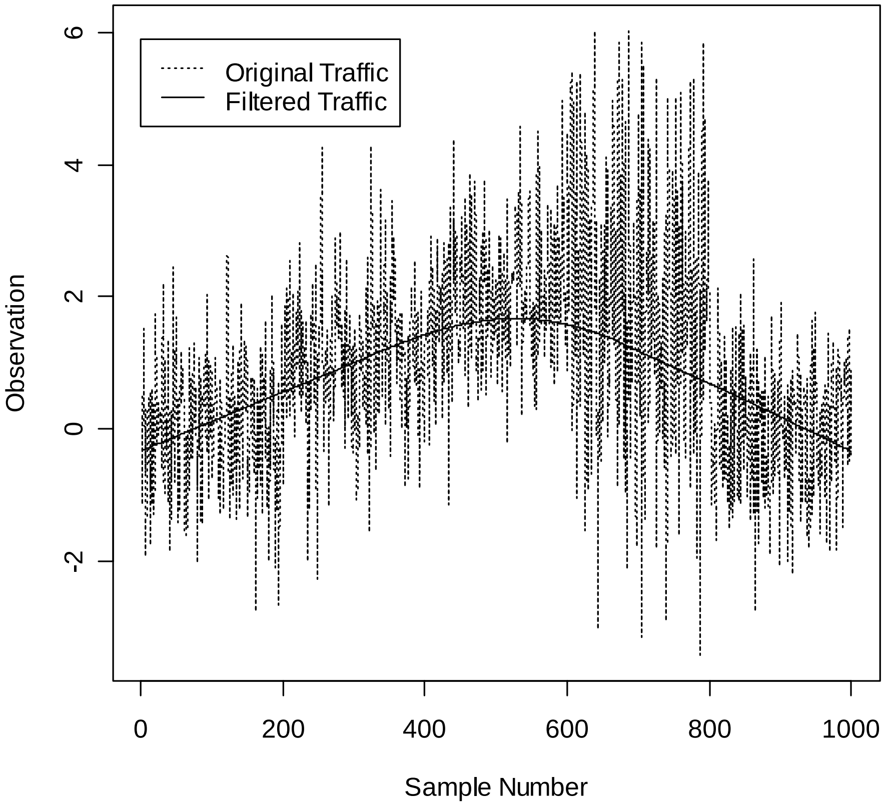

Figure 40 - LOWESS simulation

Tukey's (Running Median) Smoothing

One should notice that the smoothing process is dynamic. Therefore, one can smooth sample data repeatedly, with different spans and kinds of averaging. Tukey [219] introduced a notation for identifying long smoothing plans. In Tukey's short notation, 3 means running median of length 3, 3R stands for Repeated 3 until convergence, and S for Splitting of horizontal stretches of length 2 or 3 [172]. Additionally, a hanning (code H) operation multiplies the three values in a window by .25, .5 and .25, respectively, and sums the results. In this work, we use several combination for the smoother, namely “3RS3R”, “3RSS”, “3RSR”, “3R”, “3”, “S”.

Figure 41 presents the simulation result for the “3RS3R” smoother, which did not achieve good results for filtering noise from data.

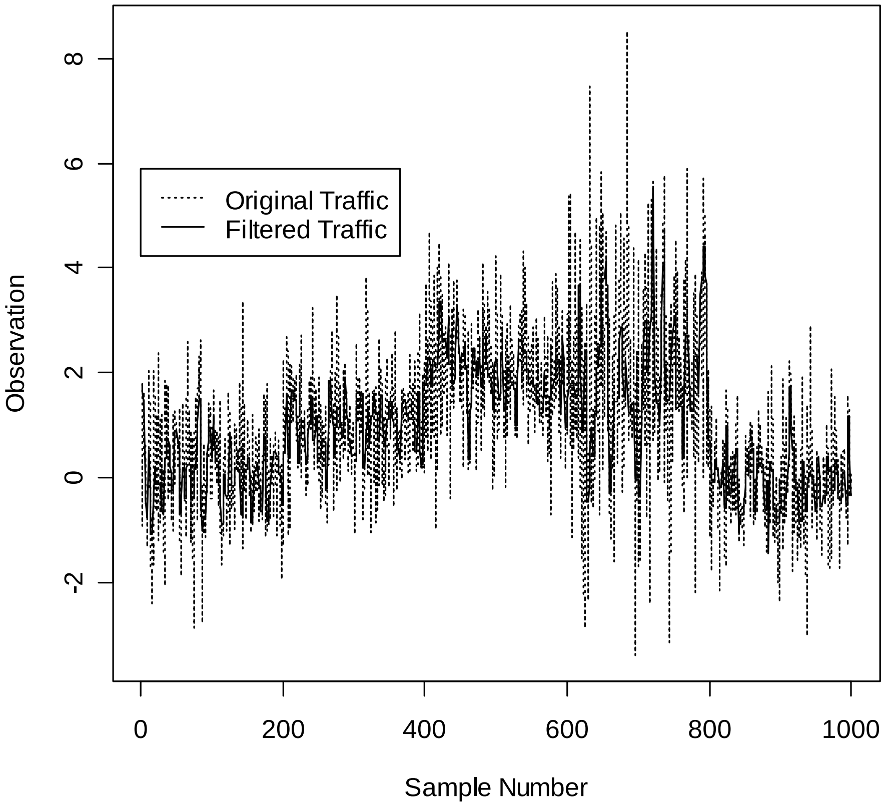

Figure 41 - Tukey's Smoothing simulation

General Causal FIR Filter

Filtering in time domain imposes restrictions to causal filters (see Appendix 1), especially when the filter must deal with real time applications. In such a case, the whole process deals with continuous data stream arriving at real time and yields output-filtered values at the same rate as the arrival one. In the scope of this thesis, the available bandwidth information arrives in real time, thus creating a real limitation by narrowing our decision scope to causal filters. Researchers in Signal Processing have relying on Finite Impulse Response filters (FIR), sometimes called Convolution filters, for a long time, since they are simple to implement and are stable, e.g., it is completely stable at all frequencies regardless its order (see Appendix 1). The filter coefficients are the impulse response of the filter (i.e., they determine the characteristics of a given filter) whereas the filter order is essentially the number of previous inputs used to evaluate the current output.

A general causal FIR filter has the following formulation:

$$y_n = \sum_{k=0}^{M} \alpha_k x_{n-k}$$

where α_k is filter vector of coefficients, $\{x_n\}_{n=0}^{\infty}$ is the raw time series data, $\{y_n\}_{n=M}^{\infty}$ is the output signal, and M is the filter order. For the moving average representation,

$$\alpha_k = \frac{2\alpha_k^{'}}{(M+1)(M+2)}.$$

We implemented a variation of this approach, namely the Almon lag specification [67], where the filter coefficients are determined following a second-order polynomial, such as

$$\alpha_k = \phi_0 + \phi_1 i + \phi_2 i^2, \quad i = 0, \ldots M \, .$$

Based on previous simulations, we set the FIR filter order as $M = 50$. According to the FIR specification, one gains additional smoothing properties when setting a higher value to this parameter. The drawback in this approach is a higher memory and computational requirements. Please note, that we re-estimate the filter coefficients periodically, every M input samples.

Figure 42 shows an example of a causal FIR filter simulation. As expected, using FIR Filters we achieved good performance as the previous rules.

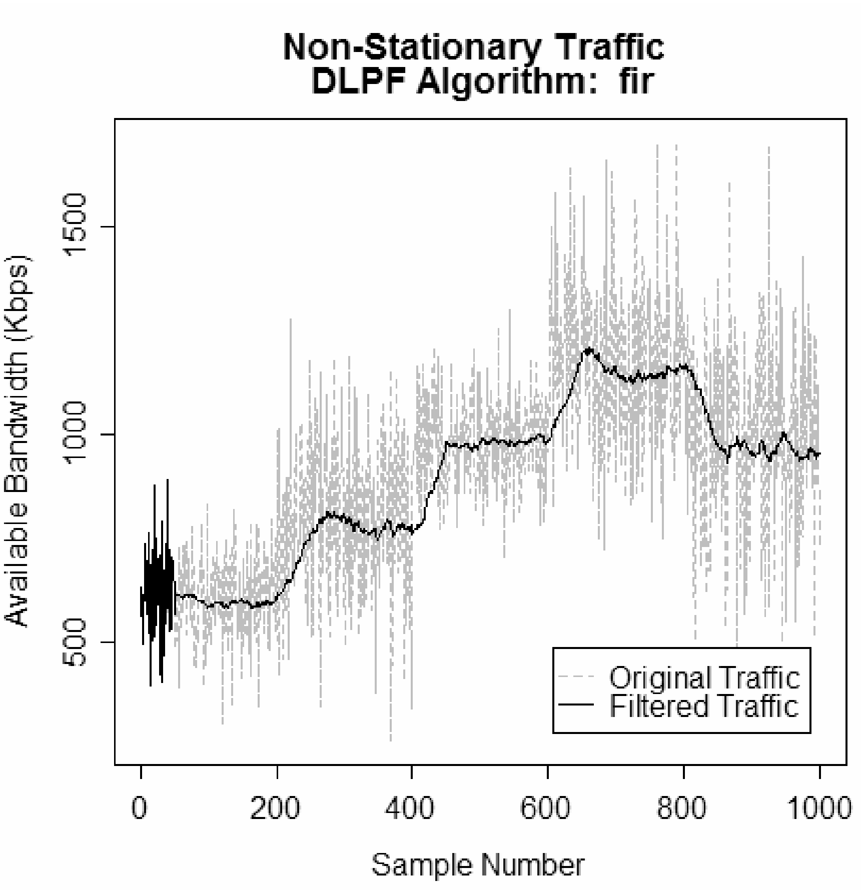

Figure 42 – Causal FIR Filter simulation

Smooth Transition Exponential Smoothing (STES)

The STES adaptive exponential smoothing method is a type of smooth transition (ST) [215]. In ST models, parameters depend on a continuous function of a transition variable. The following equation represents a Smooth Transition Autoregressive model (STAR),

$$y_n = \sum_{j=0}^{N} \alpha_{j,n} y_{n-j}$$

if $\alpha_{j,n}$ is not constant and is a monotonically increasing function of the transition variable, V_n. For instance, the following formulation for $\alpha_{j,n}$, transforms the autoregressive model in the STAR model:

$$\alpha_{j,n} = \frac{\omega}{1 + \exp(\beta + \gamma V_n)},$$

where ω, β and γ are constant parameters, and $\gamma < 0 < 0$ and $\omega > 0$. The parameters of smooth transition models can be estimated using nonlinear least squares.

Based on this general ST model, Taylor [215] proposes a Smooth Transition Adaptive Exponential Smoothing (STES) method, with smoothing parameter at defined as a logistic function of a user-specified transition variable. The STES model is written as

$$y_n = \alpha_n x_{n-1} + (1 - \alpha_n) y_{n-1}$$

where

$$\alpha_n = \frac{\omega}{1 + \exp(\beta + \gamma V_n)}$$

If $\gamma < 0$, α_n is a monotonically increasing function of V_n. Therefore, as V_n increases, the weight on x_{n-1} rises, and correspondingly the weight on y_{n-1} decreases. The logistic function restricts at to lie between zero and one, which is a requirement for stability in filtering. Please note that the choice of the transition variable, V_n, is essential to achieve good results when using the STES method. Since the value of the smoothing parameter depends on the extent of the forecast error, the best choices for the transition variable are the square, the mean squared, the mean absolute, or the mean percentage value of the forecast error [215]. One could also combine STES methods with the previously presented adaptive filters. As such, the transition variable could be the resulting adaptive stepsize from any adaptive exponential smoothing technique, e.g., the Trigg and Leach parameter. Taylor claims that STES method, enables recalibration of the existing adaptive methods. In this thesis, we implemented the STES method with the squared error from the previous period as transition variable, V_n.

Figure 43 shows an example of STES simulation. As expected, STES method achieved similar performance as the previous rules.

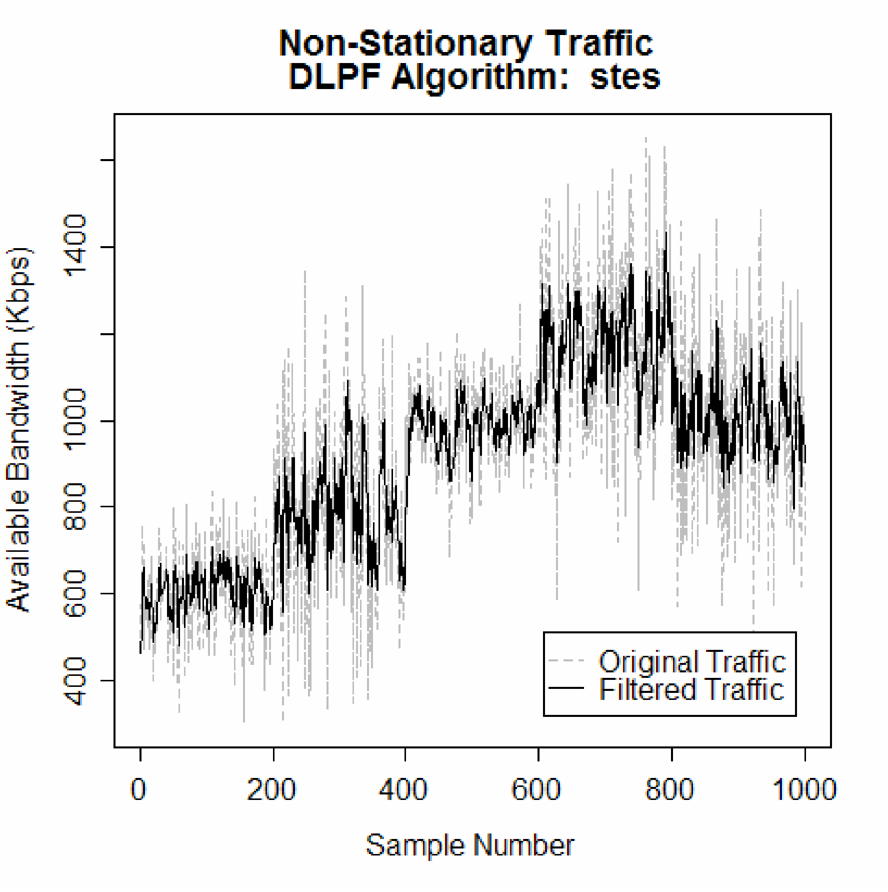

Figure 43 - STES simulation

4.5. Performance Analysis

Our approach to undertaking performance evaluation of our novel architecture is to separate it by system components. That is, we now start a performance analysis of the DLPF unit. We will undertake the complete performance evaluation in the next chapter, including the impact of the prediction and the control decision functional blocks on the overall system performance.

4.5.1. Evaluation Scenario and Description

The development of the MPEG-4 Fine Granularity Scalability (FGS) video standard had as consequence an improved flexibility for video streaming over networks with variable available bandwidth. FGS encoding design tries to cover a wide range of bandwidth while maintaining a simple scalable structure [138] [176] [198]. Within the FGS encoding framework, a base layer (BL) and one enhancement layer (EL) compose the video. The main advantage of FGS compared to conventional scalable video (e.g., spatial scalability, layered encoding, etc) is that the EL stream can be truncated anywhere at the granularity of bits within each frame before transmission, thus providing partial enhancement proportional to the number of bits decoded at the frame level. Using this approach, video servers can adapt the streamed video to the network conditions with no changes in the quantization parameters, i.e. re-encoding, which is computationally intensive. The issue of how video servers can possibly adapt to network conditions is the topic of this thesis as well as an ongoing research in the multimedia and Internet community.

In a similar approach to the performance evaluation that we carried out in chapter 2, in this chapter we use the framework for evaluating the streaming of FGS video with rate–distortion traces provided by Seeling et al. [198] along with the guidelines described in [199]. We continue to use the Peak Signal to Noise Ratio (PSNR) for our numerical studies, although we are aware that subjectively perceived video quality is very complex to evaluate. With the evaluation framework for FGS video streaming provided in [198] it is easy to use the video quality as performance metric for video. We use the rate-distortion traces of the videos from reference [46]. We choose the movies with a variety of scenes characteristics, following the guidelines in [199], which describes the importance of selecting as many different videos as possible from the several genres available. Such variety in videos traces provides different frame quality properties, thus putting our architecture under a number of network loads. To this end, we select a priori a set of four movie types, namely Thriller (The Firm), Science Fiction (Star Wars), TV Show (Oprah) and Cartoon (Toy Story).

In order to undertake a proper quantitative evaluation, we first grouped frames in Group of Pictures (GOP), and then we use GOP-based metrics to evaluate the performance of all system components. In essence, we follow some metrics described in [46]. Let $Q_n, n = 1 \ldots N$ be the quality of the n^{th} received GOP. The mean and the sample variance of the GOP quality are calculated as follows:

$$\overline{Q} = \frac{1}{N}\sum_{n=1}^{N} Q_n$$

is the mean quality, and

$$\sigma_Q^2 = \left(\frac{1}{N-1}\right)\sum_{n-1}^{N}\left[Q_n - \overline{Q}\right]^2$$

is the sample variance.

The most important metric in our context is the coefficient of variation, which is calculated as follows:

$$CoV_Q = \frac{\sigma_Q}{\overline{Q}}$$

It is worth noting that in Chapter 3 we used frame-base metrics (q metrics) when evaluating the received video quality since we had short video traces. For this evaluation, we have long video movies with 108000 frames grouped into a GoP pattern of 12 frames (see Appendix 2). We decide to follow a common procedure for performance evaluation of video streaming [65] [198] [199]. The methodology utilized by Galluccio et al [65] gives us confidence to define all metrics as GoP-based (Q metrics). In that work, they defined an analytical framework for the evaluation of the performance of a real-time MPEG video transmission system over wireless links. As it deals with real-time issues, the framework involves a rate controller instead of a streaming server. The rate controller in the encoder adapts the output sending-rate by adjusting the Quantizer Scale Parameter (QSP) according to the bandwidth fluctuations. The key point for the choice of GoP-based metrics for our performance analysis relies on the results presented in [65]. They carried out a performance evaluation in different situations, namely when using frame- or GoP-based feedback laws. Their obtained results show that both feedback laws achieve good performance in terms of quality (PSNR). However, results show that GoP-based solutions achieve higher average PSNR and higher stability. By adopting GoP-based time-scale and metrics in our architecture, we expect to maintain stable quality during the whole GoP while lessening variability.

We used some FGS encoded videos available at [46] that were encoded in QCIF format (176 x 144 pixels). Figure 44 to Figure 47 show, for the movie traces The Firm, Star Wars, Oprah and Toy Story, the quality for each frame. The plots show that there are considerable variations in quality in the base layer. Although we did not show here due to clarity in the presentation, most enhancement layers (from 200Kbps to 1400Kbps) also have large quality variations. However, the enhancement layer rate at 2Mbps has the lowest variation level for all videos.

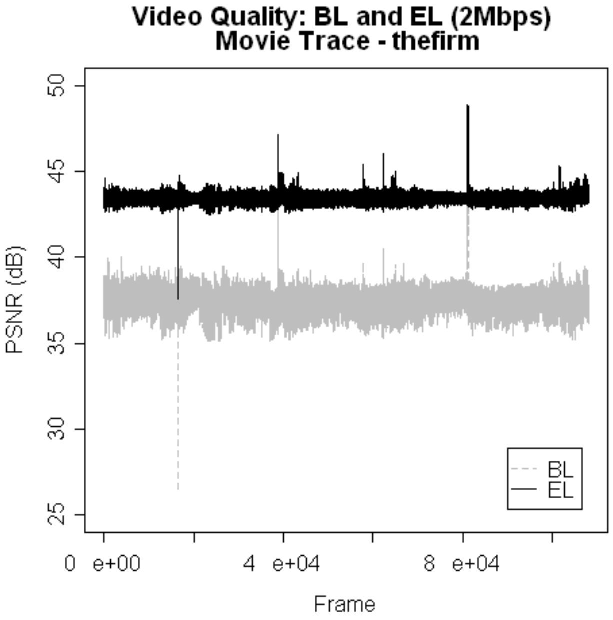

Figure 44 - BL and EL - Movie: The Firm

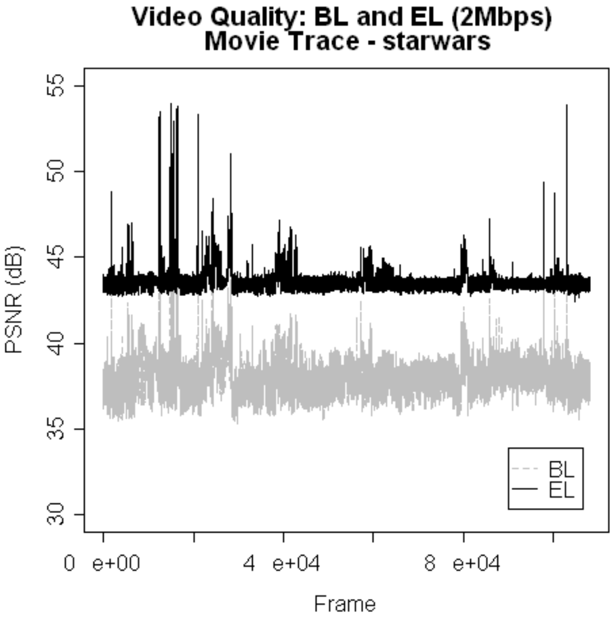

Figure 45 - BL and EL - Movie: Star Wars

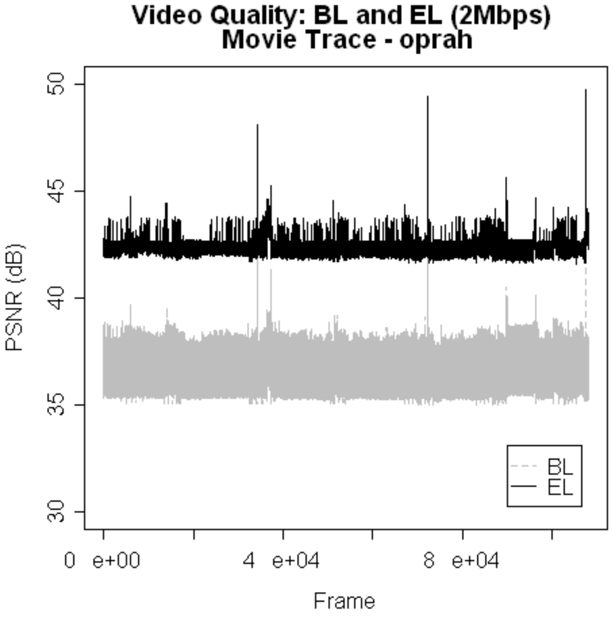

Figure 46 - BL and EL - Movie: Oprah

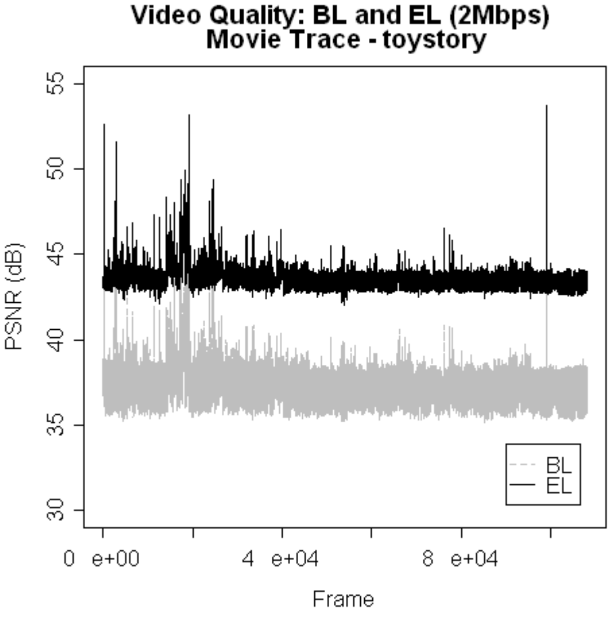

Figure 47 - BL and EL - Movie: Toy Story

We choose the video trace for the movie The Firm to give details about statistical properties of such FGS encoded video. We grouped video frames in a GOP with the length of 12 frames and the pattern "I B B P B B P B B P B B" (See Appendix 2). Figure 48 to Figure 51 present the histogram for I, B, P frames as well as the average quality in a GOP for the base layer (BL). In addition, Figure 52 to Figure 55 show similar properties for I, B and P frames in the EL (2Mbps).

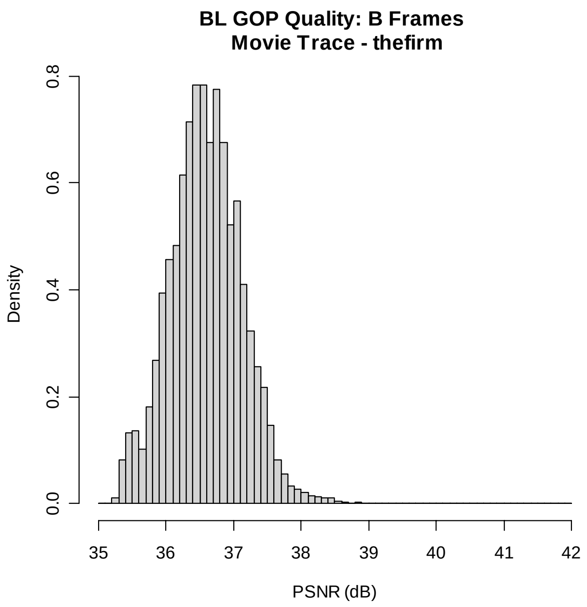

Figure 48 - Histogram for B Frames - BL

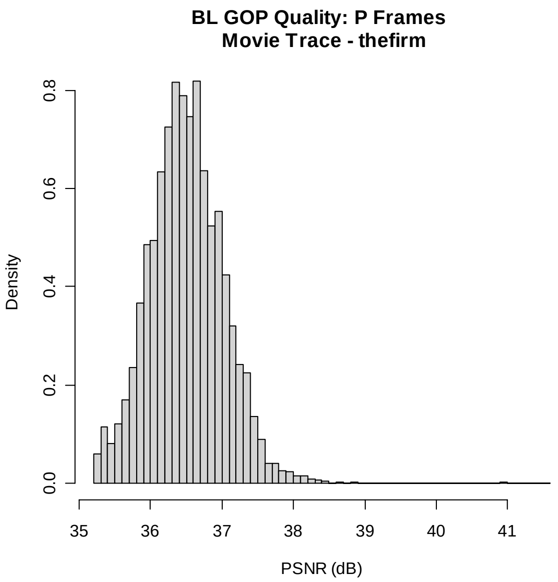

Figure 49 - Histogram for P Frames - BL

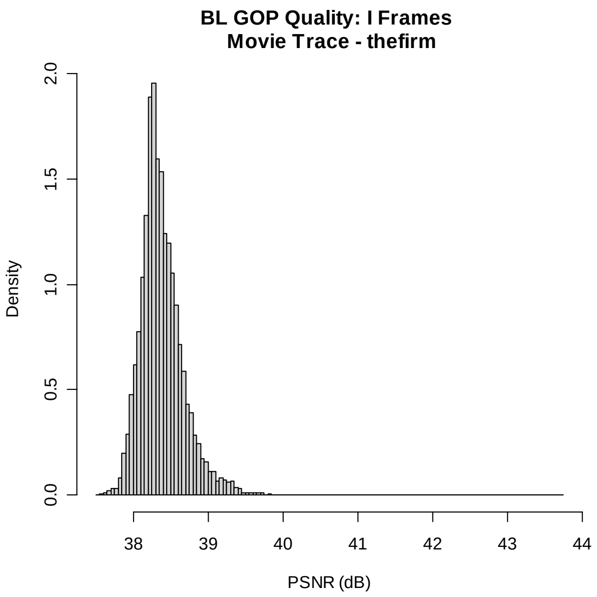

Figure 50 - Histogram for I Frames - BL

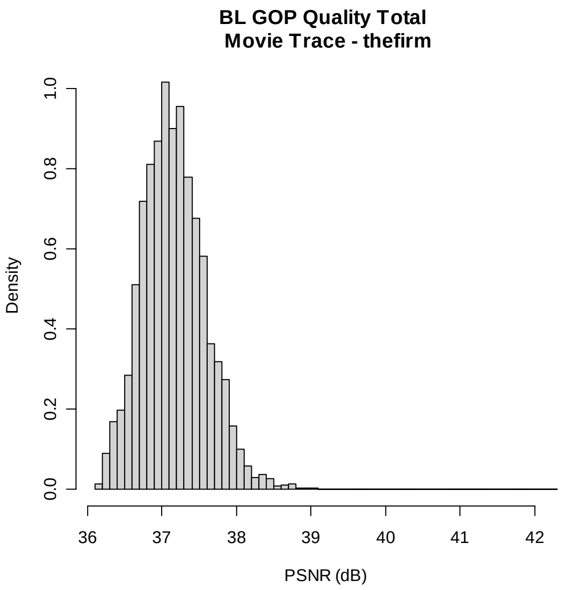

Figure 51 - Histogram for Frames - BL (Average)

The main difference between the statistical characteristics of BL and EL is that the latter has a sharper peak than the former, which characterizes low variability. Obviously, the level of EL is around 6-7dB higher than the BL quality level.

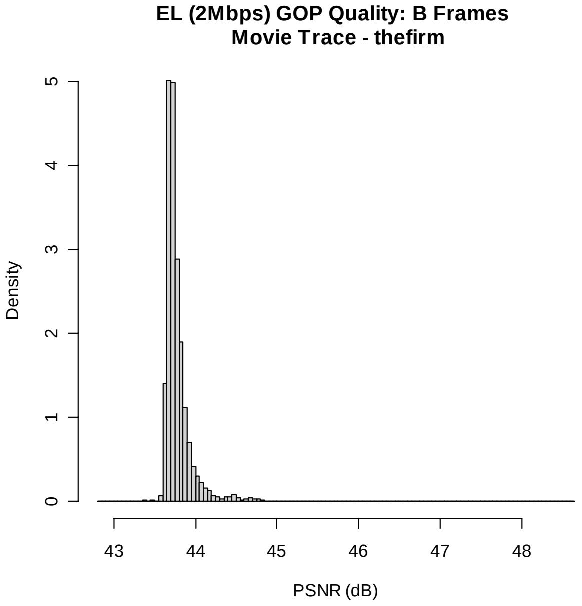

Figure 52 - Histogram for B Frames - EL (2Mbps)

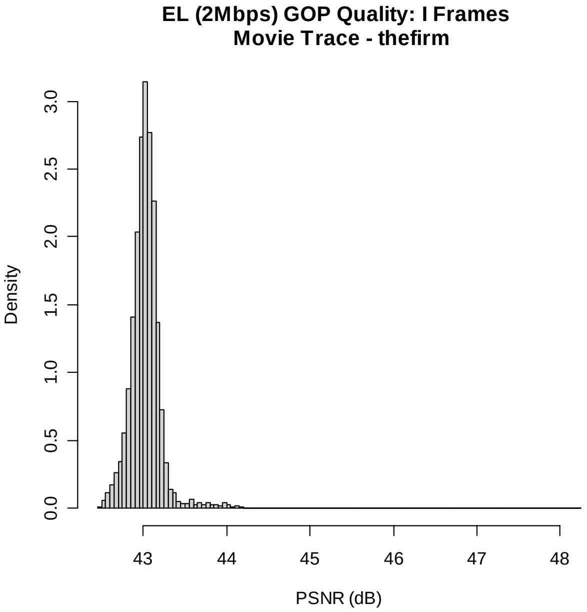

Figure 53 - Histogram for I Frames - EL (2Mbps)

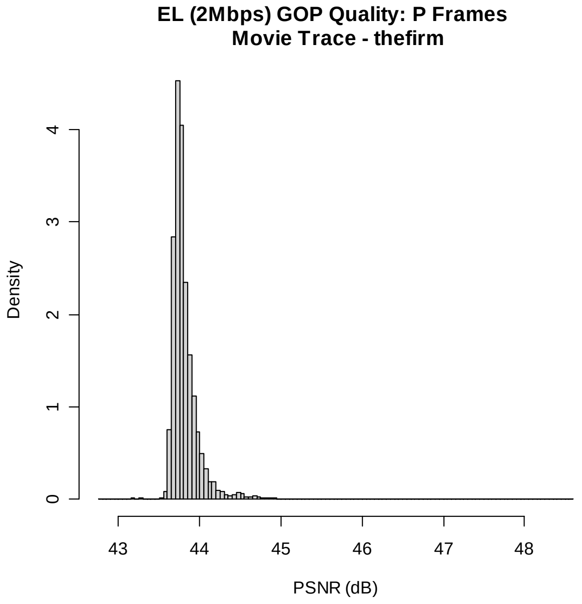

Figure 54 - Histogram for P Frames - EL (2Mbps)

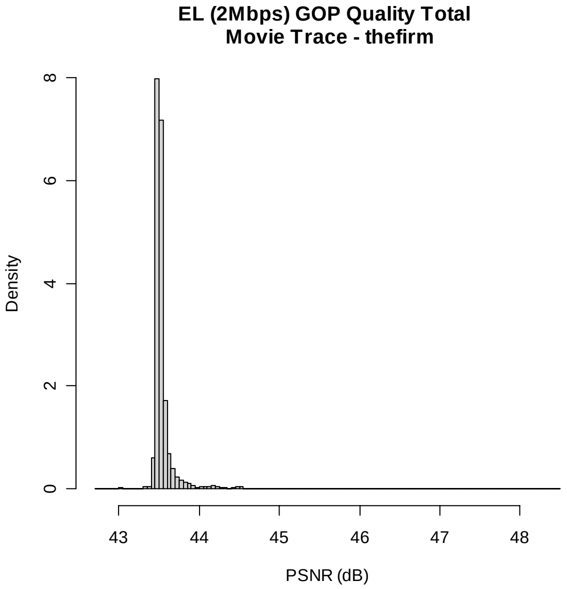

Figure 55 - Histogram for - EL (2Mbps) - Average

SIMULATION: CONFIGURATION AND PARAMETERIZATION

Before presenting the simulation results, we argue that it is important to have a clear understanding of the results. For instance, in stochastic simulation is imperative that simulation results allow assessment of the uncertainty in a given metric. To do that, a confidence interval gives an estimated range of values, which is likely to include an unknown population parameter. The width of the confidence interval gives us some idea about how uncertain we are about the unknown parameter [92]. However, in this thesis we are dealing with a massive amount of data, since each video trace is one-hour length and contains around 100,000 frames. For our exploratory data analysis, that characteristic makes the presentation of the simulation results a challenge. In such situation, it is prohibitive to show the confidence intervals to indicate statistical significance, since it will worse exactness and comprehensibility on plots. For that reason, we decide to rely on an efficient method for displaying data summary, called Box and Whisker plot or simply Boxplot [26].

Boxplots have the power to summarize statistical measures, such as median, upper and lower quartiles, minimum and maximum data values, and confidence intervals in one single representation. Briefly, a Boxplot consists of a rectangle, which encloses the median, with an end at each quartile. The length of the box is also the interquartile range of the sample, i.e., the upper edge indicates the 75th percentile of the data set, and the lower edge indicates the 25th percentile. A line drawn across the box represents the sample median. Additionally, a whisker outside the two ends of the box represents the sample maximum and minimum and the points outside the ends of the whiskers are outliers. One can enhance information presentation of Boxplots using a small variation of the original one, called Notched Boxplots. In Notched Boxplots the sides of the box are notched. Notches are graphical representations of a confidence interval about the median of the data set. Some statisticians emphasize that with skewed data, the notches should be only inferred as a coarse sign of statistical significance. In this thesis, we rely on Boxplots to summarize the efficiency of each algorithm in the DLPF unit and movie type. Using a Boxplot for each implementation of the adaptive stochastic rule and showing side-by-side on the same graphic, we can compare the simulation results confidently.

In order to evaluate our architecture in highly dynamic networks, we used the same procedure that we undertook in chapter 2, thus generating self-similar background traffic with five different shifts in level. Such procedure allowed us to evaluate the benefits of applying DLPF algorithms in comparison with the use of information provided by the underlying transport protocol, in this case, XCP. Therefore, we evaluate the expected quality at the receiver in two approaches: “DLPF” (several algorithms) and “ABR” (no filtering technique applied). Our main objective here is to have a clue whether some chosen smoothing technique are suitable for deployment into the PU unit.

In addition, it is imperative to have a clear picture concerning fairness with competing flows. In Katabi’s thesis [107], she claimed that XCP is significantly fairer than TCP. With 30 long-lived FTP flows sharing a single 30 Mbps bottleneck link, simulation results indicated that XCP provides a fair bandwidth allocation (i.e., with a fairness index [93] close to 1) and does not have bias against long RTT flows.

We define a nomenclature for the DLPF algorithms in order to plenty identify them in graphics. Table 1 presents the label used in plots with the respective DLPF algorithm.

Figure 56 to Figure 60 present an instance of simulation where we can observe five DLPF algorithms' behavior when facing non-stationary background traffic. We will label the original traffic as “ABR” in the following plots, which represents the non-stationary background without smoothing.

Table 4 - Nomenclature for the DLPF algorithms

Label	Algorithm
“kesten”	Kesten
“gaivo”	Gaivoronski
“trigg”	Trigg and Leach
“whybark”	Whybark
“tukey”	Tukey's Smoothing

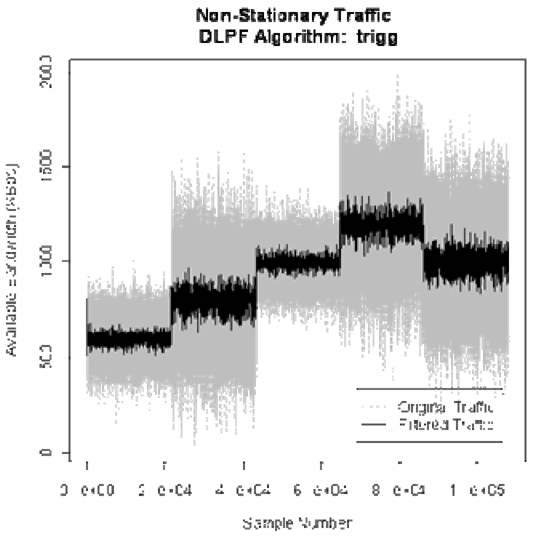

Figure 56 - Smoothed Traffic - DLPF: Trigg & Leach

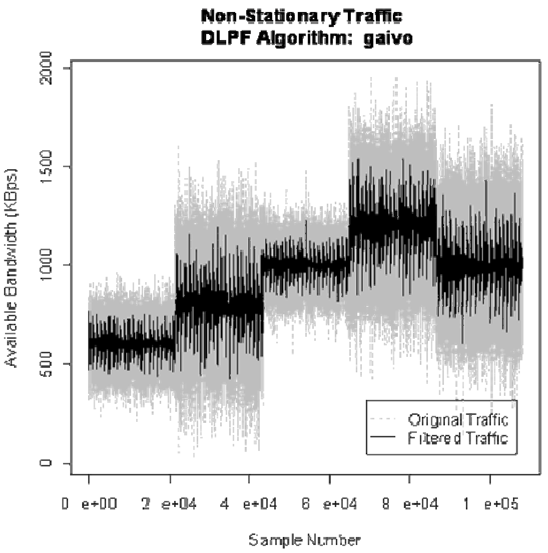

Figure 57- Smoothed Traffic - DLPF: Gaivoronski

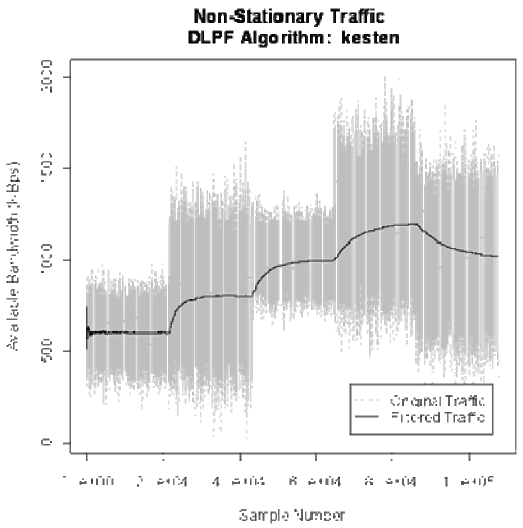

Figure 58- Smoothed Traffic - DLPF: Kesten

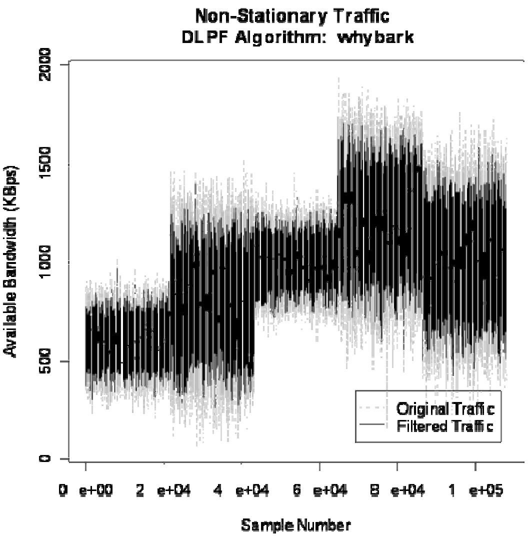

Figure 59- Smoothed Traffic - DLPF: Whybark

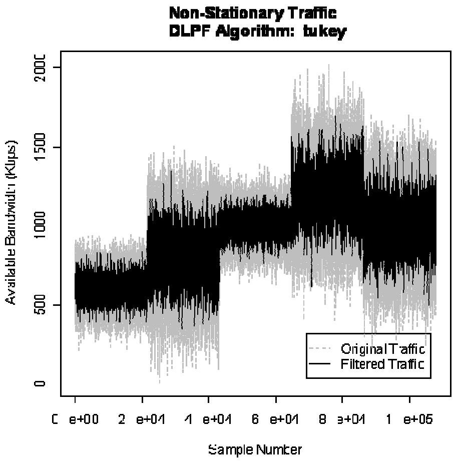

Figure 60- Smoothed Traffic - DLPF: Tukey Smoothing

4.5.2. DLPF: Simulation Results

We based our methodology for presenting simulation results in an explanatory statistical analysis from the data sets. First, we present the received mean quality for the ABR and DLFP for each movie while showing the original BL and EL for comparison. Second, we describe the absolute and relative differences between the BL and EL when compared to the GOP-grouped received frame quality. Finally, we present ECDF and Boxplots for the same data set. It is worth stressing that we will repeat the same procedure for the remainder of DLPF algorithms and movie traces.

4.5.2.1. Movie: The Firm

We now present the simulation results for the movie trace "The Firm" along with the Trigg and Leach implementation at the DLPF. Figure 61 and Figure 62 show the mean quality perceived by the user after video flow has passed through the DLPF unit. We also provide the original quality for the base layer (Figure 61) and enhancement layer (Figure 62) in order to facilitate a fair comparison between DLPF and ABR results.

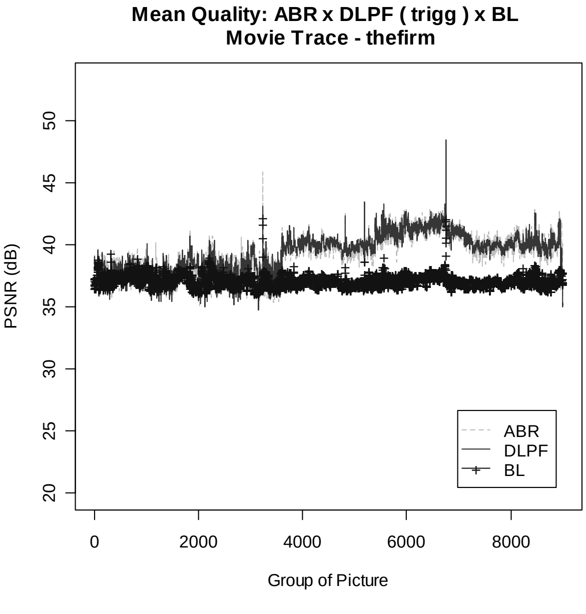

Figure 61 – Comparison of the Mean Quality (BL) : ABR x Trigg – The Firm

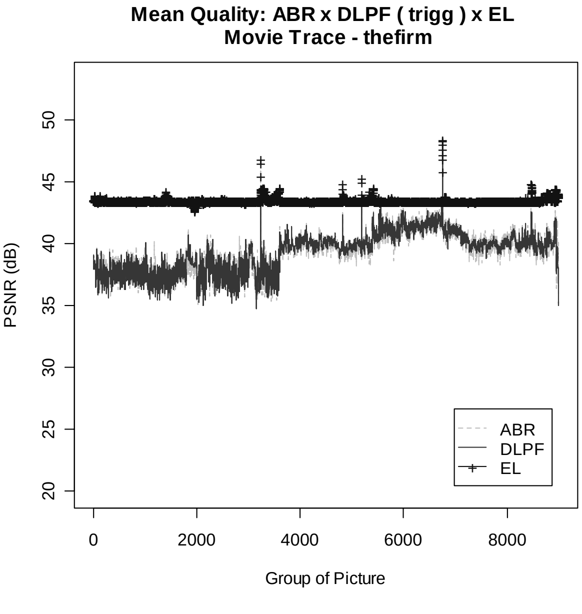

Figure 62 – Comparison of the Mean Quality (EL) : ABR x Trigg – The Firm

We can observe from these results that the general behavior for both ABR and DLPF (Trigg & Leach) is to follow the changes in regime in the available bandwidth. However, from these figures we cannot affirm which approach leads to less quality variation. Therefore, Figure 63 and Figure 64 show the relative differences in the mean quality related to the BL and EL, respectively. These plots can only give us some hints about the variability. It seems that DLPF

(Trigg) provides less variability than ABR. In addition, Figure 65 and Figure 66 present the Empirical Cumulative Distribution Function for the average quality of the ABR and DLPF in all simulations. We included the BL and EL curves for reference. Likewise, the only conclusion that we can draw from these plots is that the mean quality is almost the same for both ABR and DLPF. It also seems that the ABR has longer tail than DLPF, but it this difference is almost indistinguishable.

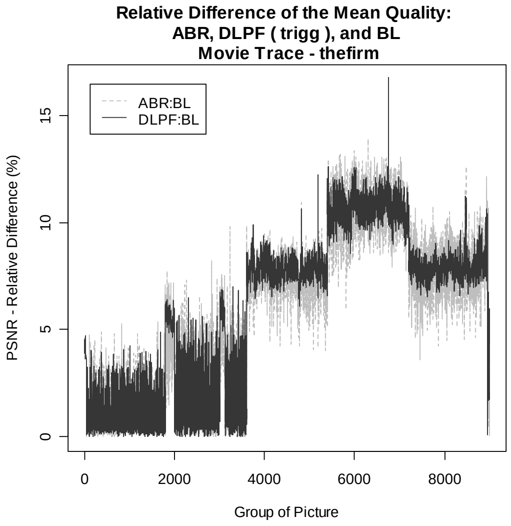

Figure 63 – Relative Difference for the Mean Quality: (ABR x Trigg) to BL

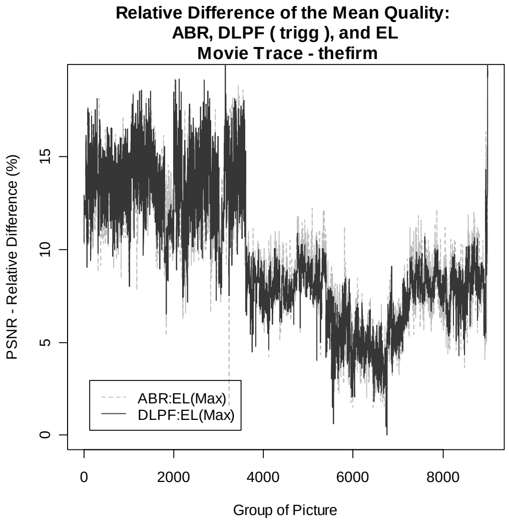

Figure 64 – Relative Difference for the Mean Quality: (ABR x Trigg) to EL

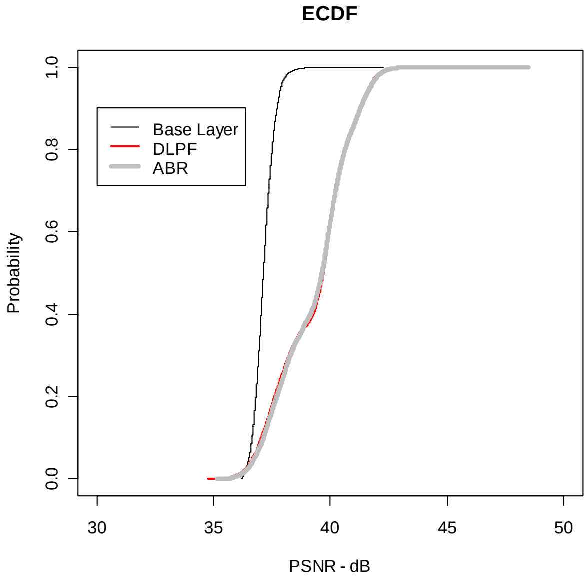

Figure 65 – Empirical Cumulative Distribution Function: DLPF (Trigg) x ABR x BL

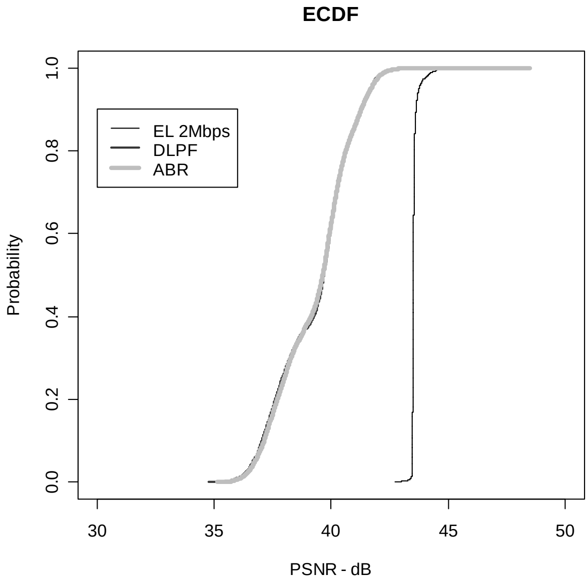

Figure 66 – Empirical Cumulative Distribution Function: DLPF (Trigg) x ABR x EL

From the visual inspection of Figure 61 to Figure 66, it is clear that there exists an intrinsic obstacle in the graphical evaluation of the benefits of the system components in our novel architecture. As we mentioned earlier in this chapter, Boxplots are a reliable graphical methodology to make data analysis more accurate. Figure 67 presents the Boxplot for the mean quality and Figure 68 shows the coefficient of variation for the previously presented movie trace and DLPF algorithm. In Figure 67 we now can see that there is no statistically significant difference between both simulation results, since the notches (i.e., the confidence intervals) overlap. Although there is no statistically significant difference between ABR's and DLPF's mean value, this does not imply that there is no reason to deploy any DLPF technique. Recall that we are looking for less variability while maintaining acceptable quality level. Therefore, in order to verify the resulting quality variability, we should evaluate either the Coefficient of Variation (CoV) or the Standard Deviation (SD). Please note that when the mean values are statistically significant similar, we are allowed to present only the resulting standard deviation. Otherwise, the CoV gives a clear picture of the quality variability for a fair comparison between the ABR and the DLPF approaches. We present the Notched Boxplots for the coefficient of variation in Figure 68. For this scenario, we conclude that Trigg and Leach implementation on the DLPF unit (for the movie trace “The Firm”) provides less variability in video quality with statistically significant difference around 6%. Certainly, it is necessary a further investigation to verify whether other algorithms can provide similar performance levels. Another possibility for investigation is to observe results when applying movie traces with different characteristics.

Considering we made our point with reasonable arguments in favor of Notched Boxplots, from now on we rely strongly on this kind of graphics to verify the effectiveness of either system component or algorithms in our architecture. We advocate that we can make clear and fair comparisons for all simulation results.

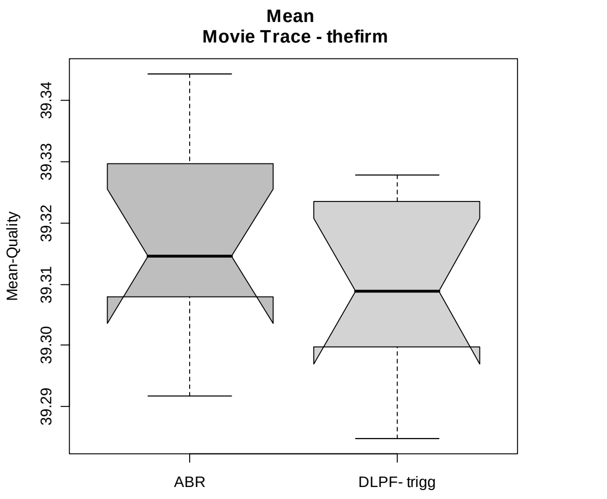

Figure 67 - Notched Boxplots - Mean Quality: ABR x DLPF (Trigg & Leach)

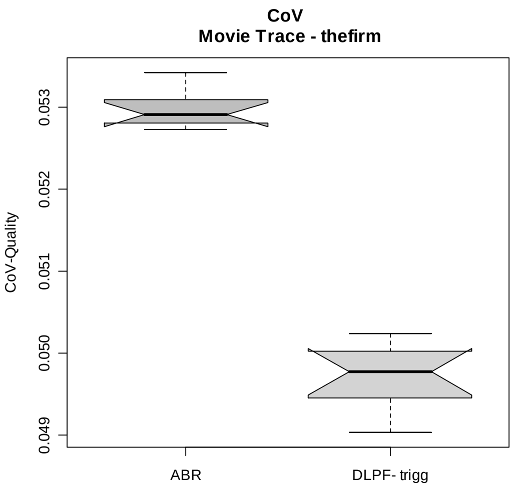

Figure 68 - Notched Boxplots - Coefficient of Variation: ABR x DLPF (Trigg & Leach)

Figure 69 to Figure 78 show the Notched Boxplots for the mean quality and the coefficient of variation of the remainder of the simulation results. Figure 69 presents the mean quality for the Gaivoronski's rule. Although the simulation result shows that the ABR mean is better than that of the DLPF, the difference is irrelevant, since it reaches around 0.1%. However, the coefficient of variation performs equally to the Trigg and Leach technique (Figure 70).

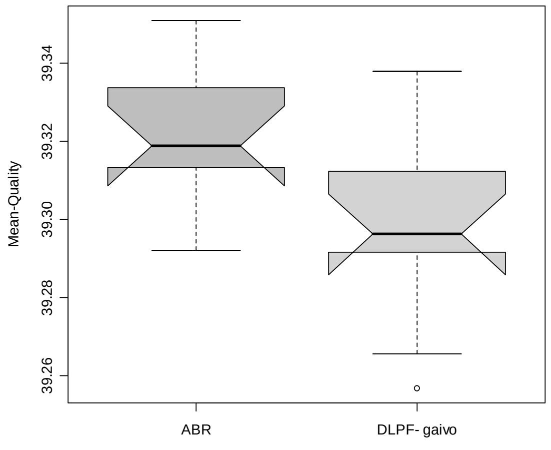

Figure 69 - Notched Boxplots - Mean Quality: ABR x DLPF (Gaivoronski)

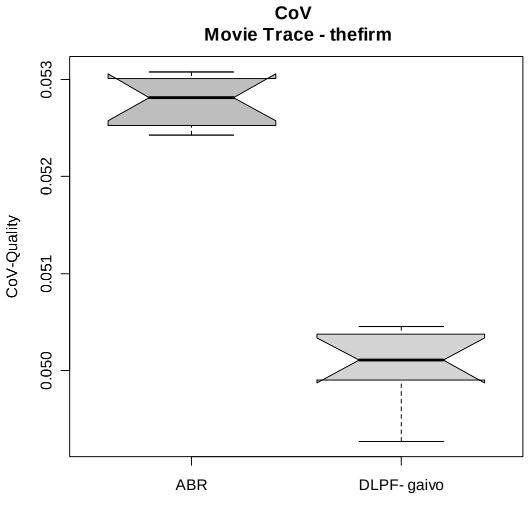

Figure 70- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Gaivoronski)

Figure 71 presents the mean quality for the Kesten's rule. Although the simulation result shows that the ABR mean is better than that of DLPF, the difference is also irrelevant, since it reaches 1%. However, the coefficient of variation performs equally to the ABR (Figure 72). One possible explanation for the poor performance for the Kesten's implementation in the

DLPF is the smoothing profile presented in Figure 33. In such simulation instance, we can observe that when the background traffic shifts its mean level, the algorithm converges slowly to the target level. Supposedly, this behavior produced the undesirable performance in the video stream.

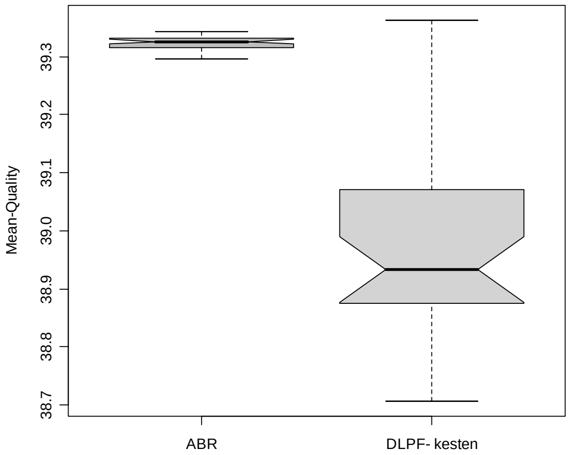

Figure 71 - Notched Boxplots - Mean Quality: ABR x DLPF (Kesten)

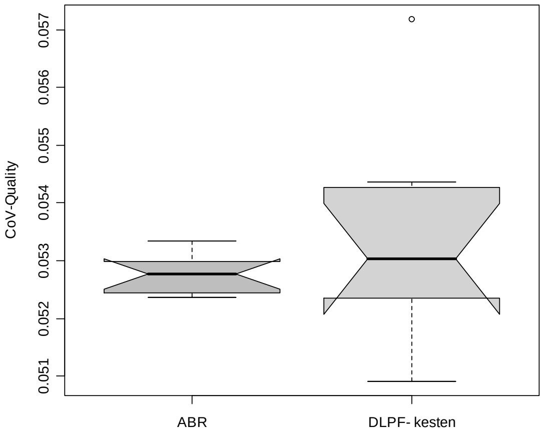

Figure 72- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Kesten)

Figure 73 presents the mean quality for the Whybark's rule where we can observe that there is also no statistically significant difference between both simulation results. In addition, the coefficient of variation performs equally to the Trigg and Leach technique, as shown in Figure 74. We observe similar performance for both metrics mean quality and coefficient of variation for Dennis' and Tukey's rule (Figure 75 to Figure 78). One should observe that some outliers appear during simulation with Dennis' rule in the DLPF.

We can conclude from these simulations that there is an overall improvement for most of the smoothing techniques. Therefore, we should conclude that it is worth deploying the DLPF unit in our architecture, since we can get the similar average quality with less variability.

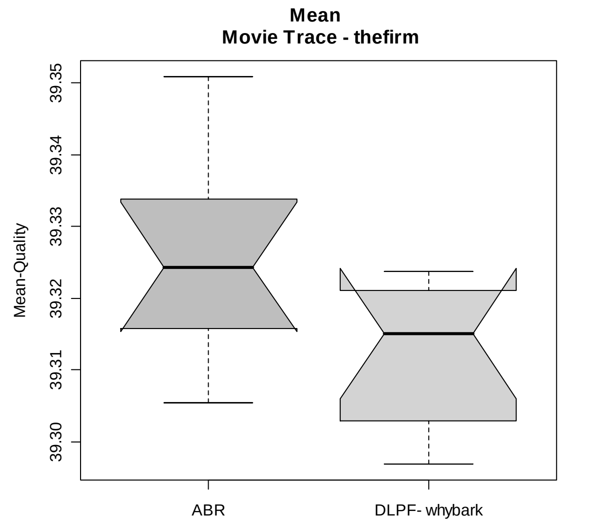

Figure 73 - Notched Boxplots - Mean Quality: ABR x DLPF (Whybark)

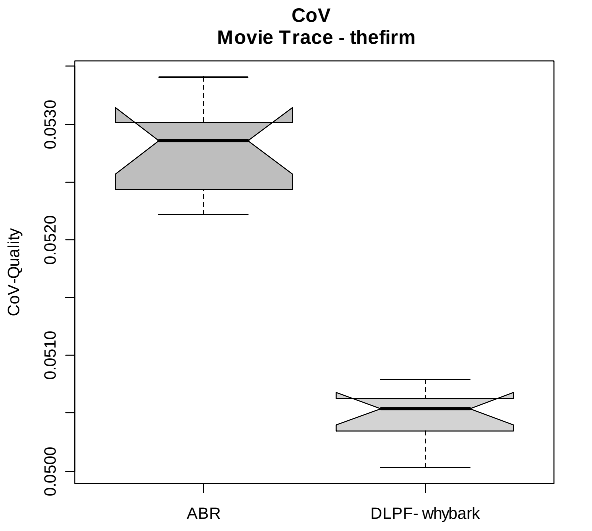

Figure 74- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Whybark)

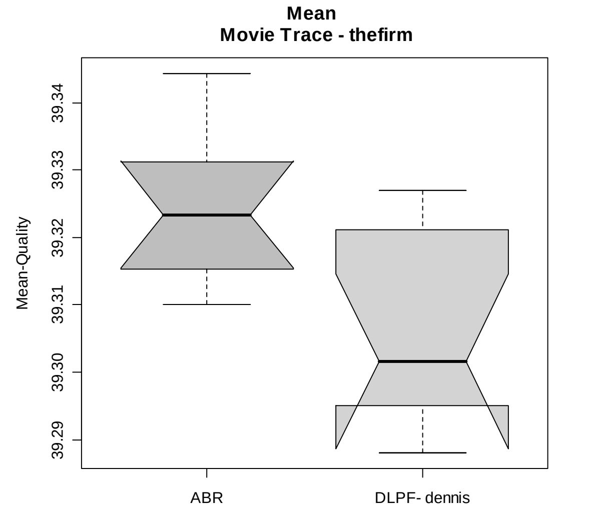

Figure 75 - Notched Boxplots - Mean Quality: ABR x DLPF (Dennis)

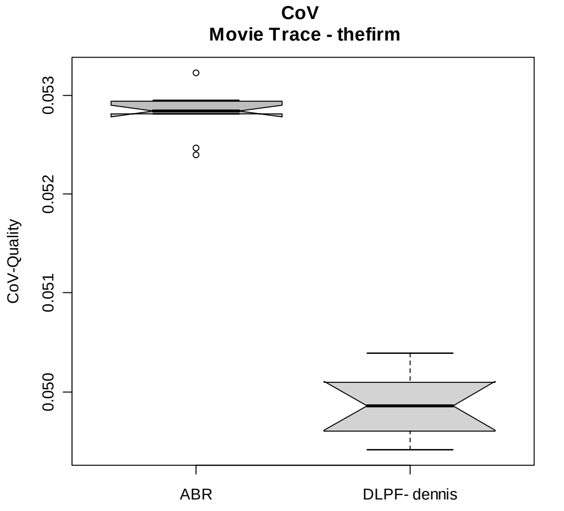

Figure 76- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Dennis)

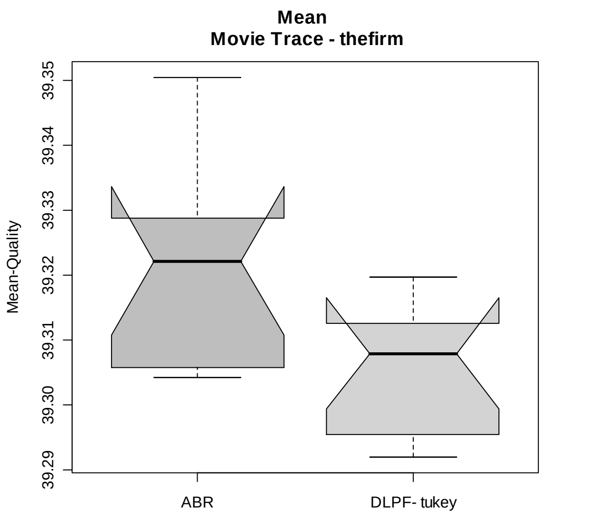

Figure 77 - Notched Boxplots - Mean Quality: ABR x DLPF (Tukey's Smoothing)

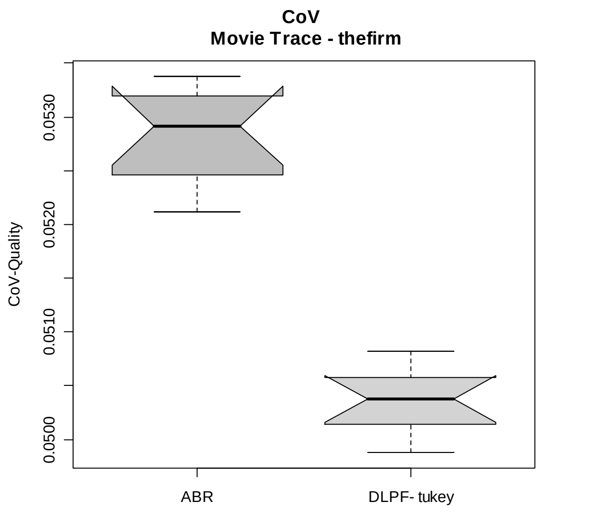

Figure 78- Notched Boxplots - Coefficient of Variation: ABR x DLPF (Tukey's Smoothing)

4.5.2.2. Movie: Star Wars

We now present simulation results for the Movie Star Wars. We present the mean quality and the standard deviation or CoV for the DLPF algorithms. It is worth noting that one could use SD for comparison of quality variability considering that variations equal or below 1dB is barely visible to the user. On the other hand, variations around 2dB or above are noticeable [198] [199].

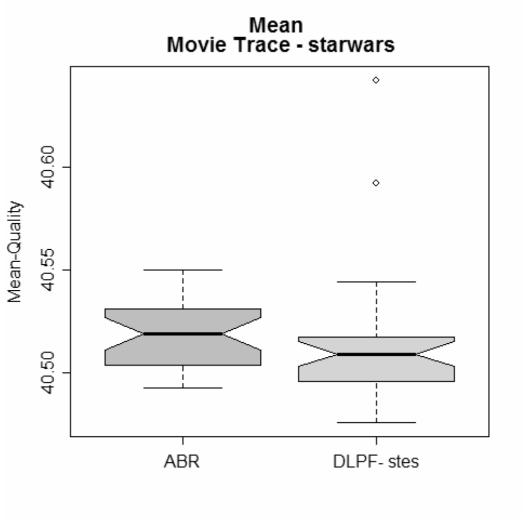

Figure 79 - Notched Boxplots - Mean Quality: ABR x DLPF (STES)

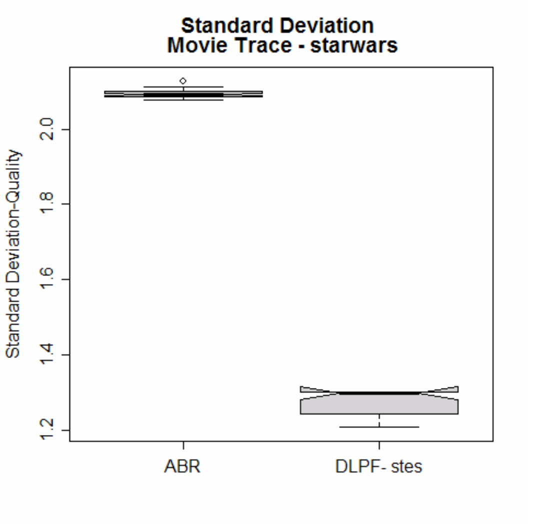

Figure 80 - Notched Boxplots - Standard Deviation: ABR x DLPF (STES)

Figure 79 to Figure 80 show the Notched Boxplots for the mean quality and the standard deviation respectively (STES algorithm). Simulation result shows that the ABR mean is equal to that of the STES. However, the standard deviation performs better with difference around 1dB.

Figure 81 and Figure 82 present the mean quality and the standard deviation for the Dennis's rule. Although we can observe a statistically significant difference between both simulation results, such difference is below 1dB, therefore unnoticeable. On the other hand, we have better values for the standard deviation for the STES algorithm as shown in Figure 82. We observe similar performance for both metrics mean quality and standard deviation for the FIR implementation (Figure 83 to Figure 84).

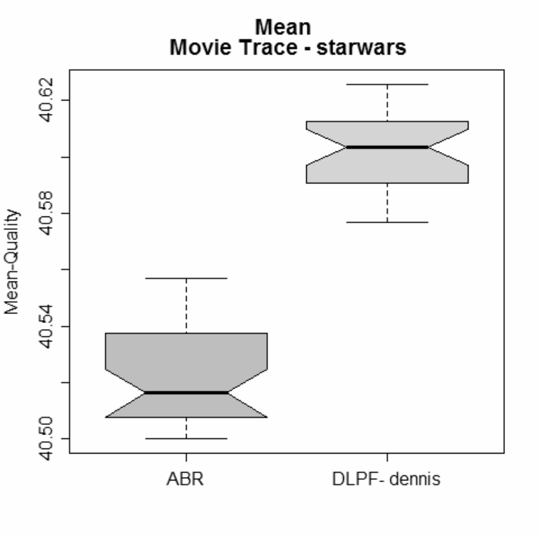

Figure 81- Notched Boxplots - Mean Quality: ABR x DLPF (Dennis)

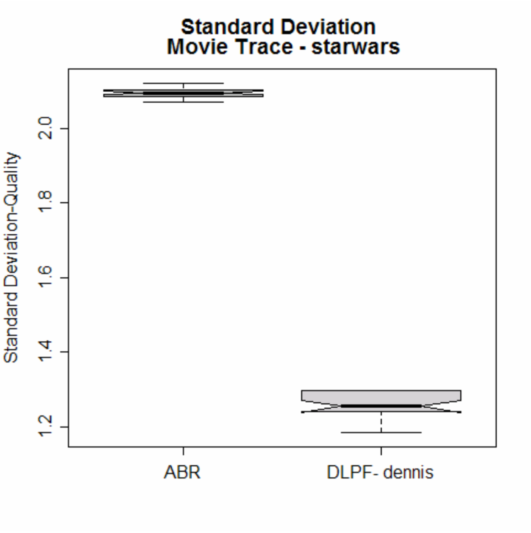

Figure 82- Notched Boxplots - Standard Deviation: ABR x DLPF (Dennis)

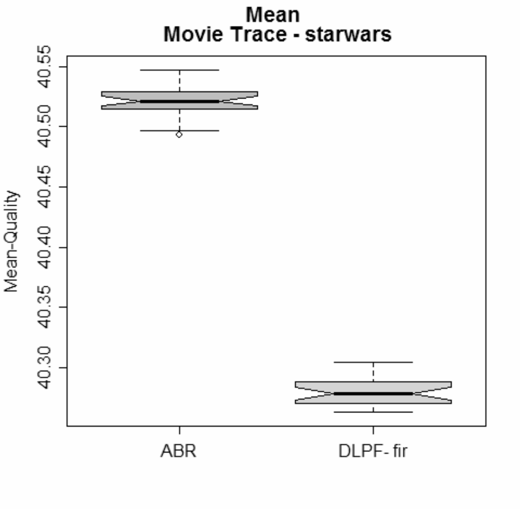

Figure 83- Notched Boxplots - Mean Quality: ABR x DLPF (FIR)

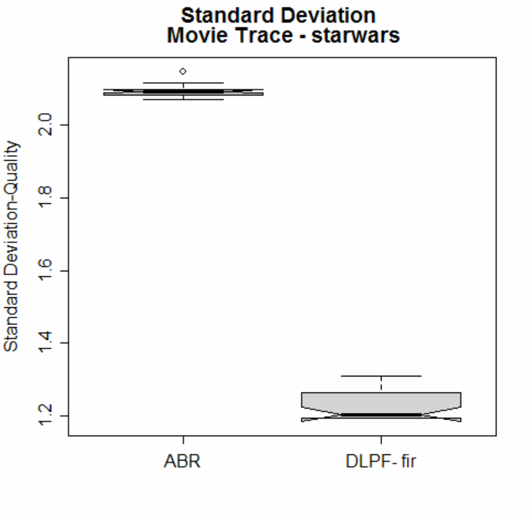

Figure 84- Notched Boxplots – Standard Deviation: ABR x DLPF (FIR)

We can observe in Figure 85 the performance of our architecture with the deployment of the rest of the DLPF algorithms when streaming the movie "Star Wars". From the previous results, we expect that the remaining algorithms perform similarly. By the analysis of the coefficient of variation, it is clear that all DLPF algorithms outperform the ABR approach. The difference in the CoV is roughly 15% for Kesten, 10% for Gaivoronski and Trigg & Leach, and 8% for Whybark and Tukey rules.

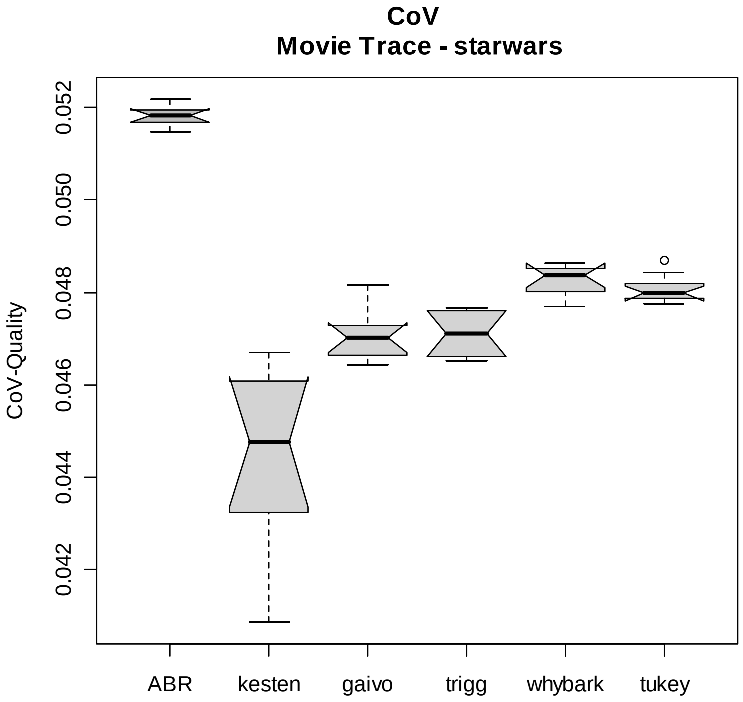

Figure 85 - Performance Comparison of DLPF algorithms - Star Wars

4.5.2.3. Movie: Oprah

We now present simulation results for the movie *Oprah*. As the mean quality for all algorithms is statistically equivalent, we present either the standard deviation or the CoV for the DLPF algorithms.

Figure 86 to Figure 88 present the standard deviation for the STES, Dennis and FIR implementation. We can observe a statistically significant difference between ABR and the correspondent DLPF algorithms. For ABR simulation results, the standard deviation is above 1dB, but still below 2dB. We suppose that the type of video (i.e., talk show) has a strong influence in such result. The main characteristics of talk show videos is low variability in frame sizes, mainly in I frames. This is because there are few changes in most scenes, where the scenario is a motionless background with two people talking.

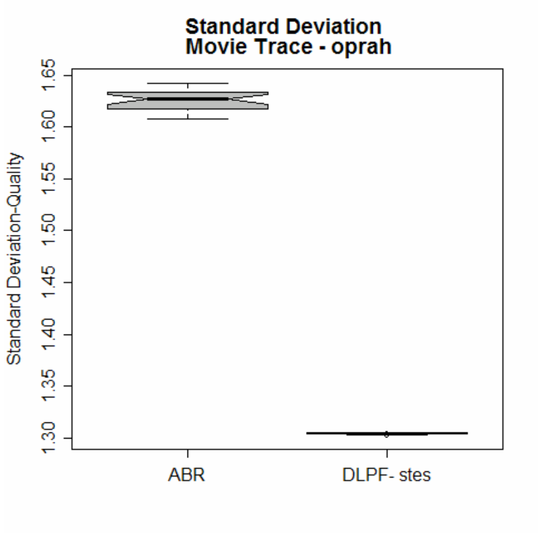

Figure 86- Notched Boxplots – Standard Deviation: ABR x DLPF (STES)

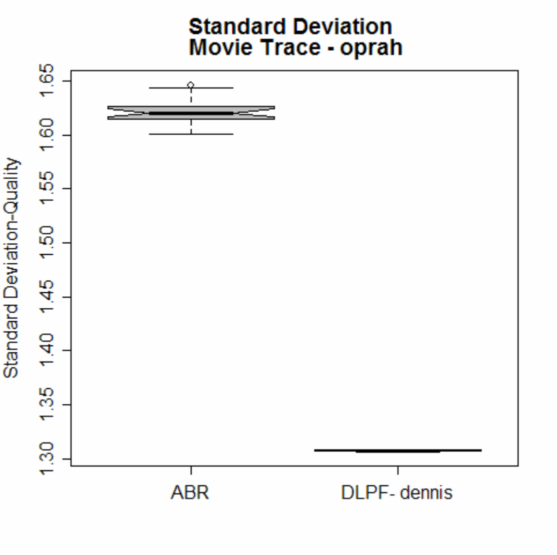

Figure 87- Notched Boxplots – Standard Deviation: ABR x DLPF (Dennis)

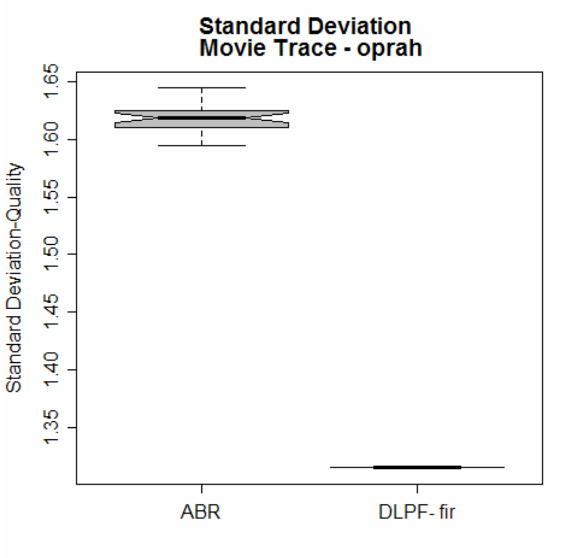

Figure 88- Notched Boxplots – Standard Deviation: ABR x DLPF (FIR)

Figure 89 presents the simulation results of the rest of the DLPF algorithms when streaming the movie trace “Oprah”. By the analysis of the coefficient of variation, it is clear that most DLPF algorithms outperform the ABR approach. In fact, only the Kesten rule achieved the same CoV level as the ABR approach. For the rest of algorithms, the difference in the CoV is roughly 12% (Gaivoronski, Trigg & Leach, Whybark, and Tukey). We should emphasize that Kesten’s rule appears to have great instability, since with some movie traces characteristics it achieves the best performance, whereas with others it achieves the worst one.

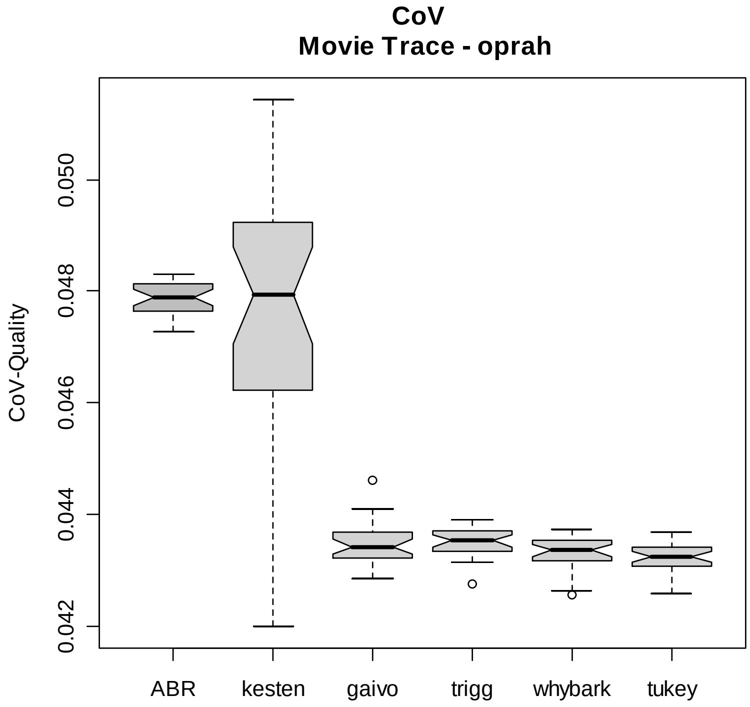

Figure 89- Performance Comparison of DLPF algorithms - Oprah

4.5.2.4. Movie: Toy Story

We now present simulation results for the movie *Toy Story*. Similarly to the previous evaluation of the movie Oprah, as the mean quality for all algorithms is statistically equivalent, we just present either the standard deviation or the CoV for the DLPF algorithms.

Figure 90 to Figure 92 present the standard deviation for the STES, Dennis and FIR implementation. We can observe a statistically significant difference between ABR and the correspondent DLPF algorithms. For all ABR simulation results, the standard deviation is above 2dB. We expected this result since the type of video (i.e., cartoon) has a sharp histogram for the frame sizes, which strong influence the sending rate requirement. The main characteristic of cartoons is medium to high variability in frame sizes.

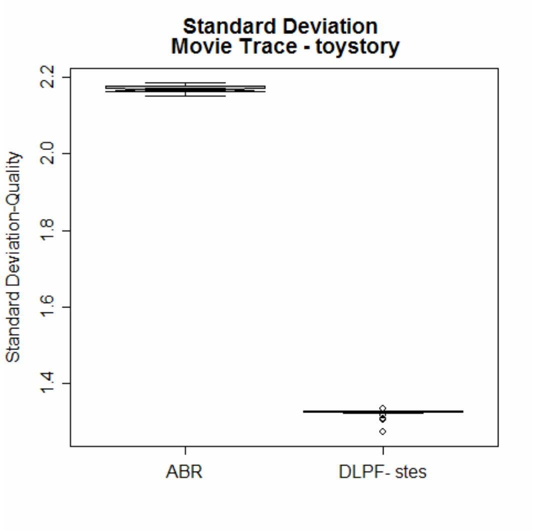

Figure 90 - Notched Boxplots – Standard Deviation: ABR x DLPF (STES)

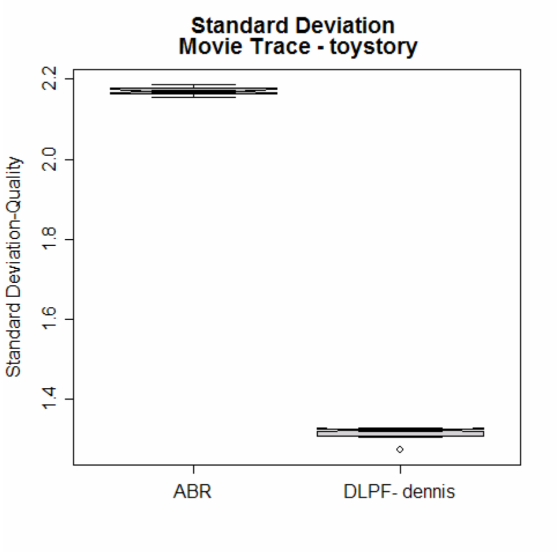

Figure 91 - Notched Boxplots – Standard Deviation: ABR x DLPF (Dennis)

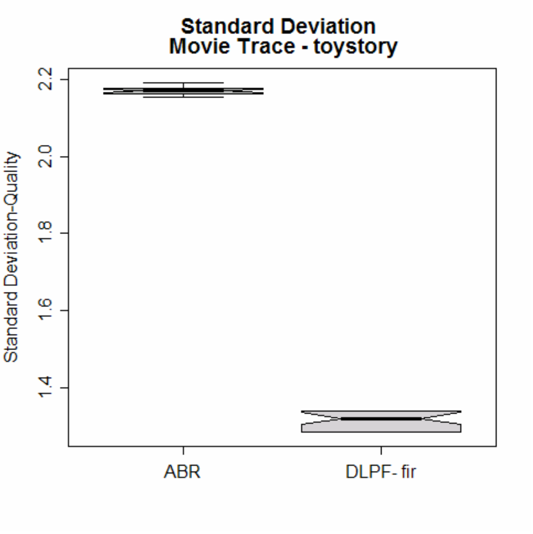

Figure 92 - Notched Boxplots – Standard Deviation: ABR x DLPF (FIR)

Finally, we observe in Figure 93 the performance of our architecture with the deployment of some DLPF algorithms when streaming the cartoon-type movie, "Toy Story". By the analysis of the coefficient of variation, it is also clear that all DLPF algorithms outperform the ABR approach. The difference in the CoV is roughly in the range of 7%-8% for Kesten, Gaivoronski, Trigg & Leach, Whybark and Tukey rules.

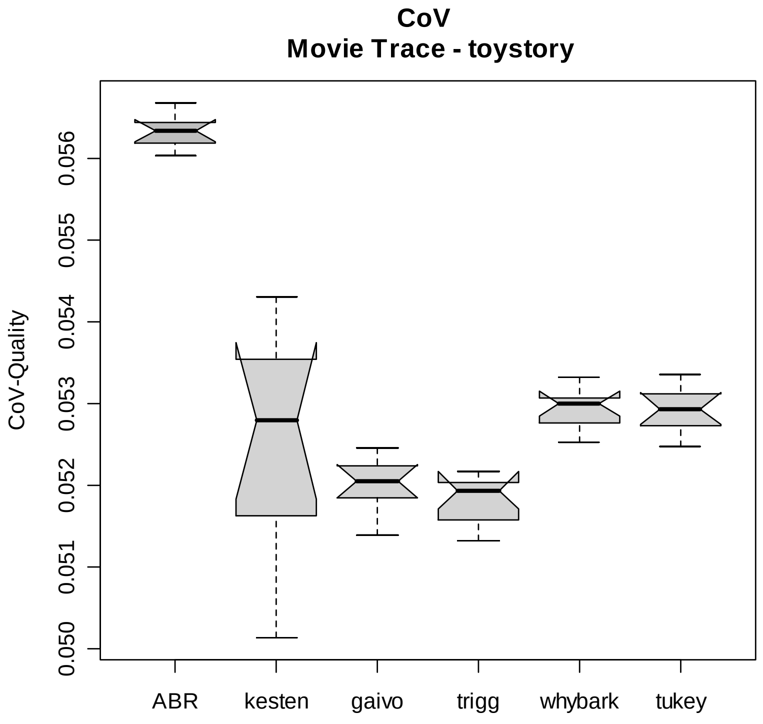

Figure 93- Performance Comparison of DLPF algorithms - Toy Story

Chapter 5. Forecast and Control of Scalable Video Streaming

In the previous chapter, we proposed the use of adaptive low-pass filter for smoothing the raw traffic information flowing from the transport levels. We clearly observed that the low-pass filter functional block provided better results when compared with the direct use of such information. We suggested that in cases when the RTT is large and rate variations are fast, the performance of applications in explicit feedback networks is still dependent on the variations of the background traffic. In such conditions, smoothing the stochastic behavior of the network information throughout time emerged to be a better approach.

In this chapter, we take a further step on the analysis and solution to the problem of decision control for streaming fraction of stored scalable video. To do that, we introduce a methodology to identify how and when the Decision Unit (DU) must take decisions with respect to which portion of the MPEG-4 FGS stored video the server should stream into the IP network. Using the available bandwidth information samples coming from the DLPF, we are able to perform one-step ahead forecasting or to analyze such signal in the Wavelet domain, in order to extract the essential signal energy as well as noise energy. We found this necessary since the traffic profile from the DLPF could still exhibit variability at multiple time scales.

One stage of our methodology relies on the Wavelet Multiresolution Analysis (MRA). By using Wavelet MRA, we are able to identify the overall long-term (mean) available bandwidth along with the variability (noise) at multiple time scales. A second stage (concurrent) is modeling the main signal with a parsimonious linear time series model. This model will eventually be able to forecast changes in the main signal. The Decision Unit may use the Mean Percentage Error (MPE) in this prediction or the noise energy information heuristically. By applying a multiresolution analysis and linear forecasting, we are sure that the additional functional blocks in our novel architecture will reduce the amount of variability in the expected video quality significantly.

The remainder of this chapter is set out as follows. In Section 5.1, we present motivation for using Multiresolution analysis and some promising studies on the analysis of network traffic in the Wavelet domain. Section 5.2 provides definitions, notation and an overview of the Wavelet MRA and Time Series Analysis (TSA) with linear models, since these concepts are necessary for the clear understanding of the methodology used throughout this chapter. We describe in Section 5.3, some important design decisions for the proposal of our architecture. We give details about the Prediction Unit in Section 5.4, and about the Decision unit in Section 5.5. We will carry out an extensive simulation-based performance analysis in Section 5.6. Finally, to conclude this chapter, we discuss our results in Section 5.7.

5.1. Signal Processing Techniques for Network Traffic Analysis

The use of Wavelet transforms for modeling and analysis of network traffic has received a lot of attention recently [1] [2] [43] [87] [95] [96] [117] [136] [147] [148] [150] [155] [161] [182] [188] [213] [231]. The networking and Internet research community have applied Wavelets transforms in a variety of ways. For instance, the usefulness of the Wavelets transforms has proved to yield reliable results when evaluating burstiness at multiple time scales for the network traffic [136] [188] [213], modeling heterogeneous traffic [87][161] or video VBR traffic profiles [150], investigating (multi)fractality behavior [43] [95] [96] [155] [231], and even detecting congestion [1] and traffic abnormalities [2][117].

In [136], Jiang and Dovrolis used Wavelet MRA to analyze the burstiness of a traffic process in a range of time scales. As such, they relied on Wavelet-based energy plots, which have the property to reveal the variance of the wavelet coefficients of a traffic process as a function of the scale index *j*. One should interpret such variance as the variance of the traffic variation. By applying the Wavelet MRA approach, they were able to identify the actual mechanisms responsible for creating burstiness in network traffic in time scales on the order of 100-1000 milliseconds. Using a similar technique as in [136], i.e. relying on energy plots, Huang et al. [117] developed a tool for Wavelet-based inference for detecting network performance problems, whereas Li and Lee [136] utilized energy distribution based on wavelet analysis to detect DDoS attack traffic. For anomaly detection issues, Jiang and Papavassiliou [97] came up with the idea of traffic separation based on a frequency domain analysis and filtering. Specifically, they separated the network traffic into two main components, namely the baseline component and the short-term component. The former includes the low frequency and presents low intensity burstiness, which may present non-stationarity features (e.g. first-order non-stationarity). The latter includes the highly dynamic component. The authors use an Autoregressive Moving Average (ARMA) model for forecasting the short-term behavior. Their results show that the proposed scheme of separating the traffic on the frequency domain and forecasting each component separately, improves the prediction accuracy of the aggregated network traffic. Although we rely on methodologies in the Wavelet domain, it seems that both approaches eventually will have similar results. We just stress that we are not interested in forecasting the exact network available bandwidth level. As our architecture concerns the minimization of the rendered video quality, one important feature is the short-term variability level. We can clearly see that the Wavelet MRA will provide us with the adequate set of information, rather than those provided by the frequency domain analysis.

In the area of congestion management, Kim et al. [1] proposed a Wavelet -based novel technique for detecting shared congestion of paths with or without a common endpoint. On the other hand, for modeling issues, Ma and Ji [161] applied a wavelet approach for modeling heterogeneous traffic joint with an in-depth investigation of the performance of the obtained wavelet models. Finally, Papagiannaki et al. [87] [164] [165]combined the Wavelet MRA approach along with other statistical techniques (e.g., ANOVA and ARIMA models) in order to forecast upgrades in an IP backbone network.

Wavelet transforms provide insights on the scale-dependent properties of data through the coefficients of the joint scale-time decomposition. Such characteristic implies that almost none *a priori* information about the process is necessary. In addition, Wavelets transforms work well with non-stationary processes, i.e. it is robust intrinsically. Such characteristics fit in our application domain, since our architecture will eventually face non-stationarity such as for available bandwidth data. It also has the desired feature of parsimoniousness.

5.2. Techniques for Multiscale Analysis and Forecasting

In this Section, we provide the necessary background for a complete understanding of our approach. We first describe how to carry out Wavelet MRA followed by a brief description of precise forecasting with linear time series modeling and analysis.

5.2.1. Wavelet Multiresolution Analysis

Wavelet Multiresolution Analysis (MRA) relies on the decomposition of a signal using an orthogonal set of basis functions, which includes a high-pass Wavelet function and a low-pass scaling filter. In other words, MRA synthesizes a discrete time series (signal) from a very low-resolution signal at large time-scales to higher resolution versions at small time-scales. It is worth emphasizing that Wavelets are able to model univariate and multivariate signals. The output of the high-pass filter is a new signal (series) joint with the detailed coefficients of the function. On the other hand, the output of the low-pass filter is the approximation coefficients of the original signal. An important feature of the orthogonal set of basis functions is that few parameters are required to represent any signal, but this does not affect accuracy whatsoever. In fact, by choosing an adequate set of basis functions one can obtain a high level of precision.

We now provide a formal description of MRA. It starts with the decomposition of the main signal into an approximation signal along with some detailed ones. Let *2^j* be the time scale or resolution at level *j*, which specifies the depth of the decomposition.. Let also $\psi(t)$ be the analyzing (mother) wavelet function and $\phi(t)$ be the mother scaling function. By dilating and scaling both mother functions one obtains a set of scaling and wavelet functions, which will be used to represent the original signal at the Wavelet domain. In equation terms:

$$\psi_{j,k}(t) = 2^{-j/2}\psi\left(2^{-j}t - k\right)$$

$$\phi_{j,k}(t) = 2^{-j/2}\phi\left(2^{-j}t - k\right)$$

where the variables *j* and *k* are integers that scale and dilate the mother functions $\psi(t)$ and $\phi(t)$. $\psi_{j,k}(t)$ and $\phi_{j,k}(t)$ are then the set of basis functions.

Let $x(t)$ be a random process generated from an independent wavelet model for discrete time *t* ($t \geq 0$). Hence, by applying MRA, the original signal is represented as follows:

$$x(t) = \sum_k c_{p,k}\phi_{p,k}(t) + \sum_{0\leq j\leq p}\sum_k d_{j,k}\psi_{j,k}(t)$$

where $d_{j,k}$ are the wavelet coefficients for the detailed series and $c_{j,k}$ are the scaling coefficients for the approximation series at the time scale *j* ($1 \geq j \geq \log(N)$) and shift k ($k \geq 0$), where N is the number of samples at the original discrete signal. The wavelet coefficients must satisfy some constraints, as follows:

$$\sum_{k=0}^{N-1} d_k = 2$$

and

$$\sum_{k=0}^{N-1} d_k d_{k+2l} = 2\delta_{l,0}$$

The combination of the low-pass and high-pass filters is often seen as a filter bank [70] [203]. One can observe that at scale *j*, the approximation series $\{c_{j,k}\}$ pass through the low-pass filter and the high-pass filter to generate the approximation $\{c_{j+1,k}\}$ and the detail $d_{j+1,k}$ at scale $j+1$. At each stage, the number of coefficients at scale $j+1$ is decimated into half of that at scale j (downsampling). This decimation reduces the series length at coarser time scales and removes the redundant information in the wavelet and scaling coefficients [70] . Our first impression is that downsampling has an adverse side effect that is to prevent direct access to the whole series at a given time scale, since one must reconstruct them from the coefficients at that scale. As far as we are concerned, it is not clear if such removal of redundant information will result in better performance of the prediction algorithms. In fact, as a rule of thumb, linear time series models deal adequately with redundant information. Therefore, these redundant coefficients are probably useful for improving forecasting accuracy.

In order to mitigate this downsampling side effect, we will use an efficient algorithm for Discrete Wavelet Transforms (DWT), namely the “à trous” wavelet transform. The “à trous” algorithm eliminates the decimation effect and generates integral approximations and detailed series. It produces better approximations by filling the holes (trous in French) caused by downsampling, using redundant information from the original signal [70] [203].

Let $h(l)$ be a low-pass filter with compact support, which in turn means that a generic function is non-zero only over a finite interval. Using the “à trous” wavelet transform, the approximations at different scales are:

$$c_o(t) = x(t) \text{ and}$$

$$c_j(t) = \sum_{l=-\infty}^{\infty} h(l) c_{j-1}\left(t + 2^{j-1} l\right)$$

where $1 \geq j \geq \log(N)$.

Following the computation for the approximation series, the detailed series (wavelet coefficients) at scale j for the original signal are the result of the difference of the successive smoothed version of that signal. In mathematical terms:

$$d_j(t) = c_{j-1}(t) - c_j(t)$$

The vector $d_j = \{d_1, d_2, \ldots, d_j\} = \{d_j(t),\ 1 \leq t \leq N\}$ represents the wavelet coefficients at scale j, whereas $c_j = \{c_j(t),\ 1 \leq t \leq N\}$ denotes the signal residual. Hence, $\{d_1, d_2, \ldots, d_j, c_j\}$ is the “à trous” wavelet transform of the original signal up to resolution level j. The inverse procedure to obtain the original signal is a backward reconstruction as a linear combination of the non-decimated wavelet and scaling coefficients, as follows:

$$x(t) = c_p(t) + \sum_{j=1}^{p} d_j(t)$$

In order to complete the process, one should choose an adequate filter for one's application domain. There are a number of low-pass wavelet filters $h(l)$. For instance, the Haar "à trous" wavelet transform uses a Haar filter. With such filter, the scaling coefficients at higher scales are obtained from the scaling coefficients at lower scales effortlessly. We can also choose one from the Daubechies' family, B3 splines, etc. Figure 94 shows an example of Haar and Daubechies orthonormal compactly supported wavelet (extremal phase family).

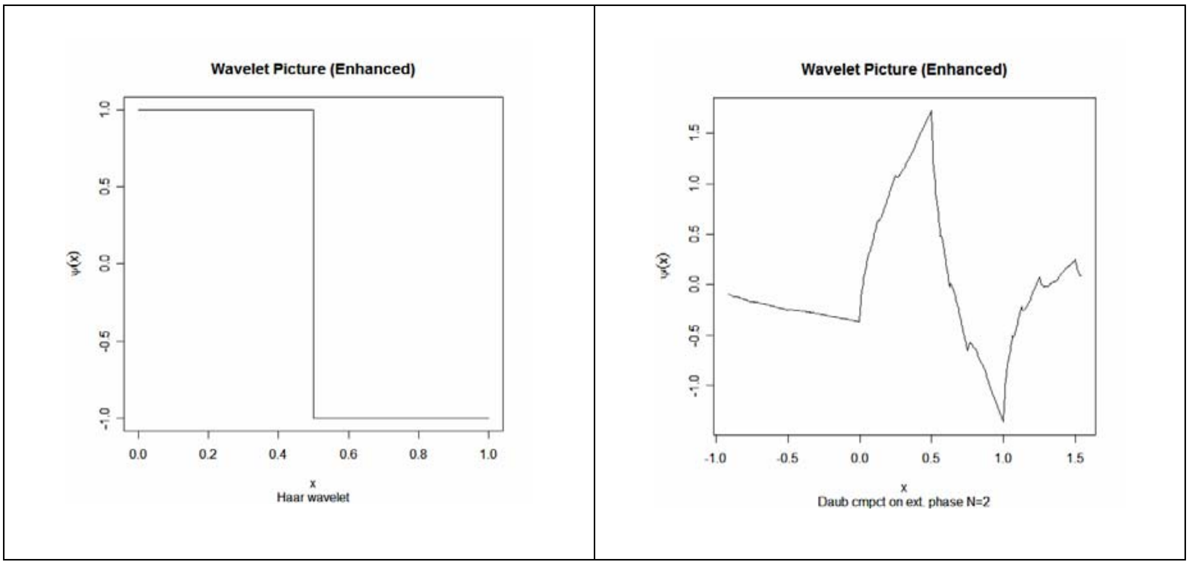

Figure 94 - Wavelet Filters. (a) Haar Filter (b) Daubechies

5.2.1.1. Preliminary Wavelet MRA Simulations

Although there are strong evidences that Wavelet MRA is useful for a number of applications, we still face the problem of how we will apply such decomposition in our problem domain. For this reason, we decide to investigate, in a simulation-based approach, the best strategy for utilizing Wavelet MRA in the PU functional block.

First, we generate four synthetic traces that simulate the available bandwidth information flowing from the previous functional block of our architecture, the DLPF. Figure 95 presents the four synthetic traces, which we identified them as "High Variability", "Medium Variability", and "Low Variability". We generated the synthetic traces by controlling the shape parameter of a heavy tail probability distribution function, namely the Weibull. We kept the scale parameter fixed in all parameterization for a fair comparison among them, since we are only interested in the variability information.

As we are only interested in the identification of statistical variability of each trace in the Wavelet domain, we should define an appropriate metric in order to undertake fair comparisons. Recall that the decomposition of the synthetic trace by using Wavelet MRA results in several approximation series jointly with the detailed ones. Since the detailed series contain information about variability in several time scales, we will rely on them for our performance evaluation.

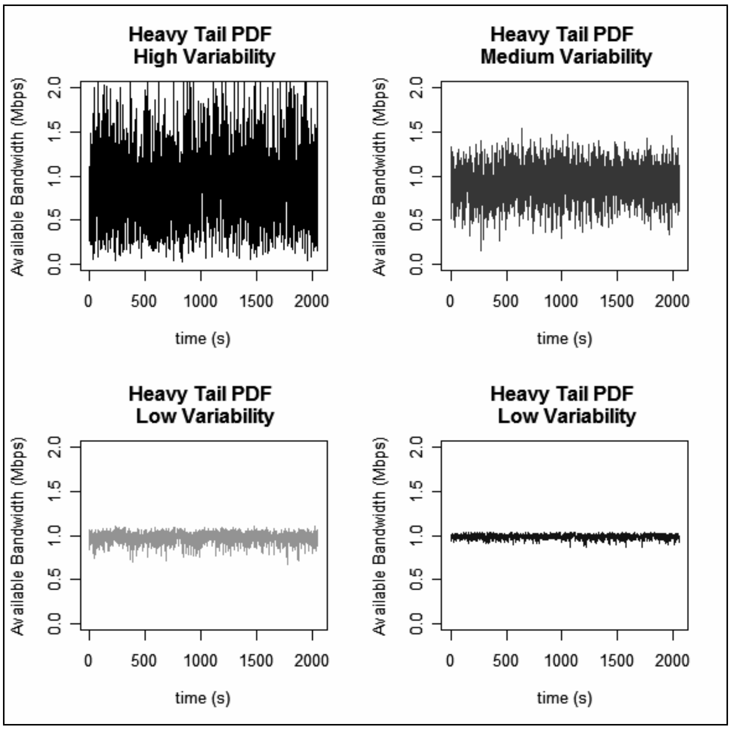

Figure 95 - Simulated Available Bandwidth from Heavy Tail distributions

We mentioned that Jiang and Dovrolis [95] [96] relied on energy plots in order to evaluate burstiness (i.e., statistical variability) of the traffic process at several time scales. In a similar approach to [95] [96], Magnaghi et al. [150] proposed a wavelet-based energy analysis to identify traffic anomalies effectively due to network misconfiguration whereas Lee and Li [136] utilized energy distribution based on wavelet analysis to detect traffic profile from a DDoS attack. Thus, energy distribution plots seem to be a wise choice for traffic variability detection in our architecture at the time scale of interest.

It is clear that high variability implied in a more fluctuating traffic load, as we observed in our synthetic traces. Since we are only interested in variability in a short range of time scale (e.g., from RTT to GOP time scales) there is no need to gather information in a wide range of scales. However, we believe it is necessary to look closer at the general profile of the energy distribution over a wide range of scales, just to make sure that most energy in variability is within our time scale of interest. Even though there is dominant variability in time scales beyond that of our interest in a real network, we can focus our analysis only on a short-range one without loss of generality.

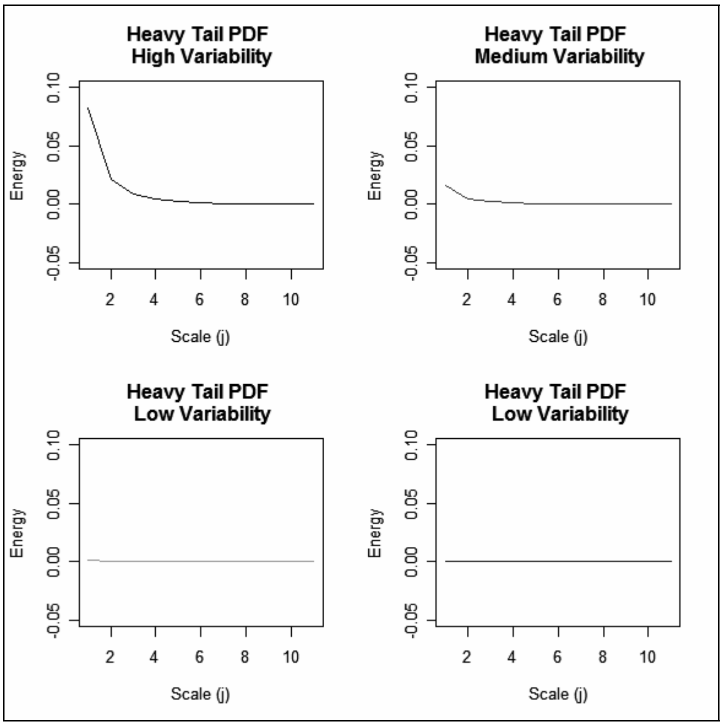

Figure 96 - Energy Distribution over Different Scales – Definition #1

The Wavelet MRA energy plot description presented in [136] refers to a plot that shows the variance of the wavelet coefficients of the traffic process as a function of the scale *j*. In mathematical terms, the energy at scale *j* is $\varepsilon_j = \text{var}\left\{ d_{j,k} \right\}$. Figure 96 shows the energy distribution for the four synthetic traces we generated. Since we are using the same y-scale in all plots, we can clearly see that the signal energy is representative for the traces with high variability in the first and second scale index *j*. For the traces with low variability, the energy level for all scales is very low.

An alternative definition for energy at a given scale is that presented in [87] [95] [96], where the energy at scale *j* is given by $\varepsilon_j = \sum_{k=1}^{N} \left\{ c_{j,k} + d_{j,k} \right\}$. Figure 97 shows the energy distribution for the four synthetic traces we generated according to this new formula. Please note that we did not apply the same y-scale for all plots although we can clearly see that the signal energy has the same decreasing profile for all plots. However, for the traces with high variability, the energy level is significant for the first time scale until the third one. For the traces with low variability, the energy value for all scales is very low.

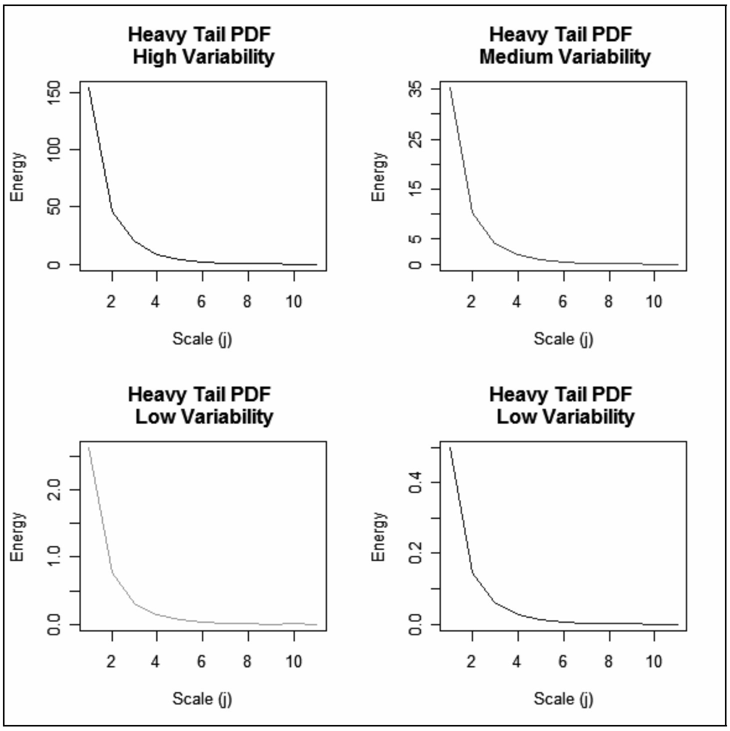

Figure 97 - Energy Distribution over Different Scales – Definition #2

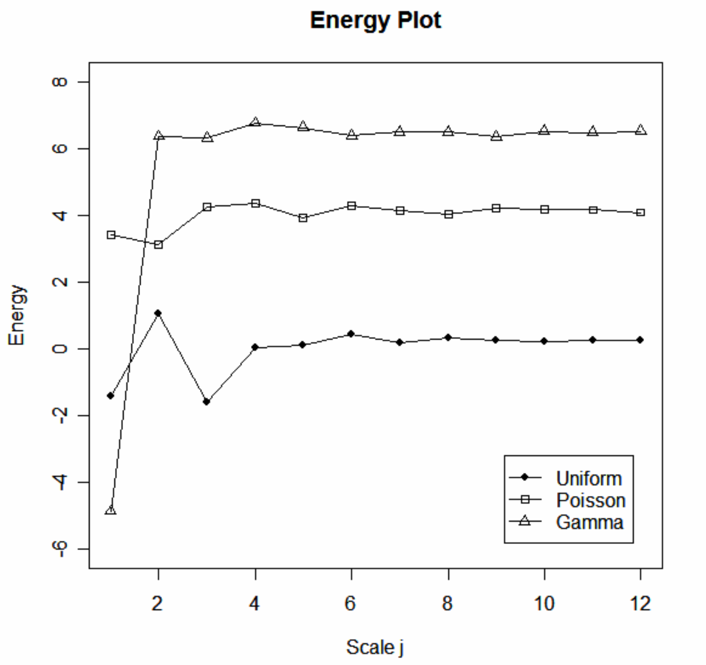

Figure 98 - Noise level for Uniform, Poisson and Gamma PDF at several scales - Log2 (Energy)

In [95] [96], Jiang and Dovrolis describe a technique to analyze the statistical variability of the traffic process in a range of time scales. In essence, they rely on MRA energy plots that show the variance of the wavelet coefficients (i.e., its energy) of the traffic process as a function of the scale. For instance, the energy signature of a Uniform process is a horizontal line around zero whereas the energy profile for a Poisson process is also a horizontal line but with a higher level [95] [96]. Based on this property, they show that a traffic process is bursty at a given scale if the energy profile is higher than the energy of a Poisson process with the same average rate. On the other hand, the traffic process is smooth at the same scale.

We carried out a simulation for some traffic process that follows a Uniform, Poisson and Gamma PDF. We present their energy profile where the y-axis is the base-2 logarithm of the resulting energy as a function of the scale *j*. We will utilize such property when defining a heuristic for the decision unit (Section 5.5).

5.2.2. Linear Time Series Modeling and Analysis

A time series is a set of observations generated in a successive way. Time series may have features such as trends and seasonality as well as short or long-range dependence. One goal of time series modeling and analysis is to forecast future values [80]. Such objective requires that the time series present some kind of regularity in its behavior throughout time. In the case of non-stationary time series, one should apply a transform in order to make them stationary. For example, one can take differences, logarithms or squared roots of the observations, as in the class of procedures called the Box-Cox transformation [76]. There are some classical approaches for modeling stationary time series, such as the linear models Autoregressive (AR), the Moving Average (MA) and the Autoregressive Moving Average (ARMA).

Let y_t be a stationary time series with no seasonal cycles. If y_t follows an *ARMA* process of order *(p, q)*, denoted by $y_t \sim ARMA(p,q)$, then its basic model is:

$$y_t = \mu + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \ldots + \theta_q \varepsilon_{t-q}$$

where $t \in [1,T]$, μ is its the mean level, $E[\varepsilon_t] = 0$ and $E[\varepsilon_t^2] = \sigma_\varepsilon^2$ (homoscedasticity).

A general approach for modeling non-stationary time series is the Fractional Autoregressive Integrated Moving Average model, *FARIMA(p, d, q)*, that is an ARMA model with fractional differentiation. It has the following compact formulation:

$$\Phi(L)(1-L)^d(y_t-\mu)=\Theta(L)\varepsilon_t$$

Taking a close look at the previous equation, one should observe that it is necessary to find a way to estimate the values of the set of parameters $\theta \equiv (\mu,\phi_1,\phi_2,\ldots,\phi_p,\cdots,\theta_1,\theta_2,\theta_q)$, known as the vector of population parameters, based on observations of y_t. As such, a usual inference technique is the Maximum Likelihood Estimation (MLE). First, we need to calculate the likelihood function (LF), $L(\theta;y)$. Hence, the maximum likelihood estimate of θ is the value for which this sample is most likely to have been observed, that is $\theta = \arg\max L(\theta;y)$, $\theta \in \Theta \subset \Re$. A common approach is to use the reduced log-likelihood $l(\theta;y) \propto \ln L(\theta;y)$.

For instance, the conditional log-likelihood function for a Gaussian ARIMA(p,0, q) process is

$$l(\theta;y)=\frac{-T}{2}\log(2\pi)-\frac{T}{2}\log(2\sigma^2)-\sum\frac{\varepsilon_t^2}{2\sigma^2}$$

where $\varepsilon_t = y_t - c - \phi_1 y_{t-1} - \cdots - \phi_p y_{t-p} - \theta_1\varepsilon_{t-1} - \cdots - \theta_q\varepsilon_{t-q}$.

Using MLE the estimation is performed by solving the system of equations $\nabla l(\widehat{\theta})=0$, usually referred to as likelihood equations. In both cases, there is no closed-form or explicit solution and therefore, one should use numerical maximization. The idea would be to make a number of distinct guesses for θ, and try to infer the value of $\widehat{\theta}$ for which $l(\theta;y)$ is the largest.

We refer the interested reader to the references [151] and [76] in order to get an in-depth understanding of the linear time series modeling and analysis. One can also find there concepts about the criterion for choosing ARIMA model order, i.e. the parameters *p* and *q*, parsimoniously.

5.3. Design Rationale

In our architecture, the second functional block, namely the Prediction Unit (PU) plays the important role of verifying how stable is the information flowing from the DLPF unit. Additionally, the PU extracts the main component of such signal by removing incidental noise. With such signals in hand (trend and noise), the PU is able to provide the Decision Unit (DU) with more reliable and accurate indications about variability in the network state.

In order to remind the reader the location and main purpose of all functional blocks, in Figure 99 we again reproduce the whole Adaptive-Predictive Unit previously presented in

Chapter 4. Recall that the DLPF does not perform any aggregation of available bandwidth information, hence providing samples to the PU at every RTT. The PU accumulates some estimates before taking any action. We place the PU unit between the DLPF and the DU strategically so that it can decouple the time scale handled by the network and application levels. In such position, it will perform other two important tasks that are providing information about prediction errors and noise levels present in the signal.

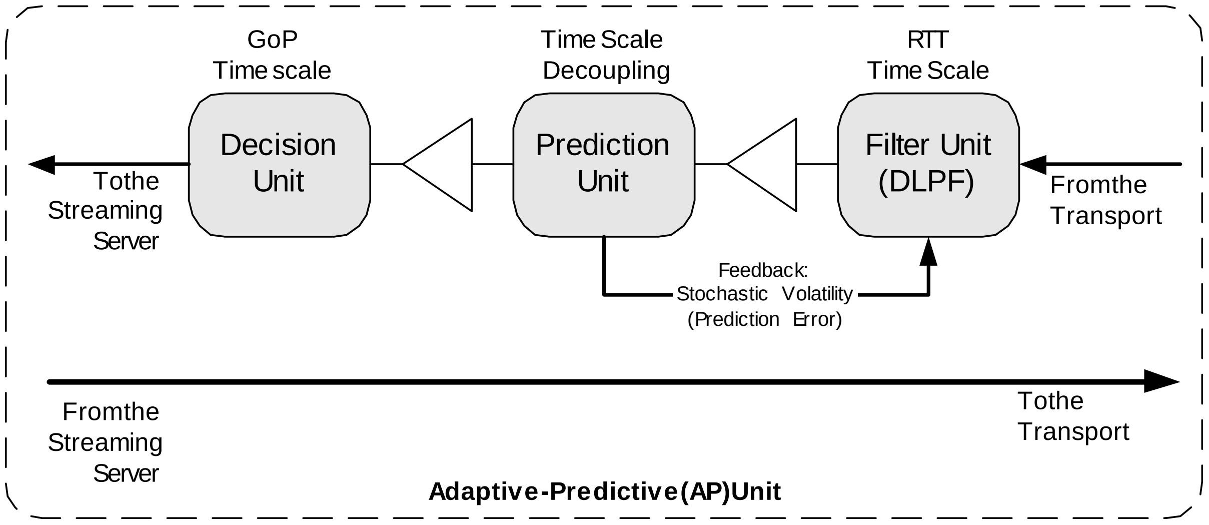

Figure 99 - The Adaptive-Predictive Unit

It is worth stressing that if one is only interested in extracting the main amplitude components at low frequencies, a simple low-pass filter is the appropriate solution. Low-pass filtering would smooth the signal effortlessly. However, it fails to provide information about variability at different time scales. Therefore, any attempt to evaluate noise along with trends should rely on an approach that gathers information in both time and scale (or frequency) domains. The use of Wavelet MRA meets such requirements as we have shown in previous Sections. In our point of view, it is very important to look into the properties of the signal at different time scales other than the original sampling time scale, which in our case is on the order of RTT measurements. With the Wavelet MRA approach, we are able to identify several properties at time scales of frames, GOP, video scenes or any other coarser time scale. Bear in mind that we are not interested in time scales below the RTT, since we are not trying to investigate fractal or multifractal properties on the network traffic.

Therefore, in order to integrate the Wavelet MRA approach into the architecture, we redesign the DLPF, by adding several new components to it including a control decision function. Figure 100 shows the redesigned DLPF functional unit where we can observe the insertion of Wavelet MRA and control decision components.

The control decision plays a strategic role in the redesigned DLPF. It will decide which filtering technique will be active in a certain moment. In other words, based on prediction errors from the PU, it may decide to apply a more complex and precise filtering approach by activating Wavelet MRA. We will provide details of this idea on Section 5.4. Control decision's usual behavior is simply to evaluate for how long the prediction error has been above a given threshold (e.g., using the same threshold level as set in the DU). When the adaptive smoothing techniques are not capable to provide the PU with a less variable time series, i.e. the PU still receives highly variable available bandwidth information and sustains high prediction error levels, the control decision switches to the Wavelet MRA filtering strategy.

The Wavelet MRA component plays the same role as the statistics-based low-pass filtering. However, it will perform noise filtering along with an energy analysis, thus conveying both denoised series (approximation) and variability (energy) level to the DU. During the Wavelet MRA phase, our novel architecture will carry out a signal decomposition process where it extracts the approximation and detailed signals from the original stochastic process from the transport protocol. The output of the Wavelet MRA functional block feeds both decomposed signals into different segments. It will perform soft threshold denoising in the Wavelet coefficients on the approximation signal in order to get the cleanest trend in signal. Meanwhile, the architecture processes the original signal in order to evaluate the energy-based variability evaluation.

We observe that filtering with Wavelet MRA is more complex than the adaptive smoothing. However, it produces very precise information about the mean available bandwidth. For that reason, we advocate that when using the Wavelet MRA functional unit, we can set the prediction error threshold in the DU to a low level (e.g., 10% MAPE). We will explain this strategy later in Section 5.5.

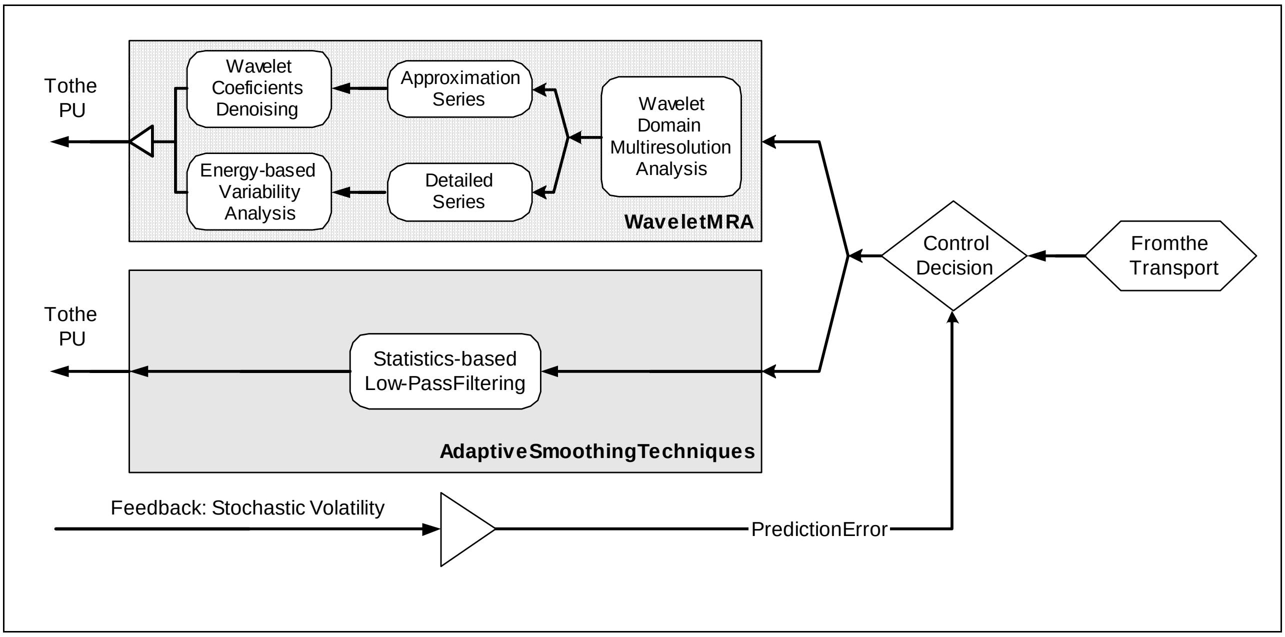
Tothe PU
Wavelet Coeficients Denoising
Approximation Series
Wavelet Domain Multiresolution Analysis
Energy-based Variability Analysis
Detailed Series
WaveletMRA
Control Decision
Fromthe Transport
Tothe PU
Statistics-based Low-PassFiltering
AdaptiveSmoothingTechniques
Feedback: Stochastic Volatility
PredictionError

Figure 100 - Filter Unit (DLPF) - Redesigned

5.4. The Prediction Unit

Before scrutinizing the techniques for analyzing available bandwidth signals from the network, we discuss our decision for placing the PU into our architecture.

As we stated in the previous chapter, a common approach to obtaining smoothness is to use low-pass filters. In this work, we applied low-pass filtering techniques to provide the PU unit with a less variable time series (Figure 101). However, although we showed that we could improve performance in terms of higher video quality with less variability relying only on the DLPF, we do believe that there is space for additional improvement to the overall performance. First, it is well known that EWMA strategies do not need to accumulate samples prior to begin any smoothing procedure. We showed that with a couple of samples we could start filtering immediately. In the case of Wavelets or splines, it is necessary to collect a number of samples before performing its algorithm. Additionally, for linear time series modeling, some approaches need more samples in order to carry out accurate parameterization through maximum likelihood optimization. We argue that in case of highly smooth available bandwidth time series arriving from the network, the resulting smoothness from the DLPF will eventually be satisfactory for the control decision functional block. We refined our architecture by means of evaluating when it would be necessary to activate the Wavelet decomposition in the DLPF.

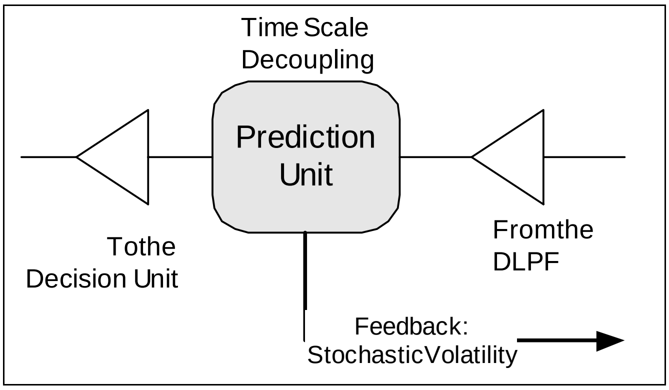

Figure 101 - The Functional Block Prediction Unit

Figure 102 presents our proposal for the PU. We divide such functional block in two phases, namely the Linear Time Series Analysis and the Prediction Error Analysis.

The architecture undertakes all phases straightforwardly. During the first phase, it just receives the denoised signal and carries out the Box and Jenkins procedure that takes the trend signal, find the model order parsimoniously (e.g., using some criterion such as BIC, AICC, etc.). After that, it performs the one-step-ahead prediction. During the second phase, the PU evaluates the prediction error according to a given metric (e.g., MAPE, MSE, MPE, etc).

In an unlikely event that the DLPF fails to provide available bandwidth samples for a short period, the PU will just replicate some latest received samples in order to continue performing its forecasting procedures. This methodology could be seen as a resampling technique [54]. We do not foresee this imposing major impact on the overall system performance.

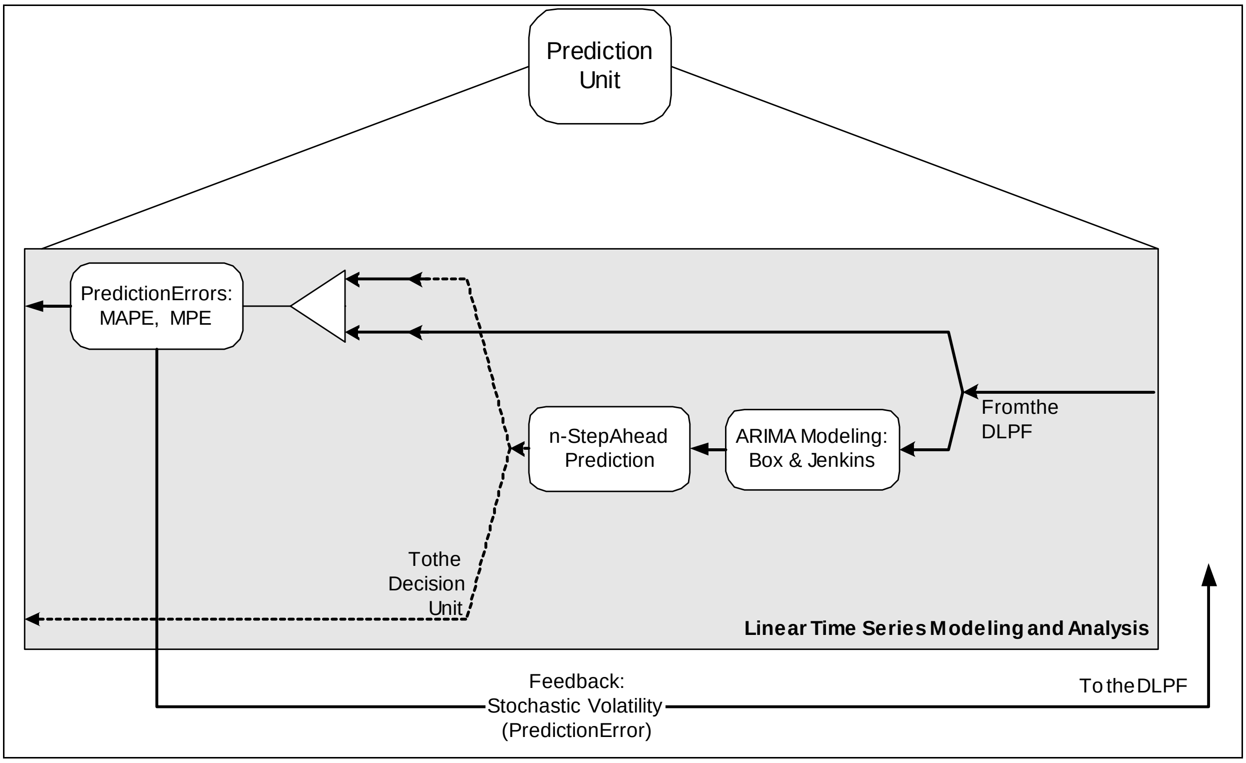
Prediction Unit
PredictionErrors: MAPE, MPE
n-StepAhead Prediction
ARIMA Modeling: Box & Jenkins
Fromthe DLPF
Tothe Decision Unit
Linear Time Series Modeling and Analysis
Feedback: Stochastic Volatility (PredictionError)
To the DLPF

Figure 102 - Functional Description of the Prediction Unit

We propose that at this point, the decision unit (DU) should not use only the prediction values from the PU. We find that it is wiser to the decision unit to use the one-step ahead forecasting value in conjunction with the prediction errors in order to verify if the PU is providing accurate values. By analyzing a given metric for evaluating prediction errors, the DU will make a decision if the network is in steady state and if it is possible to improve quality beyond its current level. If so, it can increase the number of bits of the stored MPEG-4 FGS video in the next GOP. We will present the whole heuristic associated with the control decision performed by the DU.

5.5. The Decision Unit - DU

In the previous chapter, we argued that the video server should only change its quality level when it is safe to do so. By safe, we mean that such change will bring a positive impact on the perceived user quality, presumably. Recall that in our proposed architecture the streaming server works at the Group of Pictures (GOP) time scale domain (i.e. on the order of a few seconds), whereas transport protocols work at the RTT domain (i.e. on the order of ten to hundreds of milliseconds). As far as our architecture is main concerned for adaptability of the streaming server, the Decision Unit focuses on how to manage state transitions, which primarily means managing adaptability for the EL, in order to provide reduced quality variation. By relying on the evaluation of the smoothed available bandwidth, which is the result from the filtering and prediction procedures, the DU provides efficient management for streaming additional fractions of the EL by means of heuristics. It will use some auxiliary sources of information for the adaptation heuristics. Among them are the prediction errors of the available bandwidth or the energy-based variability level. A low value of these metrics means network stability and that it is apparently safe to increase quality using additional bits from the EL. On the contrary, higher values point out instability and the streaming server should keep its current transmission rate to ensure low variability. All information used in this functional block comes from the PU.

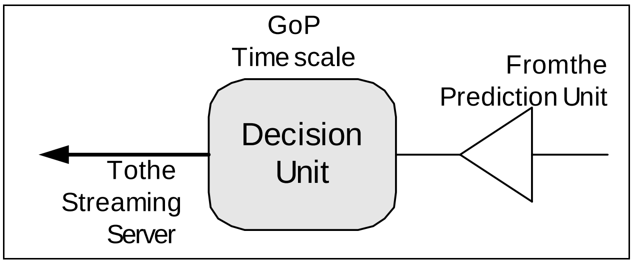

Figure 103 - The control decision functional block - DU

To summarize, the DU obtains knowledge about the network conditions frequently. Using the predicted value along with the prediction errors the DU verifies the stability in the network. In the long run, the energy-based variability level will also provide detailed information about stability in the available bandwidth samples coming from the DLPF. By relying on this information set the streaming server determines how to send portions of the EL. Figure 103 presents the information flow between the functional blocks PU and DU.

Following the work of Balk et al [14] and Gotz [69], we describe the heuristic in terms of a Finite State Machine (FSM). There are some remarkable differences between their approach and ours. First, one should notice that although we also rely on a FSM with similar states for developing our control decision functional block, the rules or conditions, variables, and input events are highly distinguishable in each approach. For instance, they rely on the regular estimation of the available bandwidth in order to decide when the system will switch from one state to another, whereas we use a set of information, namely the predicted value, the denoised series, and the prediction error. Second, dissimilarity refers to the type of the scalable video used. They work with multi-layered encoded videos whereas we work with fine granular scalable videos.

As expected, the FSM is similar for both proposals, since we designed states and state transitions with similar objectives. In order to simplify our problem, we decide to work with coarser granularity at the EL level. For instance, the streaming server could set this granularity based on the following method. As the encoder limit the maximum sending rate for the EL at R_{EL} (see Appendix 2 for details), the streaming server is able to identify how many bits in a GoP will generate a sending rate that matches the predicted available bandwidth. The work in [47] follows this approach and utilizes a dynamic programming methodology to distribute available bits into frames in a GoP or scene shot. They found out that the scene-by-scene adjustment (aggregation) of the EL rate (R_{EL}) reduces the computational complexity of the optimization significantly compared to video frame-by-video frame optimization. Streaming with aggregation also smoothes bandwidth requirements in the network. However, the main drawback in this technique refers to the proper segmentation of the video into scene shots. As the authors discuss in [47], determination of scene cuts that takes into consideration those motion changes between director cuts is still a subject of ongoing research. In this thesis, we assume a GoP-based aggregation in the streaming server.

It is worth stressing that we will certainly improve the final rendered quality if we set an appropriate set of levels for the grouping the EL. With today's available technology, we argue that such difference in the sending rate for each layer should be approximately 200Kbps [46] although it deserves a careful investigation.

Let $EL = \{EL_i\}, i = 1 \ldots 10$ be the set of states, each corresponding to distinct MPEG-4 FGS EL encoded video levels (R_{EL}), which means the minimum available bandwidth requirement within the corresponding level. Let also $FGI = \{FGI_i\}, i = 1 \ldots 10$ be the Fine-Grained Increase rate states, and $FGD = \{FGD_i\}, i = 1 \ldots 10$ represents the Fine-Grained Decrease rate states.

Figure 104 shows the FSM of the proposal algorithm for the control decision functional block, DU. The initial state EL_0 corresponds to sending rate at the Base Layer level with the lowest quality available. We argue that in this state, the available bandwidth at the network is at least above the BL requirements. Intuitively, we are relying on any kind of admission control before the system enters in the state EL_0. Such admission control can be performed by the media provider or can be a simply consequence of the end-user decision. Hence, when the DU receives the set of information from the PU, namely the one-step ahead predicted available bandwidth, the prediction error and possibly the energy-based variability level, the system is able to initiate the transition to FGI_0. The *FIN* state reveals that there is no sufficient available bandwidth, and the system are not able to jump to initial state EL_0.

While in any FGI_n states, the system keeps track of a number of state parameters. Considering the information from the PU arrives at GOP time scales, the system checks the first condition that is if the available bandwidth estimate is equal or above its actual level.

Consider that the system starts in a given state, EL_n. We describe the case where the available bandwidth is less than its current sending rate. In this phase, we argue that such shortage of network resources can be for a short period and server and receiver buffers can handle this appropriately. In the next GOP time window, if the available bandwidth is still below the current level, the system switches to the FGD state, where it stays until the next GOP time window. In the case of persistence of shortage of bandwidth, the system finally switches from the FGD to the EL_{n-1} state. One should observe that the DU could take up to three GOP time windows in order to switch to a lower state.

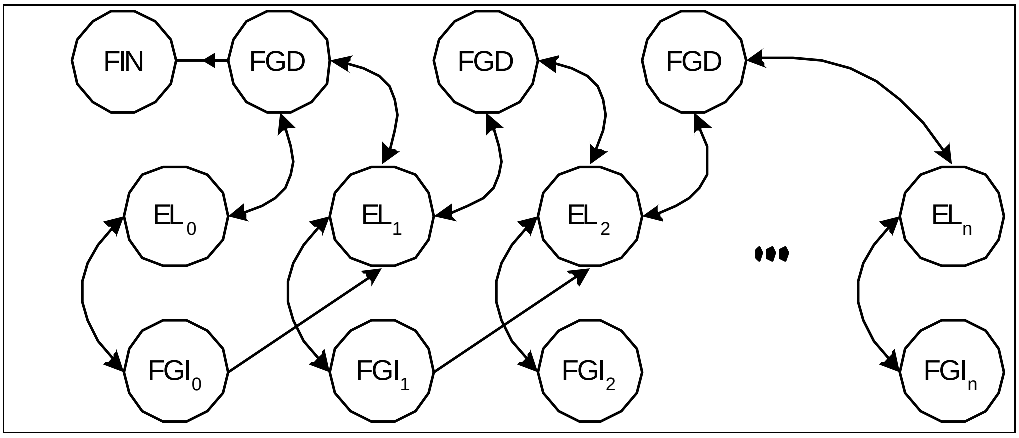

Figure 104 - Finite State Machine for the Control Decision functional block

In the following, we describe the case where the available bandwidth is more than the actual EL_n requirement. Please note that we are departing from a FGI_n state. The DU should identify whether the available bandwidth is large enough to support switching the EL_{n+1} level. If not, the DU stays in state EL_n. If it is possible for the network accommodate the minimum sending rate for the EL_{n+1} level, the DU should firstly verify how stable is the information coming from the PU. As such, it checks if either the energy-based variability level or the prediction error is below a given threshold. If so the system could safely switch to the EL_{n+1}. On the other hand, if one of these variables is above the threshold, one of the following is happening within the network:

a) In the case of a large prediction error, it means that a non-stationary behavior could exist in the available bandwidth series with some long-term increase or decrease in the mean value. In such a case, the linear model could not be able to forecast precisely, thus leading to a sudden increase in the prediction error.

b) In the case of a variability level above a given threshold level, it means that the network is in a situation of high variability, as we have shown in Figure 95 to Figure 97. In both cases, the DU should either not switch to the EL_{n+1} state nor to FGI_n. By following this behavior, the DU does not become aggressive in response to sporadic excess of available bandwidth on the network path. Intuitively, it will only switch to the EL_{n+1} state in a scenario of low variability and sustainable increase of network resources. In other words, transitions to higher levels only occur when all conditions are satisfied.

At this point, we propose the use of a timestamp label for each measurement of variability (i.e., prediction errors or energy-based noise level). DU knows which information is more recent by checking timestamps in both prediction errors and energy-based noise level. After that, it can use appropriate control decision when following the FSM heuristics.

Our proposal follows a conservative behavior and differs from other research studies as far as it concerns to fairness. We do believe that we can achieve less quality variation since there is minimal concern to fairness issues.

We conclude this Section by making comments on the computational effort for each independent unit and its corresponding processing time. Determining the execution time for each functional block is an intricate task. An accurate analysis of the processing time for the whole architecture is only possible with real implementation. The performance evaluation carried out in this thesis relies on a simulative approach, thus each occurring event in the simulation is likely to be unrelated to real-world time. However, based on previous experience [49] we conjecture that such processing time will be limited to a few milliseconds. All DLPF techniques place a minimal performance burden on the architecture since it does not need to accumulate samples and its computational complexity is very low. Simulation time for time series forecasting (on the Prediction Unit) and Wavelet analysis (on the DLPF) depends on the length of the time series. As we use only 25 samples for the forecasting procedure, we assume that the resulting processing time will also have a negligible load on the overall performance. Finally, we can use the above arguments to suppose that the proposed algorithm for the DU will not add significant processing time, since it is very simple to implement.

5.6. Simulation Results

We now go through a complete performance analysis of the proposed architecture. In Chapter 4, we simply evaluate the benefits of tracking the mean value of the available bandwidth by relying on adaptive statistical filtering techniques. We observed that most tested algorithms in such functional unit provide improvements on the expected perceived video quality rendered to the end user. However, with the proposal of new functionalities, such as the Wavelet MRA, the Prediction Unit, and the Decision Unit, it is imperative that we conduct a new set of simulation experiments in order to verify their correctness, thus validating our proposal.

As the careful reader may have noticed, the incorporation of the PU requires setting a threshold value for a proper behavior of the DU, as we described last Section. Therefore, in order to support our decision concerning acceptable range of threshold values, we carried out some simulations to assess the magnitude present in the prediction error metric. By performing such evaluation, we are trying to reveal any persistent variability in the available bandwidth estimates.

In time series analysis, it is common to observe and evaluate several metrics for the prediction error. The decision concerning which metric is more appropriate will essentially depend on the application and type of data. After fitting forecasting models to a given data set, statisticians usually have a sort of performance-related criteria in order to compare them. In fact, a usual procedure is to evaluate periodically some error measures, i.e., some statistical measures of goodness of fit. Typical forecasting errors include the Mean Squared Error (MSE), the Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE) or Mean Absolute Deviation (MAD), the Mean Absolute Percentage Error (MAPE), the Mean Error (ME), the Mean Percentage Error (MPE), the Total Error (TE), and the Total Absolute Error (TAE). Among these measures, some of them do not have useful interpretation in our application scope, such as the TE and TAE, since we do not evaluate accumulated errors overtime. For the same reason, MSE, RMSE, MAE (MAD) and ME will depend on the range of the input values (i.e., they are similar in scale to the available bandwidth data set) used to model the general ARIMA (*p, d, q)*. In other words, as such information is not known a priori, we should set the threshold value by guessing. Therefore, the Mean Percentage Error (MPE) and Mean Absolute Percentage Error

(MAPE) metrics match our purposes, since we can evaluate the prediction error by inspecting the percentage error in the prediction errors. Particularly, MPE indicates whether the forecasts are positive or negative biased, where as MAPE do not consider biases. Both measures will roughly give a hint of the variability persistence on the network.

We present a formal definition for the MPE and MAPE metrics. Let $y = \{y_n\}, n = 0,\ldots,M$ be an original series and $\hat{y} = \{\hat{y}_n\}, n = 0,\ldots,M$ a series with forecasted values. The Mean Percentage Error is defined as

$$MPE = 100 * \frac{1}{M}\sum_{n=0}^{M}\left(\frac{\hat{y}_n - y_n}{y_n}\right)$$

whereas the Mean Absolute Percentage Error is defined as

$$MAPE = 100 * \frac{1}{M}\sum_{n=0}^{M}\left|\frac{\hat{y}_n - y_n}{y_n}\right|.$$

Having selected the MAPE the MPE as prediction error measurement, we still face the problem of defining the appropriate threshold level for the control decision in DU (see the Section 5.5 for details).

Figure 105 and Figure 106 present the overall prediction errors using MPE and MAPE after streaming processing 108.000 samples from the network trace. All metrics were measured after selecting some filtering strategies (e.g., Trigg & Leach, STES), followed by the forecasting procedures in PU. In general, most results for MAPE are below the 10% limit, whereas for MPE we observe errors in the range below -4% and 4%. Although these results suggest that the overall prediction errors keep below some well-defined threshold, we should analyze their behavior over the dynamic changes in the network. In other words, we should have a clear picture of the MAPE and MPE metrics behavior when evaluated continuously in different aggregation steps.

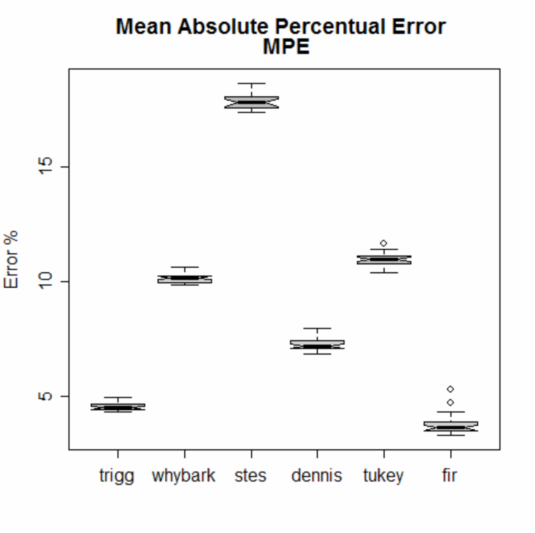

Figure 105 - Prediction Error (MAPE) Evaluation after Filtering: Overall result

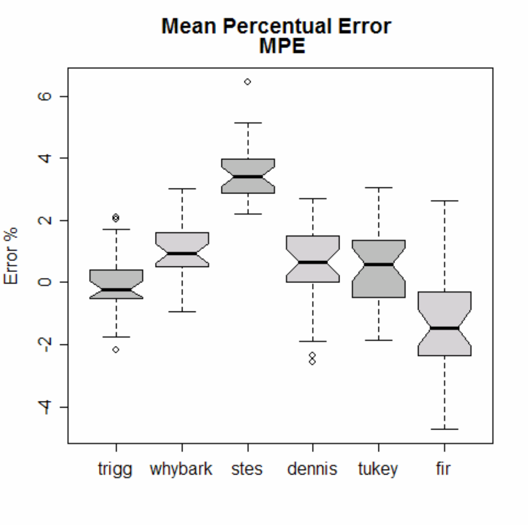

Figure 106 - Prediction Error (MPE) Evaluation after Filtering: Overall result

Considering the previous simulation scenario presented in Chapter 4, where 108,000 network traffic samples were available, we decide to aggregate samples at several aggregate levels, namely at 300, 3000, 6000 and 9000 samples. Such analysis will support us to define a suitable period for re-examination of the prediction errors.

Figure 107 to Figure 110 present the simulation results after evaluating the MPE metric at the PU and for several adaptive smoothing techniques. Comparing to the overall performance presented in Figure 106, the combination of linear forecasting with input data series coming from the adaptive filtering algorithms yield prediction errors (MPE) sometimes far beyond the general profile (Figure 106). We observe that performing aggregation at smaller sample sizes and evaluating such prediction errors, we will mainly reflect some variability still present in the network traffic. For instance, STES filtering generate MPE near 10% in highly dynamic fluctuations in network traffic. The remaining filtering strategies present MPE profiles mostly below a 5% upper limit. These results warning us to take further cautious when setting the threshold level for the DU. Observe the one could erroneously define a 1% threshold in DU for the MPE metrics based on the general profile.

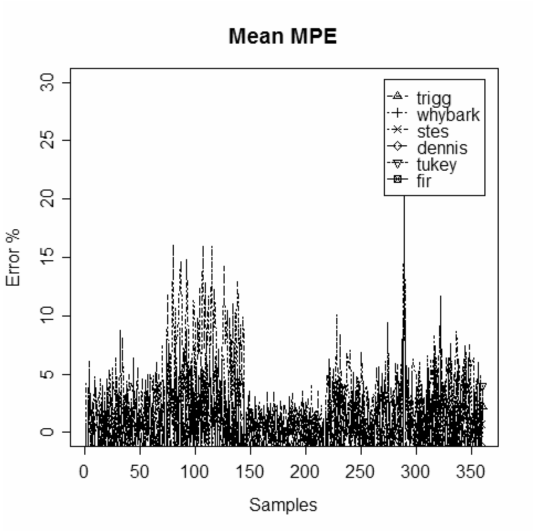

Figure 107 - Prediction Error (MPE) Profile: Aggregation at 300 samples

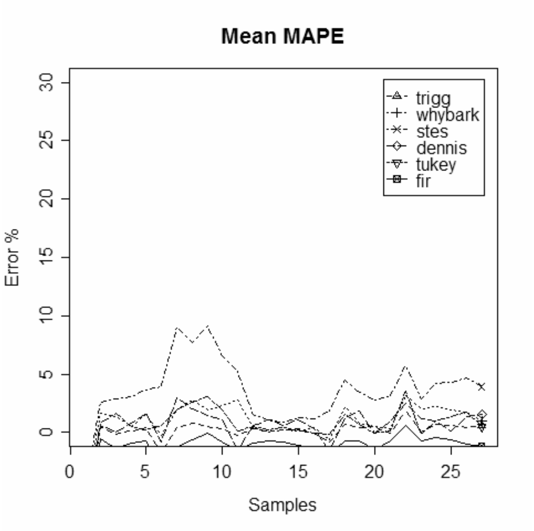

Figure 108 - Prediction Error (MPE) Profile: Aggregation at 3000 samples

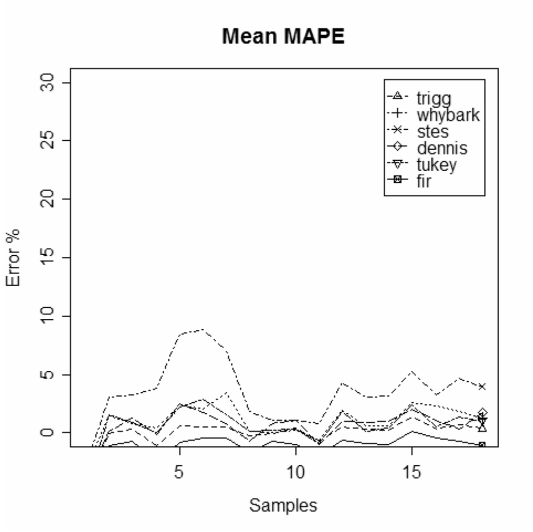

Figure 109 - Prediction Error (MPE) Profile: Aggregation at 6000 samples

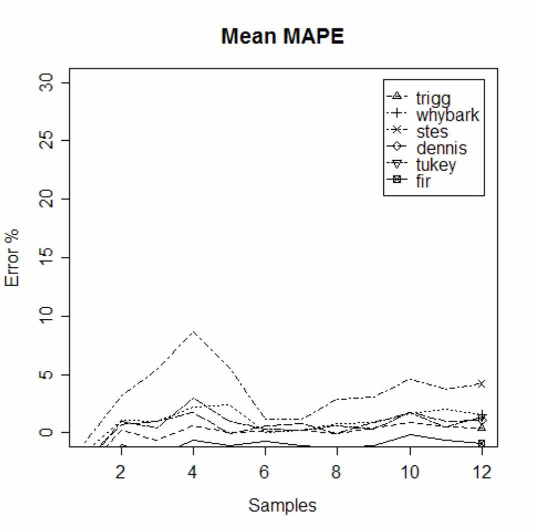

Figure 110 - Prediction Error (MPE) Profile: Aggregation at 9000 samples

Figure 111 to Figure 114 present simulation results after evaluating the MAPE metric at the PU and for several adaptive smoothing techniques. We now compare results to the overall performance presented in Figure 105. As expected, we continue observing the combination of linear forecasting with input data series coming from the adaptive filtering algorithms yield prediction errors profiles for MAPE very different from the general one (Figure 105).

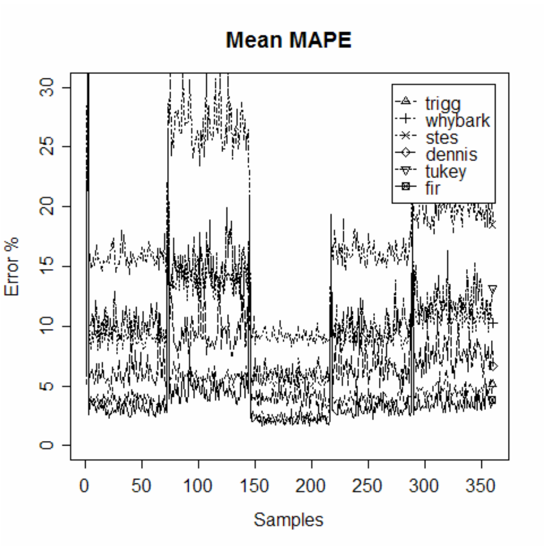

Figure 111 - Prediction Error (MAPE) Profile: Aggregation at 300 samples

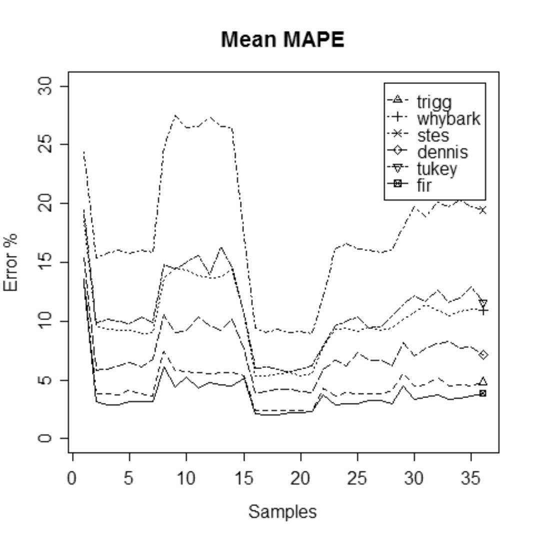

Figure 112 - Prediction Error (MAPE) Profile: Aggregation at 3000 samples

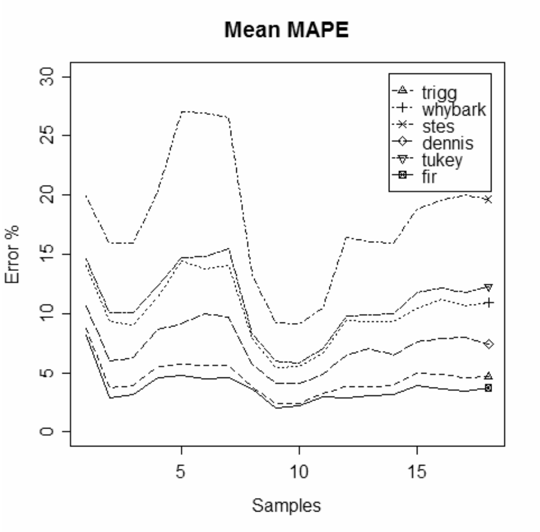

Figure 113 - Prediction Error (MAPE) Profile: Aggregation at 6000 samples

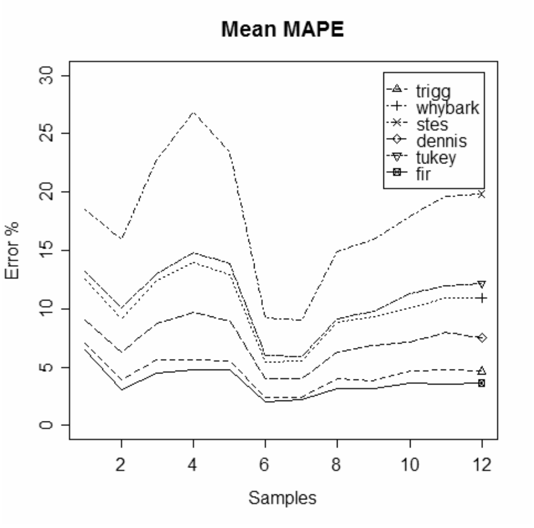

Figure 114 - Prediction Error (MAPE) Profile: Aggregation at 9000 samples

In the following four Sections, we present simulation results for the entire architecture with all functional blocks activated. For these simulations, we utilize MAPE as the prediction error measurements and a threshold level of 10% for the DU control decision.

5.6.1. DU (MAPE, Threshold: 10%) - Movie: The Firm

Figure 115 to Figure 120 show the notched Boxplots for the expected standard deviation for six distinct filter implementations (Tukey, Trigg, STES, FIR, Dennis, Whybark) when the streaming server transmits the movie The Firm.

Simulation results show that the standard deviation for ABR in all simulations presents a unacceptable level of 2dB. As commented earlier, this variability level will certainly reflect in video distortions such as flickers. All results from simulations when the whole architecture is active achieve statistically significant less variability. Looking closer at such difference (more than 1dB), we can infer that the received user will have less distortion.

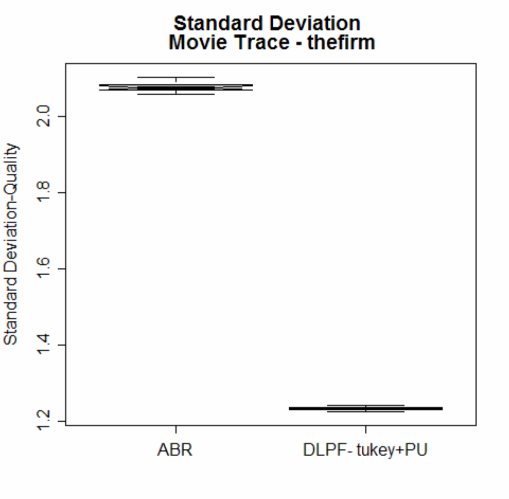

Figure 115- Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – The Firm

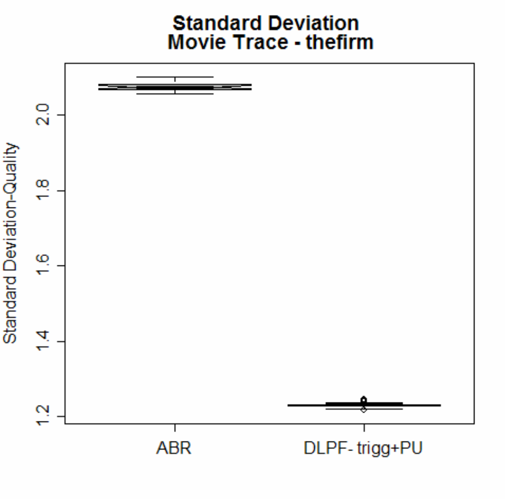

Figure 116 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – The Firm

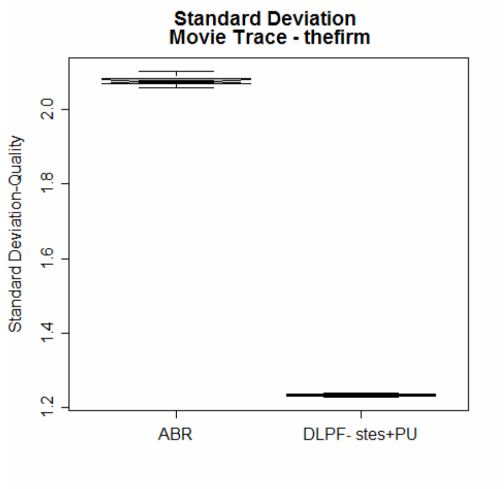

Figure 117 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – The Firm

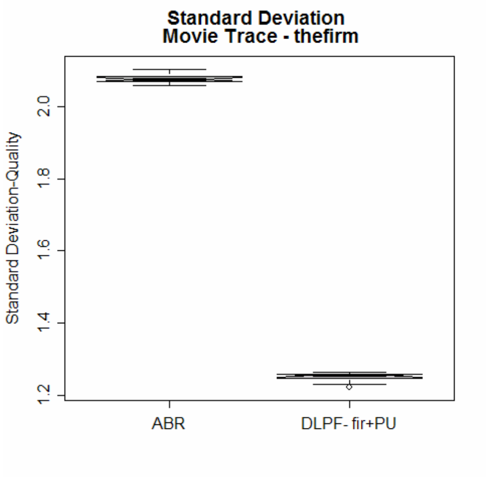

Figure 118 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – The Firm

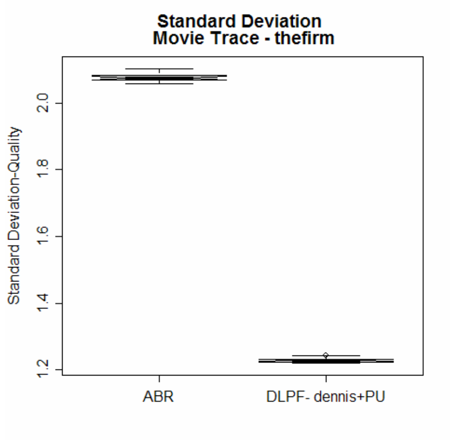

Figure 119 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – The Firm

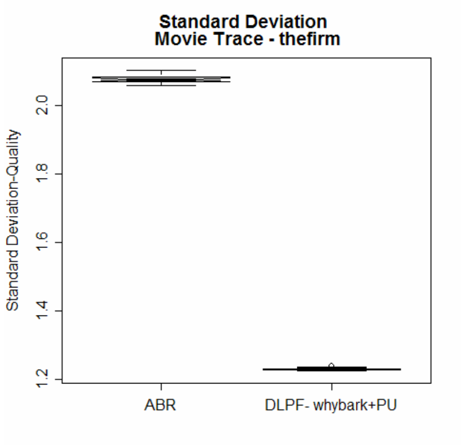

Figure 120 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – The Firm

5.6.2. DU (Threshold: 10%) - Movie: Oprah

We now present simulation results for the movie Oprah. Figure 121 to Figure 126 show the notched Boxplots for the expected standard deviation for six distinct filter implementations (Tukey, Trigg, STES, FIR, Dennis, Whybark) when the streaming server transmits the movie Oprah. Simulation results show that the standard deviation for ABR in all simulations presents a level above 1dB, but below 2dB. This variability level still reveals video distortions. All results from simulations when the whole architecture is active achieve statistically significant less variability.

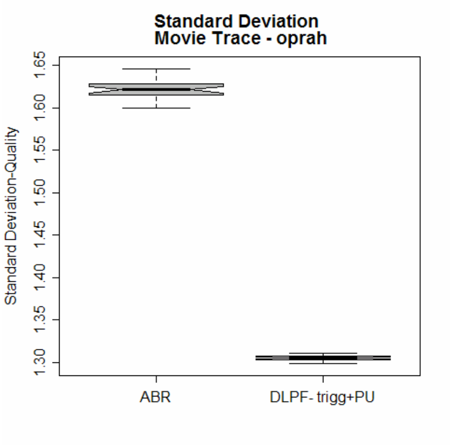

Figure 121 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Oprah

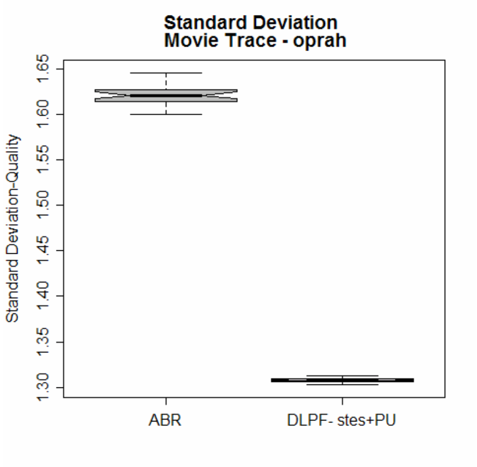

Figure 122 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Oprah

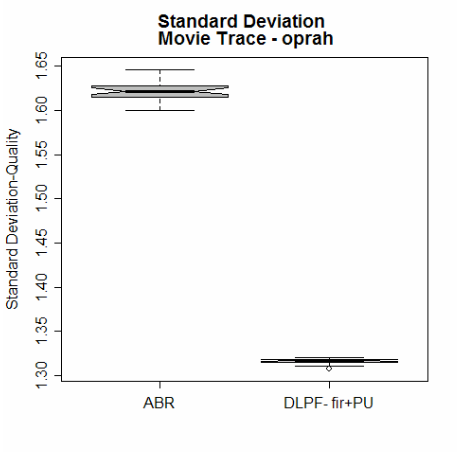

Figure 123 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Oprah

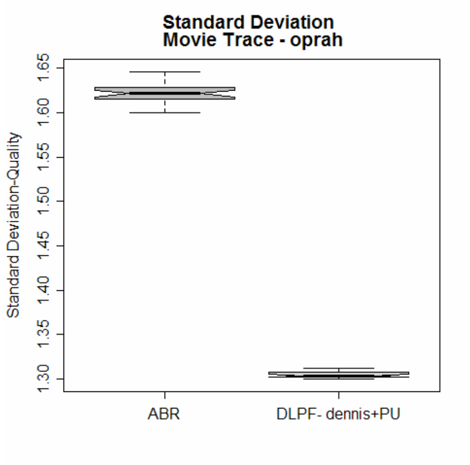

Figure 124 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Oprah

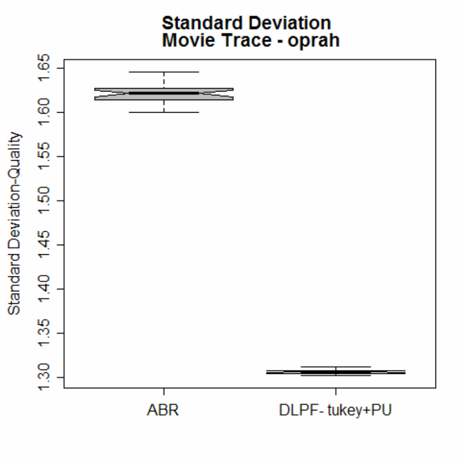

Figure 125 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Oprah

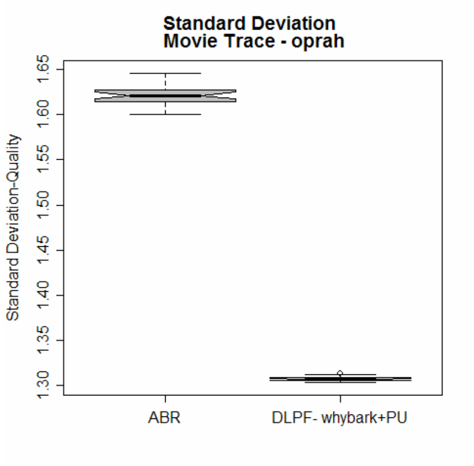

Figure 126 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Oprah

5.6.3. DU (Threshold: 10%) - Movie: Star Wars

As in the previous Sections, Figure 127 to Figure 132 present simulation results for the movie Star Wars. Simulation results show an improvement of 1dB for the standard deviation when relying on our novel architecture. For ABR-based simulations, the obtained results perform poorly with a standard deviation above 2dB.

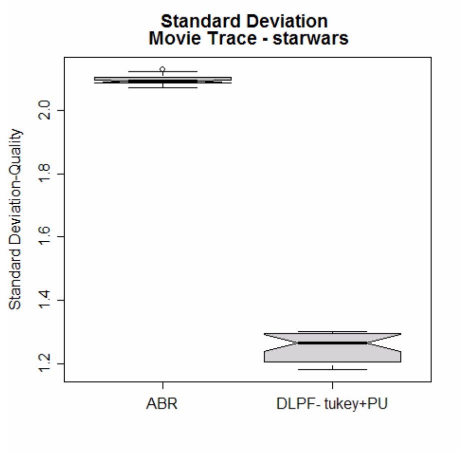

Figure 127 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Star Wars

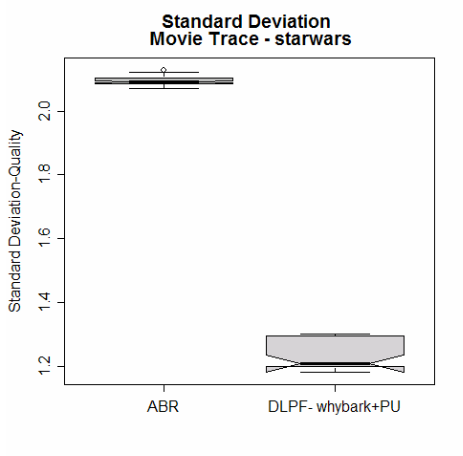

Figure 128 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Star Wars

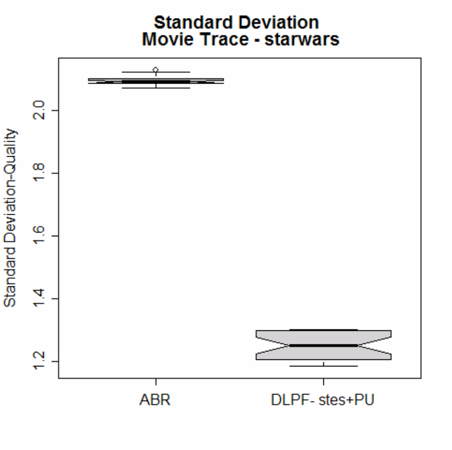

Figure 129 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Star Wars

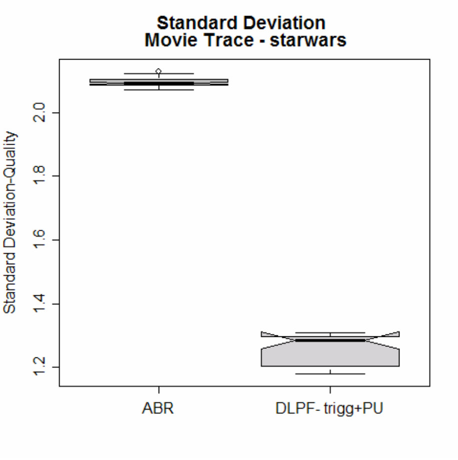

Figure 130 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Star Wars

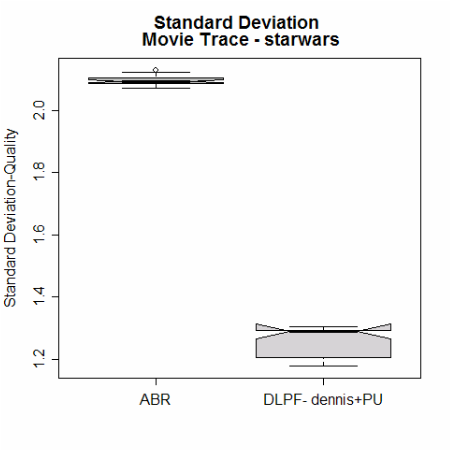

Figure 131 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Star Wars

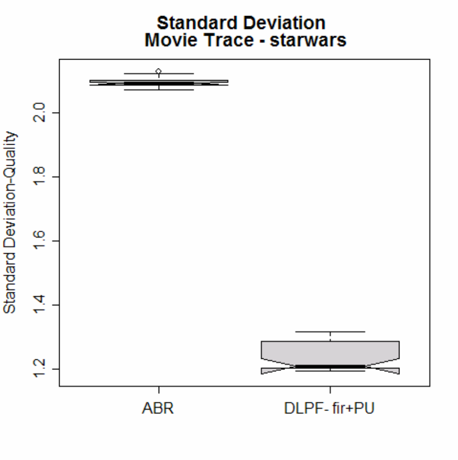

Figure 132 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Star Wars

5.6.4. DU (Threshold: 10%) - Movie: Toy Story

Finally, Figure 133 to Figure 138 present simulation results for the movie Toy Story. We do not have additional comments on these simulation results since they perform closely to all previous video traces. Simulation results still show an improvement of 1dB for the standard deviation when relying on our novel architecture whereas the standard deviation for ABR in all simulations presents a level above 2dB.

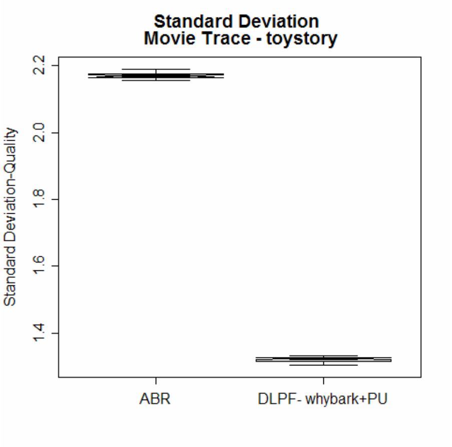

Figure 133 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Whybark) +DU(MAPE, 10%) – Toy Story

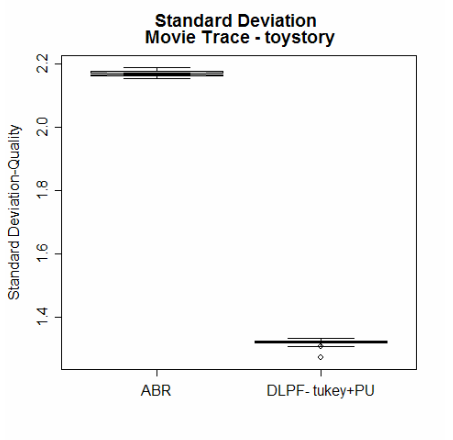

Figure 134 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Tukey) +DU(MAPE, 10%) – Toy Story

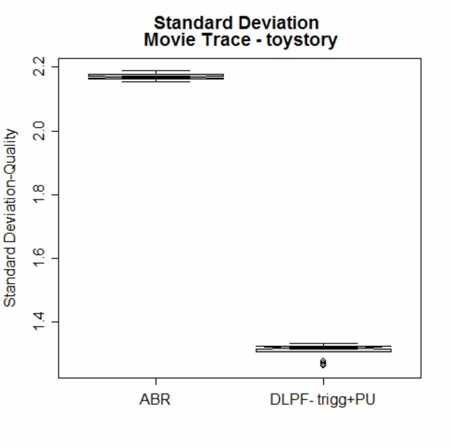

Figure 135 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Trigg) +DU(MAPE, 10%) – Toy Story

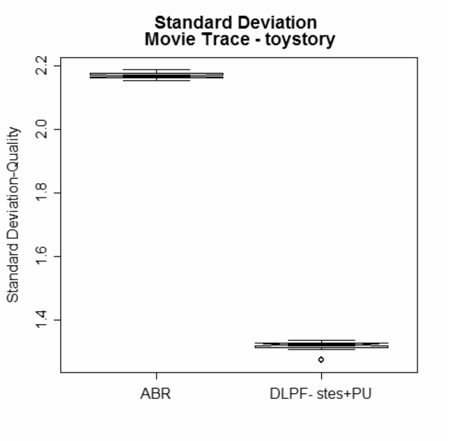

Figure 136 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(STES) +DU(MAPE, 10%) – Toy Story

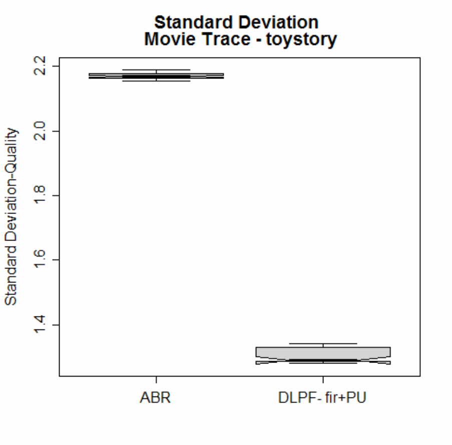

Figure 137 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(FIR) +DU(MAPE, 10%) – Toy Story

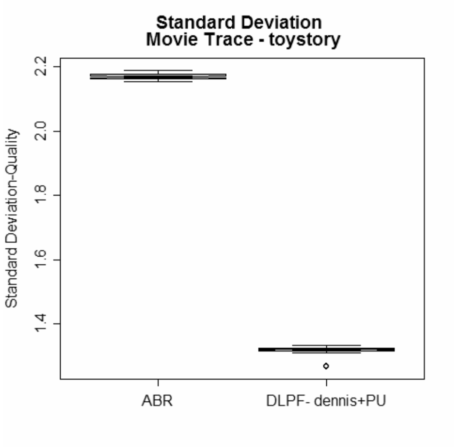

Figure 138 - Notched Boxplots – Standard Deviation: ABR x DLPF +PU(Dennis) +DU(MAPE, 10%) – Toy Story

5.7. Concluding remarks

We have embarked upon a new approach to support smooth streaming of scalable encoded videos. After an extensive description of all techniques within the functional blocks in the proposed architecture (i.e., time series analysis and forecasting techniques, Wavelet Multiresolution Analysis, and the algorithm behind the decision unit), we have investigated the performance behavior of our architecture when streaming several videos at different parameterization. It is worth stressing that we some simulations only gave support to a suitable parameterization. We have found that that variability of the expected received quality for ABR simulations presented significant quality deteriorations with dispersion levels close to 2dB, which is unacceptable. On the other hand, all results when deploying all functional units of our architecture achieved statistically significant low variability.

In order to affirm the benefits of deploying our architecture in streaming servers for pre-stored encoded video, Figure 139 to Figure 141 show the result of previous simulations but under a different perspective. During all simulations, we counted the number of state changes for several parameterizations (filter choice) in the DLPF. In addition, we define different threshold levels for the mean average percentage error in which the DU makes decisions for changing states. We should interpret the number of changes in states as an additional metric for quality. It represents how many times the quality at the receiver changed during playback of the whole video. The x-axis represents the MAPE threshold, which means that the DU only makes decision about changing states (from EL to FGD or FGI, and vice-versa) if the prediction error is below such threshold. As expected as we increase the threshold level we increase the tolerance for large prediction errors, thus increasing the number of changes. On the other hand, we should not define very low level for the error measurement since low values imply that the system will eventually never change its current state. This could impair an improvement in video quality even if there are plenty of network resources. We briefly discussed this issue in Section 5.4, but at this point, we have further arguments for defining the 10% MAPE threshold in our simulations.

As a first impression, we think that the number of changes in states is very high. Recall that all video traces used in this thesis have 108000 frames and 9000 GoPs. Figure 139 shows that for a 50% tolerance in error prediction for the decision control, the number of changes per-frame achieve around 2000, for all DLPF algorithms. This number means that around 1% of frames changed quality. Although we did not show here, for GoP-based evaluation follows the same pattern. Figure 140 shows a bar plot for the same set of simulation results. It is important to compare such results with the number of changes when the streaming server relies only on the raw information from the transport protocol (e.g., the ABR case). Figure 141 shows the same simulation results (Figure 139) in a logarithm scale. The straight line represents the ABR case. From this result, one should observe that the deployment of our proposed architecture reduce the number of changes in one to three orders of magnitude, in logarithmic scale. These results point out a promising utilization of our architecture in network scenarios with high variability, such as some access or wireless networks (see Chapter 6 for future work).

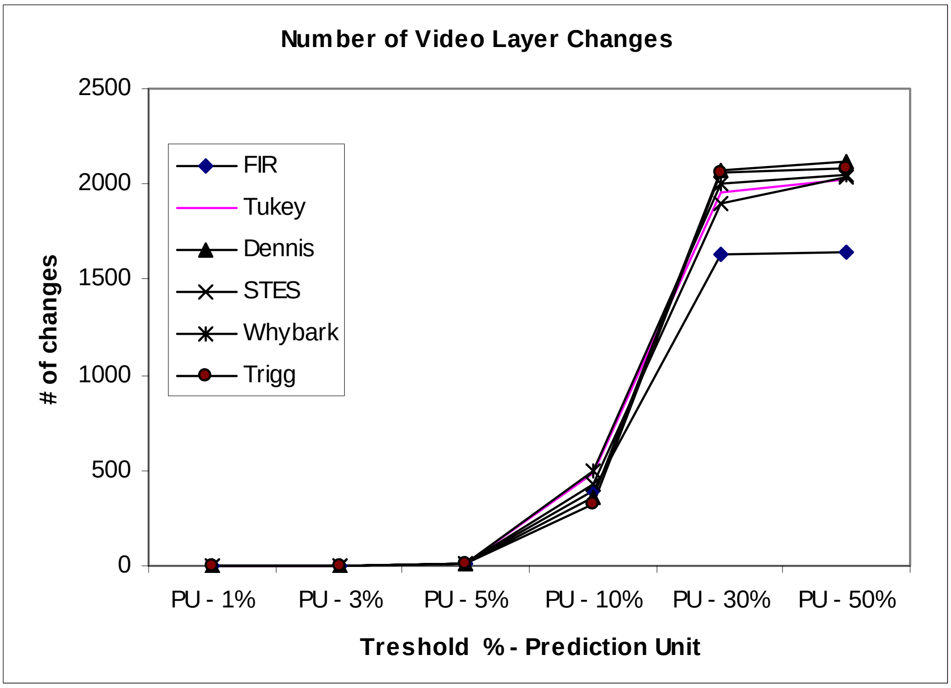

Figure 139 - Number of Changes in states (200Kbps, frame-based) according to PU Thresholds

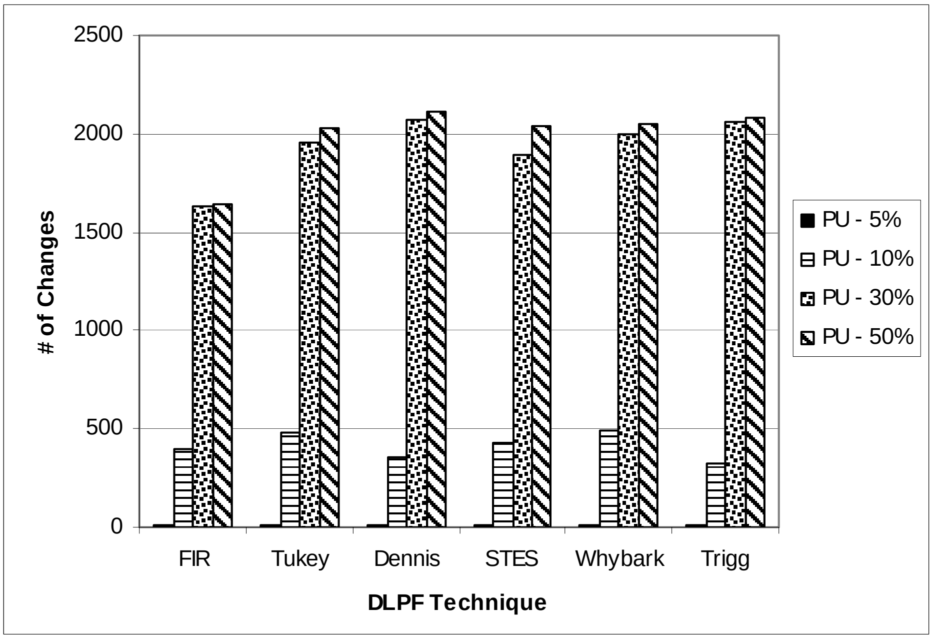

Figure 140 - BarPlot: Number of Changes in States (200Kbps, frame-based) according to PU Thresholds

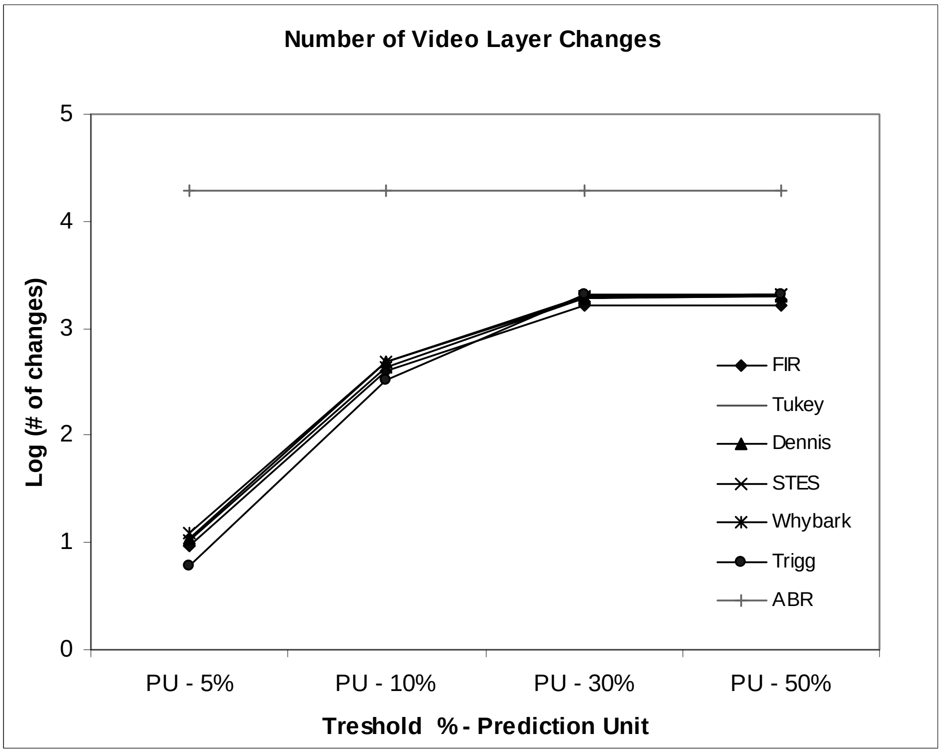

Figure 141 - Number of Changes in states (logscale, frame-based) according to PU Thresholds

Chapter 6. Concluding Remarks

In this chapter, we first summarize the major results of this thesis and then describe some ideas for future work. Section 6.1 summarizes this dissertation and highlights its contributions. Section 6.2 provides a discussion of the topics that need further research.

6.1. Summary of Thesis Research and Contributions

This dissertation relied on recent proposals for congestion control and scalable encoded video to design a novel architecture for multimedia streaming on the Internet. Chapter 1 suggested that there are many technical challenges for a wide deployment of video services over the Internet. In the video encoding and networking area, scalability is the crucial factor for delivering good video quality over unpredictable highly dynamic networks. We reveal that providing perceptually good quality streaming video is a difficult task, since in the available bandwidth in the network path can fluctuate strongly as well as encoded video can exhibit significant rate variability. To circumvent some current drawbacks, we visualized that a proper solution is to make the streaming server to adapt to the network conditions in a fine-grain manner. This is possible by relying on both flexible coding standards and subjacent congestion control mechanisms with explicit feedback notification.

Congestion control mechanisms along with techniques for streaming multimedia over the Internet were described in Chapter 2 and provided the necessary background to a good understanding of all concepts used in this thesis. We discuss several end-to-end congestion control approaches as well as the current development on this research topic. It was argued that there is an explicit trend in providing end-systems with more knowledge about network conditions, through a combination of congestion control and AQM. We also gave an overview of several recently proposed solutions to transport multimedia streaming. We concluded Chapter 2 pointing out directions towards a novel approach in this field.

In Chapter 3, we performed a quantitative evaluation of the end-user perceived media quality of video streaming under a network friendly protocol (TFRC) and a protocol with explicit feedback notification (CADPC). We evaluated the effect of throughput variation of the TFRC Protocol on the received video quality. It is worth stressing that such performance evaluation is the first major contribution of this thesis. In general, transport protocols with explicit feedback notification provide better performance than the traditional loss-based congestion control mechanisms. However, we observed that even with explicit notification, the perceived video quality can still oscillate and argued that it was possible to mitigate the effects of short- and long-term variations in video quality. We had a clear picture that a proper approach to overcome this problem was changing the solution to another level. We envisaged a mechanism that would take into account the volatility in the available rate in order to decide the output sending rate target. We discussed the possibility of smoothing the information received from the transport layer before taking any decision concerning the sending rate. With all these arguments in hand, we advocated that any adaptive multimedia streaming systems should rely on explicit feedback notification from the network, in order to provide the streaming media with both uninterrupted transport services and low quality variation.

The following two chapters (Chapter 4 and 5) presented the other major contribution of this thesis.

In Chapter 4, we took a further step on the analysis and solution of the problem of accommodating the mismatch between the available bandwidth variability and the encoded video rate variability. Recall that we defined the minimization of the quality variability and the maximization of the overall quality as the main goals. We provided all necessary definitions and notation and described some important design decisions for our proposal of a novel architecture. We then provided details about the architectural components with an extensive description of the functional units. To conclude, we carried out an extensive simulation-based performance analysis of one functional block, namely the DLPF. We proposed the use of adaptive low-pass filter for smoothing the raw traffic information flowing from the transport levels. We clearly observed that the DLPF functional block provided better results when compared with the direct use of such information. From those simulations, we concluded that there was an overall improvement for most of the smoothing techniques within DLPF where we obtained similar average quality with less variability. Smoothing the stochastic behavior of the network information emerged to be an excellent approach.

In Chapter 5, we undertook a careful analysis and provided a solution to the problem of decision control for streaming fraction of stored scalable video in networks with explicit feedback. We explored all important design decisions for the proposal of our architecture and provided details about the Prediction Unit as well as the Decision unit. First, we presented motivations for using Wavelet Multiresolution Analysis (MRA) and explained how such technique was deployed within the architecture (specifically in the DLPF). By using Wavelet MRA, we were able to extract the mean available bandwidth precisely along with the noise energy at multiple time scales. We also gave an overview of Time Series Analysis (TSA) with linear models, since such concepts were necessary for an in-depth comprehension of the proposed Prediction Unit (PU). Within the PU, we modeled the main signal with a parsimonious linear time series model, namely ARIMA, which we forecasted changes in signal and evaluated its prediction error. Using the samples from the DLPF, we are able to perform one-step ahead forecasting or to analyze such signal in the Wavelet domain, in order to extract the essential signal energy as well as noise energy. In addition, we proposed an algorithm for the Decision Unit (DU). The DU used either the prediction error measurement or the noise energy heuristically. Finally, to investigate further our proposed architecture, we carried out an extensive simulation-based performance analysis and showed that degree of dispersion of the mean quality for ABR present an unacceptable level close to 2dB. On the other hand, all results from simulations when the whole architecture was active achieved statistically significant less variability. In other words, each functional block contributed for lessening the volatility in quality.

In summary, this thesis makes the following two major contributions:

1. **It presents a performance analysis of multimedia streaming over explicit feedback networks**: In chapter 3, we undertook an in-depth performance evaluation of streaming scalable video over best efforts networks with or without explicit feedback signaling. From these promising results, we built our architecture with the main goal of lessening quality variation.

2. **It proposes an adaptive-predictive architecture for streaming scalable video**: In chapter 4 and 5, we designed and evaluated our novel architecture for streaming scalable pre-stored video. We evaluated all system components in the architecture, thus progressively showing the benefits of each functional block. In general, our framework is broad in its scope in that it treats areas of streaming stored video, performance analysis of congestion control protocols, statistical techniques for smoothing and forecasting applied to network traffic, and signal processing methodologies.

Although the novel architecture itself in conjunction with the overall combination of the statistical and signal processing techniques into the architecture are the major contributions, we believe that one could withdraw smaller contributions from a number of arguments and ideas we explored to support most of our design decisions. Therefore, in addition to the two major contributions, this thesis contains some minor contributions, which represent new insights into existing models and frameworks in the area of multimedia streaming over the Internet. From our point of view, some of these minor contributions are:

1. Based on an extensive (perhaps exhaustive) literature review, to our knowledge this thesis is the first research work to propose the use of explicit feedback information from the network to support multimedia streaming in the Internet.

2. We argue and assert that explicit feedback from recent congestion control proposals, such as XCP and CADPC, provide a solid basis in which multimedia application server designers should rely on. Dealing with only the volatile allowed sending rate, streaming servers do not need to infer network metrics in order to ensure fairness with competing flows as well as network efficiency.

6.2. Future Work

Our novel architecture opens up a broad avenue for future work. We now describe some prospective topics for further research, namely on the field of analytical modeling, application layer scheduling and prioritization, cross-layer design, differentiated networks, multicast, peer-to-peer streaming, proxy-cache and Content Delivery Networks, encoding rate control and FEC, and wireless networks. In the next subsections, we give an overview on previous research studies on each topic and present a brief sketch of how we can add functionalities to our architecture.

6.2.1. Analytical Modeling

Developing analytical models for protocols and networking systems are crucial for assuring stability in the Internet. Research studies in this field devote their energy to dissecting protocols and systems characteristics and then provide sophisticated analysis that cannot be carried out based only on simulation [9] [17] [84] [160] [171] [208] [209]. For instance, Paganini et al [160] designed a congestion control system that scales with network capacity, achieving high utilization, dynamic stability, and fairness. By relying on a primal-dual control law, they ensure the stability of the protocol and its equilibrium features in terms of utilization, queueing and fairness, under a variety of scenarios. In [171], Qiu and Shroff consider a network with both controllable (e.g. TCP) and uncontrollable (e.g., any UDP-based non-responsive application) flows, provide a general model and analyze its queueing behavior.

Under adaptive rate video encoding scenario, a video source must adjust its encoding parameters for matching the transmission rate target to the available bandwidth, thus requiring a analytical model for evaluating such interactions. Gallucio et al [65] advocates that the relationships between such parameters and other state variables for the system could be properly tuned by a careful design. The authors present an analytical framework for the design of the feedback laws, which are essentially modeled using Markov chains. To evaluate performance, they utilized a MPEG Encoder with a rate controller, which adapts the transmission rate by setting the quantizer scale parameter to match the bandwidth variations.

A fundamental constraint for a safe deployment of our architecture (or any streaming video schemes) in real networks is an extensive performance evaluation of buffer requirement at both server- and receiver-side. Recall that we introduce buffers in our architecture to assist compensating short-term rate oscillations. In general, different adaptation policies require different buffering capacity at both network and end-systems. From the point of view of queuing theory, one should observe that to sustain playback rate in a shortage of network resources, i.e., available bandwidth, a slowly responsive adaptation policy usually requires a large amount of buffering. Therefore, we consider that performing an analytical approach for the determination of the minimal amount of buffering is an important further step for continuing this thesis. Such minimal buffer capacity will help smoothing the playback rate oscillations at the receiver, since the video source only observes the filtered sending rate information from the decision unit in our architecture. As a first step in such analytical modeling and analysis, Li et al [135] studied the relationship between buffering requirements and adaptation policies that adapts the source's sending rate to the average available bandwidth. They derived the minimal buffering requirement assuming that sources rely on Additive-Increase Multiplicative-Decrease (AIMD) policy for congestion control (e.g., TCP). They show that the buffering requirement is proportional to the parameters of the AIMD algorithm and quadratic to the application rate and RTT.

An analytical approach for modeling the buffering requirements for our architecture deserves a careful investigation. First, we are dealing with cutting-edge congestion control mechanisms that do not have modeling studies beyond stability and convergence analysis. In addition, we introduce several functional blocks (e.g., the prediction and decision units) that should be also modeled to reflect as close as possible the actual architecture. Second, from the point of view of general AIMD (GAIMD) algorithms [25], the behavior of XCP and CADPC has not been modeled yet. The main reason is that both protocols have distinct characteristics from GAIMD protocols that one should take into account. Although such facts are not encouraging, we keep advocating that we should extend this work by developing an analytical model for the minimal buffer requirements as well as a control-theoretic one for stability and convergence analysis. As these tasks require a huge amount of time and effort, we left them as future work.

6.2.2. Application Layer Scheduling

Working at the application layer, Krasic et al [86] [123] [124] introduce the concept of Priority-Progress streaming, which decomposes application data into units of work, called application data units (ADUs). Each ADU is labeled with a timestamp to capture the ADU's temporal requirements and a priority label in each ADU, which is closely related to the layered characteristics of the media. Based on this concept, they presented a framework for adaptive video streaming and showed how to use adaptation policies and an associated priority-mapping algorithm to build a quality-adaptive streaming system. In summary, the ADUs are prioritized with the goal that priority-order dropping of ADUs will yield a smooth reduction in quality. The priority-mapping algorithm translates the policy specifications into priority assignments on ADUs of priority-drop video. Finally, they present an algorithm for real-time streaming called Priority-Progress streaming (PPS), which combines data re-ordering and dropping in order to react properly to variable available bandwidth.

We visualize a powerful combination of our proposal adaptation policy with the priority dropping mechanisms described in [86] [123] [124]. On one hand, we observe that our adaptation policy and filtering strategies can improve efficiency in quality reduction. On the other hand, their proposal could deal with live-encoded video streaming, which was not addressed in this thesis.

6.2.3. Cross-Layer Design

Mobile multimedia applications require networks that optimally allocate resources and adapt to dynamically changing environments. Cross-layer design (CLD) is a new paradigm that addresses this challenge by optimizing communication network architectures across traditional layer boundaries. For wireless environments, cross-layer optimization could be used to design appropriate adaptive systems. Most CLD studies focus on independent optimization of a given layer [28] [79] [116] [125] [202] [230]. In the bottom-up approach, the application layer adapts to the lower level layers, such as transport, network, data link, and physical layer characteristics, whereas in the top-down approach the lower level layers (i.e., physical, data link, or network layers) adapt to the application layer. There are some hybrid approaches where information for optimization flows in both directions. CLD deployment apparently shows some improvement in performance when compared to a traditional single layered approach.

Ksentini et al [125] address the problem of video transmission (H.264) over wireless IEEE 802.11e by proposing a robust cross-layer architecture that control the error resilience at application layer and the existing QoS-based MAC protocol features. Specifically, the architecture utilizes a data partitioning technique and QoS mapping at the MAC layer and a marking algorithm at the MAC layer that associates each partition with an access category (AC) provided by 802.11e enhanced distributed channel access (EDCA). Thus, the application layer passes its streams along with their requirements to protect the most important video information. Such proposal assures to some extent low degradation of received video stream.

In [116], the authors discuss some technical challenges of cross-layer design with the focus on application-driven for video streaming over wireless networks. Among several arguments, they show that finding a common utility function that allows optimization across different applications is an overwhelming task. They also advocate that cross-layer optimization could improve network efficiency, although they point out that temporary shortages of resources will eventually require penalizing the service quality for some users. They address such challenges by proposing a cross-layer optimization strategy that optimizes the application layer, data link layer, and physical layer using an application-oriented objective function in order to maximize user perceived quality. Ahmed et al [4] propose a CLD for content delivery that combines media content analysis and network control mechanisms. The target application and scenario for their architecture is adaptive video streaming over IP DiffServ-enabled networks.

As we constantly called attention to throughout this thesis, Haratcherev et al [79] also argue that highly dynamic networks, especially wireless environments, imposes burden on both video codecs and network. In such a case, both video codec and the radio layer should adapt to changes in the wireless link quality. In addition, background traffic influence over video flows should be taken into account. Therefore, in [79] the authors present a CLD architecture for video streaming over 802.11 that is capable of adapting to changes in the link quality of the wireless channel. The architecture relies on link adaptation to handle the effects of changes in channel conditions at the MAC level. In other words, it keeps adjusting some MAC parameters in order to achieve optimal quality of packet transmission. On this basis, the architecture uses cross-layer signaling to pass link quality information to the video encoder for adjusting its data rate.

At this point, one could see a reasonable possibility to use our novel architecture as a component in a CLD approach for wireless scenarios. The authors in [79] left as future work the implementation of advanced estimation techniques, such as filtering, in the radio rate controller in order to improve the quality of predictions. Although they are not sure about advantages of using such filtering techniques, since they do not want imposes a heavy computational load, we demonstrated that some filtering strategies are very light load. For that reason, we do believe that our design could be either extended to fulfill the requirements for wireless environments or incorporated in another CLD approach [79] [116] [125] [202].

6.2.4. Peer-to-Peer (P2P) Streaming

In the world of the emerging P2P technology, Radha and Wu [174] evaluate the impact of network-embedded FEC (NEF) in P2P multimedia multicast networks. In such an approach, FEC codecs are placed in some intermediate nodes of a multicast tree in order to detect and recover lost packets within FEC blocks. As the authors argue in their work, the proposed NEF codecs work as signal regenerators in a communication system and can reconstruct most of the lost data packets without requiring retransmission.

Under similar scenarios, namely P2P multicast video streaming, Setton et al [200] propose and carry out a performance evaluation of a peer-to-peer multicast protocol with prioritized packet scheduling at the application layer. Furthermore, they also provide an adaptive video streaming technique for P2P networks among some other contributions. In [42] the authors propose a peer-to-peer streaming solution to address the on-demand media distribution problem based on cache-and relay and layer-encoded streaming. An interesting discrepancy in their work compared to some related work is that they do not the use of P2P layered multicast.

6.2.5. Proxy-cache and Content Delivery Networks

Content Delivery Networks (CDNs) and Proxy systems enable media and entertainment companies to manage effectively multimedia content (i.e., files and streams) over the Internet [5]. By caching multimedia objects (e.g., video and music files) at proxy systems near to clients, a multimedia content provider reduces startup and playback latency as well as the requirements for bandwidth at the network core. The fact of the matter is that deploying well designed Proxy and CDN has attracted a lot of interest in the research community recently [7] [8] [30] [31] [143] [180] [229] [247] [248]. In addition, some research efforts have point out that a suitable approach is to combine proxy caching with video layering or transcoding in order to meet client bandwidth conditions [143].

Chen et al [31] study how to manage existing proxy resources to deliver media content over the Internet efficiently. In a similar research work [30], they proposed a streaming proxy system, namely Hyper Proxy, for coordinating prefetching for uncached segments and segmentation techniques whereas Liu et al [143] proposed an adaptive video caching framework for fine-grained adaptation with post-encoding rate control.

Working with real streaming systems, Zhang et al [248] conducted a study to determine the signaling and data transport details of some well-known streaming media application (RealVideo and QuickTime) and developed a prototype translation proxy for such systems. Rangaswami et al [180] propose the deployment of an interactive media proxy (IMP) server, which transforms non-interactive streams into interactive ones.

In an analytical approach, Wang et al [229] develop a technique to determine an optimal proxy prefix cache allocation to the videos in order to minimize the aggregate network bandwidth cost. The main issue addressed in their paper refers to the problem using proxies for streaming a set of heterogeneous videos from a remote server to multiple asynchronous clients, while achieving low startup delays.

Although Proxy-Cache research studies concerns mainly to optimization issues (e.g., location of proxy servers, replacement policies of multimedia content, etc.), we can observe the need of dealing with dynamic network information from lower levels. Our proposal could integrate any proxy systems to cope with the delivering mechanisms. For instance, Yu et al [247] propose an adaptive proxy-caching scheme for multimedia streaming while taking into account dynamic network conditions and media characteristics. Their main contribution is the proposal of a media characteristic-weighted replacement policy in conjunction with a network-condition- and media-quality-adaptive resource-management mechanism, which re-allocate cache resource for different types of media.

6.2.6. Wireless Networks

Efficient video streaming over wireless networks poses several deployment challenges, such as coping with highly variable limited bandwidth and loss patterns. Video services delivery in such harsh environments has to be improved by using appropriate mechanisms that take into account QoS requirements better. There is an extensive academic research on proposing and evaluating mechanisms for video streaming over wireless networks (WLAN, Cellular, etc.) [11] [29] [32] [33] [34] [35] [65] [137] [177] [212] [223] [252]. Approaches in this research topic span from parameterization at link layer to error correction at application layer. We provide the reader with some relevant research studies in this field, followed by a brief discussion on possible investigations in the context of this thesis.

By relying on priority queuing at the network layer and retry-limit adaptation at the link layer, Li and Schaar [137] propose an error protection method for providing adaptive QoS to layered coded video. In their work, video layers are unequally protected over the wireless link by applying different retry limits at the MAC layer. Cheung et al [33] [34] argue that providing additional feedback information to streaming media servers in wired and wireless networks, to supplement regular feedback from clients, is a key point to improve overall performance. In [33] they propose a double-feedback streaming agent (DFSA) which allows the detection of discrepancies in the resource constraints of both wired and wireless networks. In a follow-up work [34], they examine the use of feedback adaptation for media streaming in 3G wireless networks. In a similar approach to [33], they propose the use of a Streaming Agent (SA) at the intersection of the wired and wireless networks for providing additional feedback to the streaming servers. Radha et al [177] introduce a new concept for wireless video, called transcaling (TS), as a generalization of (nonscalable) transcoding. Within TS framework, a scalable video stream with a limited bandwidth range is mapped into one or more scalable video streams covering different bandwidth ranges. They evaluate their proposal by applying TS on streaming MPEG-4 FGS video coding over WLANs.

In the error correction subject, Gallucio et al [65] study the problem of adaptive forward error correction schemes and define an analytical framework for the performance evaluation of a video streaming system over a wireless link. They deal with the relationships between changing encoding parameters and other variables representing the state of the system (e.g., current network conditions). For instance, the proposed framework allows an evaluation of a MPEG Encoder that uses a rate controller, which adjusts the output rate to adapt to the bandwidth variations while maximizing encoding quality and stability. Similarly, Chen et al [32] develop a rate control algorithm for video encoding, where they consider the encoding complexity variation and buffer variation as well as human visual properties to optimize the rate control efficiency. Additionally, Chen and Zakhor [29] propose the use of multiple TFRC connections as an end-to-end rate control solution for wireless video streaming.

Working on scenarios where the network provides explicit feedback notification, Zhang and Mohin [252] conducted an experimental study of XCP in a wireless network and discovered that it presents some convergence problems. By carrying out a control theoretic analysis of XCP, they studied its behavior in the presence of estimation errors. The key findings are that

XCP will not settle at zero steady-state error, but it will have a error bound. An appropriate router queue size planning can cope with such estimation error. In a similar scenario, Atkin and Birman [11] examine including extra information to a network API for adaptive applications running on a wireless host, such as bandwidth notifications, as a technique for improving control over bursty traffic. Such proposal allows applications to know the state of the network precisely, and to adjust its behavior (i.e., to adjust to bandwidth variations) accordingly.

With a few modifications, we believe that we can deploy our novel architecture in wireless environment, since we designed our architecture with filtering and adaptation components, which will eventually be realistic for wireless networks. We also envisage a powerful combination of our proposal with some of the previous work for wireless environments, such as performing careful modifications and parameterization at the MAC level, in a cross-layer approach.

Appendix 1 - Filters in Time Domain

In this appendix, we provide the fundamental theory of filters in time domain, since such techniques will be extensively used throughout the thesis.

An ordinary linear filter simply converts raw time series data $\{x_n\}_{n=-\infty}^{\infty}$ into another time series $\{y_n\}_{n=-\infty}^{\infty}$ by a linear transformation. In fact, it performs a convolution of the input vector with the coefficient vector (i.e., filter coefficients, $\{\alpha^n\}_{n=-\infty}^{\infty}$), which completely defines any filter behavior. One should observe that when the filter coefficients do not change over time, the filter is classified as Time Invariant Filter. Additionally, the relation among input time series, coefficients, and the output time series implies the presence of linearity or non-linearity. Following a standard definition, by linearity we mean that the output due to a sum of input signals equals the sum of outputs due to each signal alone [77] [207]. In other words, the amplitude of the output is proportional to the amplitude of the input. A formal definition is:

A filter H_n is linear if for any pair of input signals $x_1(\cdot)$ e $x_2(\cdot)$, and for all coefficients α, it has the following properties:

$$H_n\{\alpha x_1(\cdot)\} = \alpha H_n\ \{x_1(\cdot)\}$$

$$H_n\ \{x_1(\cdot) + x_2(\cdot)\} = H_n\{x_1(\cdot)\} + H_n\{x_2(\cdot)\}$$

Therefore, if the filter is linear and its coefficients are time invariant, it is called Linear and Time Invariant (LTI) Filter. In mathematics terms, we represent such general LTI filter by following formulation:

$$y_n = \sum_{k=0}^{M} \alpha_k x_{n-k} + \sum_{j=0}^{N} \beta_j y_{n-j} \qquad (11)$$

It is worth emphasizing that the filter response will be completely defined once one has found the coefficients α_k and β_j. This general equation shows that the filter produces each new output sample value from the current and M previous input samples, and optionally from its own N previous output values. There are a number of classifications for linear filters. Any general linear filter is called causal (i.e., physically realizable, as shown in Equation 11) when its output in a given time depends only on inputs at past and present time [170] [77]. In other words, its output does not depend on any future inputs. On the other hand, it is called non-causal when there is some sort of dependency in its output on later inputs.

The simplest example of a non-casual filter is presented in Equation 14:

$$y_n = x_{n+1} \tag{12}$$

It is a non-causal filter because the output depends on a sample input that will only appear one sample into the future. Linear filters are also fully characterized according to their response to a unit impulse signal, which consists of a single sample at time zero having unitary amplitude, preceded and followed by zeros. If the response to such impulse signal is finite, the filter is then called Finite Impulse Response (FIR), otherwise is called Infinite Impulse Response (IIR) filter. FIR filters have some important properties, which make them preferable over IIR filters. For instance, FIR filters are inherently stable, since all the poles are located at the origin, i.e. within the unit circle. In addition, they require no feedback since errors are not compounded by summed iterations [141] [242].

Figure 142 shows the signal flow representation for a general FIR filter (also known as transversal filter), where z^{-1} represents a delay.

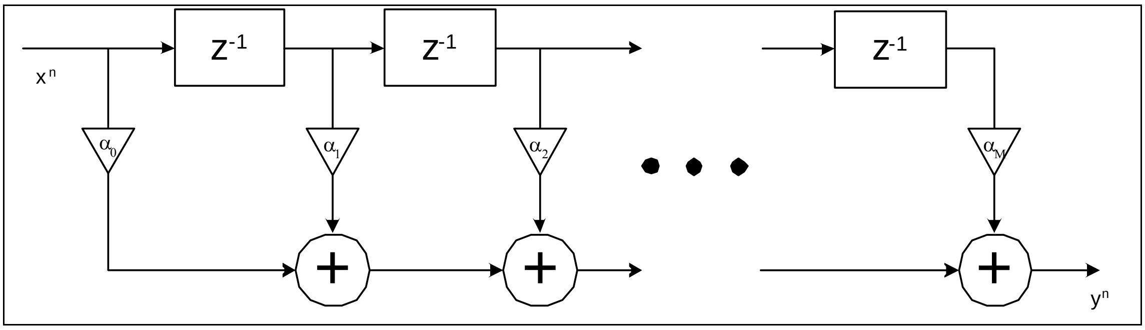

Figure 142 - A general causal FIR filter

Based on these definitions, one should be careful when using filters in time domain. Restriction to causal filters is to a certain extent ordinary, especially when the filter must deal with real time applications. In such a case, one normally wishes processing a continuous data stream and yielding output-filtered values at the same rate as the arrival unprocessed data rate. In the scope of this thesis, where available bandwidth information arrives in real time, such physical feasibility is indeed an existent constraint. At the time being, we restrict all algorithms for the DLPF unit in our architecture to the causal case.

In [207], Smith presents a convolution representation for FIR Filters. Considering that the output of the i_{th} delay element in Figure 142 is $x_{n-i}, i = 0, \ldots M$, where x_n is the input signal amplitude at time n. The output signal y_n is

$$y_n = \alpha_0 x_n + \alpha_1 x_{n-1} + \cdots + \alpha_M x_{n-M} \tag{13}$$

$$y_n = \sum_{m=0}^{M} \alpha_m x_{n-m} = \sum_{m=0}^{M} h_m x_{n-m} \tag{14}$$

$$y_n \equiv (h * x)_n, \tag{15}$$

where $*$ is the convolution operator. In general, a FIR filter convolves any input signal, x_n , with the filter's transfer function or impulse response, h_n.

Please note that if each element in the weight vector in Equation 16 declines exponentially at distant lags, the resulting causal FIR filter is the well-known EWMA smoothing technique.

Appendix 2 - Digital Video and the MPEG-4 FGS standard

Digital video comprises of video frames displayed at a given frame rate (e.g., 30 frames/sec in NTSC standard, 25 frames/sec in PAL standard). The ITU-R/CCIR-601 format has 720 × 480 pixels, where each pixel consists of three components: the luminance component (Y), and the two chrominance components, hue (U) and intensity (V). Besides reducing storage space requirements, one goal of video compression techniques (e.g., MPEG, H.264) is to take a raw video signal and packetize it in order to transport over a network. In this appendix, we give a brief overview of the main principles of scalable and non-scalable video compression. We focus on the fundamental techniques of MPEG-4 encoding as well as its advanced approach, namely MPEG-4 FGS. We refer the interested reader to the references [122] [156] [205] for more details. MPEG-4 (ISO14496) is an ISO/IEC standard developed by MPEG (Moving Picture Experts Group). In [122], the author pointed out that since applications can stream the same multimedia content compressed at low bit rates (e.g., on the order of tens of Kbps) or at high bit rates (e.g., on the order of tens of Mbps), MPEG-4 became the preferable choice for video streaming over the Internet. In addition, due to its inherent scalability, MPEG-4 the same content could be streamed to different devices over a variety of networks.

One main feature in MPEG compression is inter-frame coding using motion estimation and compensation between consecutive video frames. MPEG standard has the following frame types: intra-coded (I), inter-coded (P), and bidirectional coded (B). These frames are grouped into Groups of Pictures (GoPs). In other words, the frame sequence from an I frame up to the next I frame (excluded) is referred to as one GoP. MPEG-4 standard also refers GOP to as Group of Video Object Plane (GOV). In addition, the way of how frames are organized to compose a GoP is referred to as GoP pattern (structure). In general, two parameters define a GOP (N, M). N represents the distance between two I-frames. M is the distance between two anchor frames (I or P frame).

Details of how to encode different frame types are out of scope of this appendix. However, it is worth presenting some basic information. An I frame are always intra-coded, i.e. it carries as much information as possible for a given video frame). A P frame is inter-coded with reference to the preceding I or P frame. A B frame is inter-coded with reference to the preceding I or P frame.

Figure 143 illustrates a typical GoP pattern with three P frames between I frames and two B frames between P frames [199]. For P frames, the dashed arrows represent forward reference to the preceding I or P. For B frames, the dashed arrows represent forward reference to the preceding I or P frames whereas the solid arrows represent the backward reference to the succeeding I and P frames.

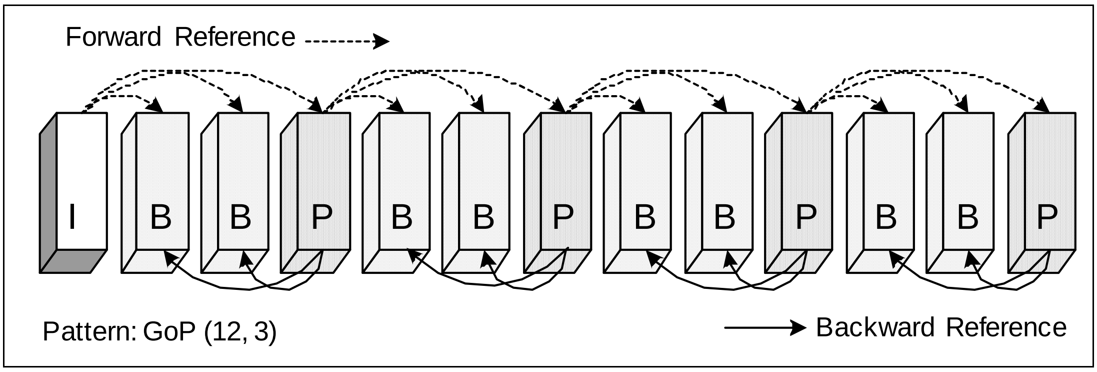

Figure 143 - Typical MPEG Group of Pictures (GoP) - From Reference [199]

Scalable Video Encoding

Under conventional layered encoding, the raw video is encoded hierarchically into a base layer (BL) and one (or more) enhancement layer(s) (EL). The BL present a basic video quality, whereas the BL+EL combination provides an improvement in video quality. MPEG standard has some scalability modes, among them the temporal and spatial ones. The temporal scalable encoding interleaves EL frames between BL frames. Figure 144 presents an example of temporal scalable encoding where the BL consists of I and P and the EL consists of B frames only [199]. Please note that within the temporal scalable encoding the insertion of the EL increases the frame rate for the overall video structure. On the other hand, the receiver can decode the BL independently of the EL.

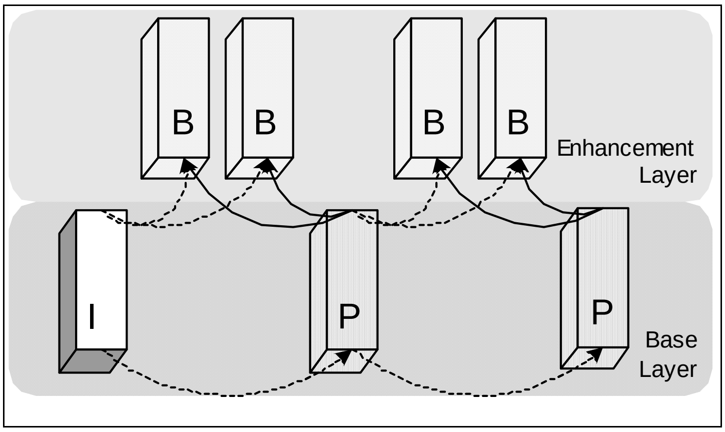

Figure 144 - Example of Temporal Scalable Encoding (From Reference [199])

Under spatial scalable encoding, the encoder downsample the raw video into a smaller BL format, thus generating I and P frames. The EL consists of P and B frames. P frames in the EL are encoded with reference to the corresponding I frames in the BL, whereas B frames in the EL are encoded with reference to the corresponding P frame in the BL and the preceding P frame in the EL, as Figure 145 illustrates [199].

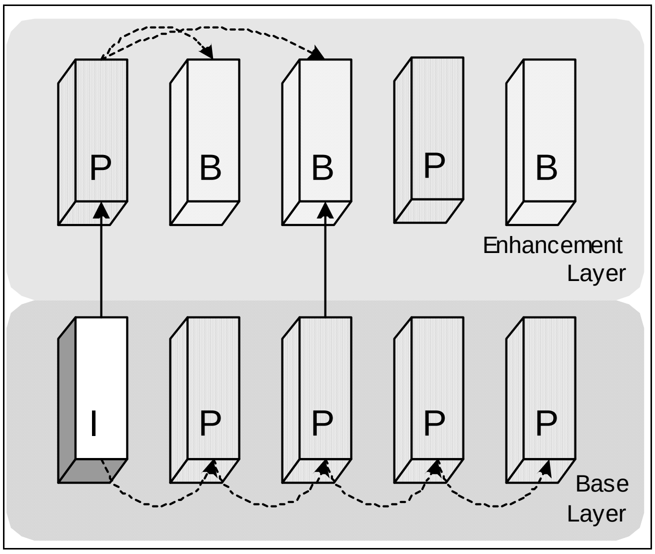

Figure 145 - Example of Spatial Scalable Encoding (From Reference [199])

The MPEG-4 Fine-Granular-Scalability (FGS) is a new approach adopted by the ISO MPEG-4 video standard as the core video-coding method for MPEG-4 streaming applications [139] [175]. It has been proposed [173] [178] to help handling the variability in bandwidth between end-systems over the Internet. Within the FGS approach, a hybrid scalability structure was developed, where it enables quality (SNR), temporal, or both temporal-SNR scalable video coding and streaming [177]. FGS has as its main goal an improved flexibility in video streaming. Video servers can take advantage of such flexibility by adapting the streamed video (i.e., the sending rate) to the available bandwidth in real time, with no need to re-encoding.

MPEG-4 FGS VIDEO CODING METHOD

MPEG-4 FGS framework covers a given bandwidth range while maintaining a simple scalability structure [177]. As in the original MPEG-4, the FGS structure consists of two layers, namely a BL coded at a bitrate R_{BL}, and a single EL coded using a fine-granular scheme to a maximum bitrate of $R_{\max}$ (see Figure 146).

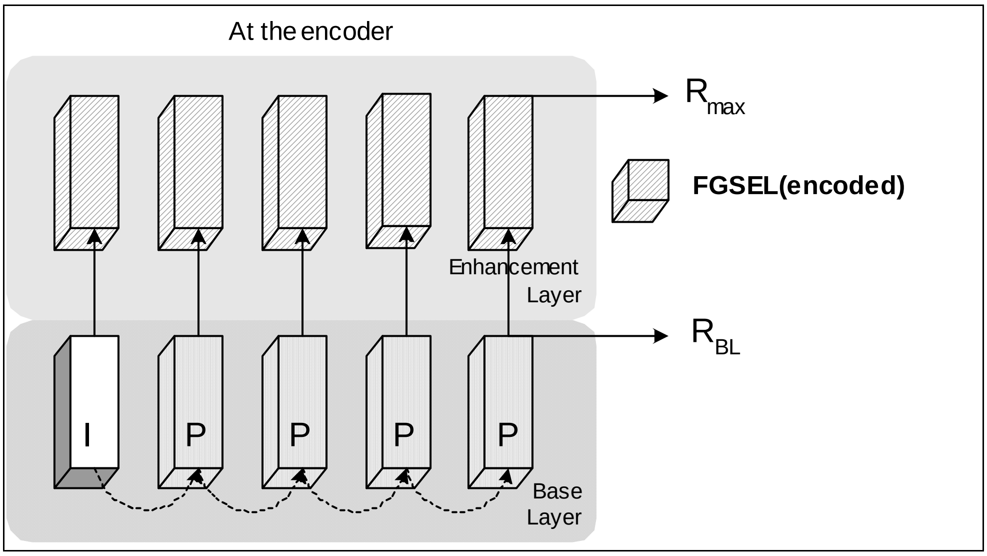

Figure 146 - FGS structure at the encoder

One requirement for MPEG-4 FGS is that the available bandwidth should be higher than R_{BL} most of the time during the streaming session, since server- and receiver-side buffering could deal with sporadic shortage of network resources. Please note that there are two main components involved in this process: the encoder and the streaming server. The encoder only needs to know the target bitrate range whereas the streaming server decides which portion of any EL frame (along with the corresponding BL frame). In other words, FGS EL can be truncated anywhere at the granularity of bits (R_{EL}), thus allowing a precise adaptation to changing network resources.

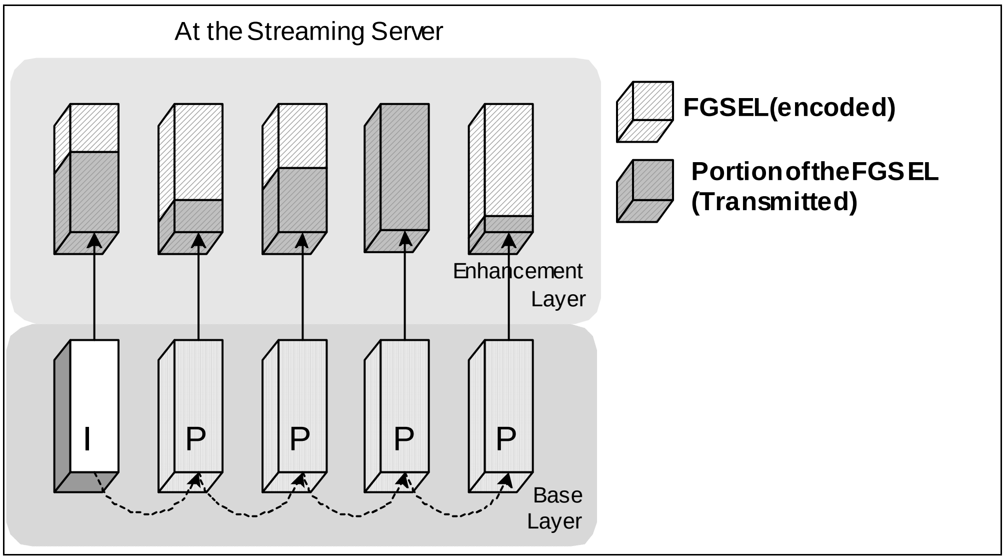

Figure 147 - FGS structure at the streaming server

At the streaming server, the addition of EL improves upon the base-layer video, fully utilizing the available bandwidth at transmission-time (Figure 147). At the receiver side, the decoder decompresses the BL and the received portion of the EL (Figure 148). Bear in mind that with conventional layered coding, the video stream can only adapt at the granularity of complete enhancement layers.

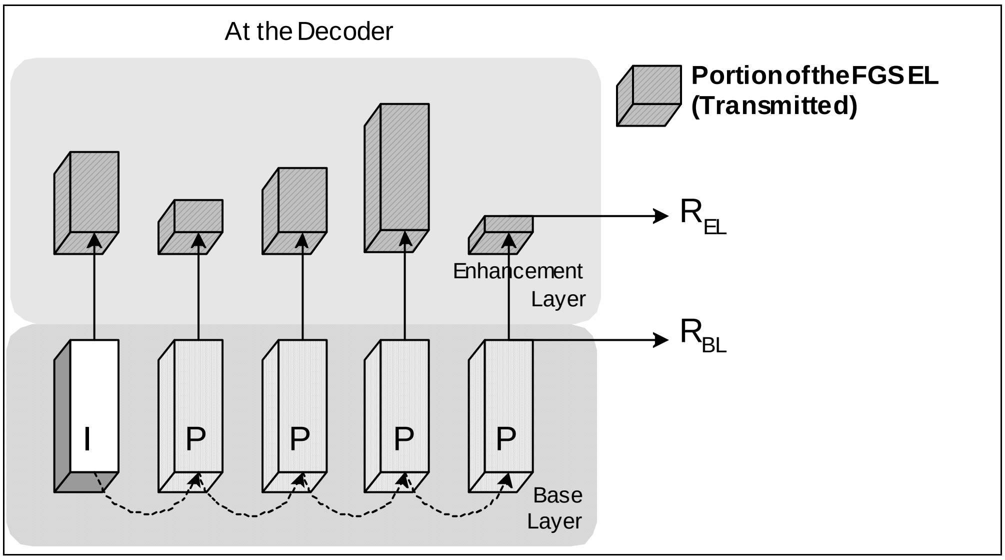

Figure 148 - FGS structure at the decoder (expected)

References

[1] Abry, P. and Veitch, D., “Wavelet Analysis of Long-Range Dependent Traffic”, IEEE Transactions on Information Theory, 44(1):2{15, Jan. 1998.

[2] Abry, P., Baraniuk, R., Flandlin, P., Riedi, R., and Veitch, D., “Multiscale Nature of Network Traffic”, IEEE Signal Processing Magazine, 19(3):28-46, May 2002.

[3] Addie, R.G., Neame, T.D. & Zukerman, M., “Performance Evaluation of a Queue Fed by a Poisson Pareto Burst Process”, Computer Networks, Vol. 40, Nº 3, p. 377-397, Oct. 2002.

[4] Ahmed, T., Mehaoua, A., Boutaba, R., and Iraqi, Y., “Adaptive Packet Video Streaming Over IP Networks: A Cross-Layer Approach”, IEEE Journal on Selected Areas in Communications, Vol. 23, No. 2, February 2005, pp. 385 – 401.

[5] Akamai, “Best Practices In Digital Media Delivery”, White Paper, 2004.

[6] Allman, M., Paxson, P., and Stevens, W., “TCP Congestion Control,” IETF RFC 2581, April 1999.

[7] Almeida, J. M., Eager, D. L., Ferris, M., and Vernon, M. K., "Provisioning Content Distribution Networks for Streaming Media", Proceedings of the IEEE 21st INFOCOM 2002, New York, NY, June 2002.

[8] Almeida, J. M., Eager, D. L., Vernon, M. K., and Wright, S., "Minimizing Delivery Cost in Scalable Streaming Content Distribution Systems", IEEE Transactions on Multimedia, Vol. 6, No. 2, Special Issue on Streaming Media, pp. 356-365, April 2004.

[9] Altman, E., Avrachenkov, K., Barakat, C., "A Stochastic Model of TCP/IP with Stationary Random Losses", in IEEE/ACM Transactions on Networking, vol. 13, no. 2, pp. 356- 369, April 2005.

[10] Athuraliya, S.; Low, S.H.; Li, V.H.; Qinghe Yin, “REM: Active Queue Management”, Network, IEEE, Volume 15, Issue 3, Page(s):48 – 53, May-June 2001.

[11] Atkin, B. and Birman, K. P., “Evaluation of an Adaptive Transport Protocol”, INFOCOM 2003, Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, 30 March-3 April 2003, San Francisco, USA..

[12] Bacelar, R., Callado, A., Kamienski, C., Fernandes, S., Mariz, D., Kelner, J., and Sadok, D., “Performance Analysis of VoIP P2P Applications”, in Portuguese, SBRC 2006 - 24th Brazilian Symposium on Computer Networks, May 2006.

[13] Balk, A., Gerla, M., Maggiorini, D. and Sanadidi, M., “Adaptive Video Streaming: Pre-encoded MPEG-4 with Bandwidth Scaling”, Journal of Computer Networks, Vol 44 (2004) 415–439.

[14] Balk, A., Gerla, M., Maggiorini, D., and Sanadidi, M., “Adaptive Video Streaming: Pre-Encoded MPEG-4 with Bandwidth Scaling”, Computer Networks 44 (2004) 415–439.

[15] Balk, A., Maggiorini, D. Gerla, M., and Sanadidi, M., “Adaptive MPEG-4 Video Streaming with Bandwidth Estimation, Proceedings of QOS-IP 2003, Milano, Italy, February 2003.

[16] Bansal, D., Balakrishna, H., Floyd, S., and Schenker, S., “Dynamic Behavior of Slowly-Responsive Congestion Control Algorithms”, Proceedings of the ACM SIGCOMM 2001, San Diego, USA, August 2001.

[17] Barakat, C., Thiran, P., Iannaccone, G., Diot, C., Owezarski, P., "Modeling Internet Backbone Traffic at the Flow Level", IEEE Transactions on Signal Processing - Special Issue on Signal Processing in Networking, vol. 51, no. 8, pp. 2111-2124, August 2003.

[18] Barbera, M.; Licandro, F.; Lombardo, A.; Schembra, G.; Tusa, G., “DARED: a double-feedback AQM technique for routers supporting real-time multimedia traffic in a best-effort scenario”, Control, Communications and Signal Processing, 2004. First International Symposium on, 2004, Pages: 365- 368.

[19] Beran, J. “Statistics for Long-Memory Processes”, Chapman & Hall/CRC, 1^{a} ed., New York, 1994.

[20] Bormans, J.; Gelissen, J.; Perkis, A.; “MPEG-21: The 21st Century Multimedia Framework, Signal Processing Magazine, IEEE, Volume 20, Issue 2, Page(s):53 – 62, March 2003.

[21] Bouras, Ch., Gkamas, A., “Streaming multimedia data with adaptive QoS characteristics”, Protocols for Multimedia Systems 2000, Cracow, Poland, October 2000, pp. 129-139.

[22] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J. and L. Zhang, “Recommendations on Queue Management and Congestion Avoidance in the Internet”, RFC2309, April 1998.

[23] Brakmo, L., and Peterson, L., “TCP Vegas: End to End Congestion Avoidance on a Global Internet,” IEEE Journal on Selected Areas in Communications, vol. 13, no. 8, October 1995, 1465-1480.

[24] Burnett, I., Van de Walle, R., Hill, K., Bormans, and J., Pereira, F., “MPEG-21: Goals and Achievements”, Multimedia, IEEE, Volume 10, Issue 4, Page(s):60 – 70, Oct-Dec 2003

[25] Cai, L., Shen, X., Pan, J. and Mark, J.W., "Performance Analysis of TCP-Friendly AIMD Algorithms for Multimedia Applications'', IEEE Trans. On Multimedia, vol. 7, issue 2, pp. 339-355, 2005.

[26] Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A., "Graphical Methods for Data Analysis", Wadsworth & Brooks Cole, 1983.

[27] Chan, S.-P., Kok, C.-W., and Wong, A. K., "Multimedia Streaming Gateway With Jitter Detection", IEEE Transactions On Multimedia, Vol. 7, No. 3, June 2005.

[28] Chen, L., Low, S. H., Chiang, M., and Doyle, J. C., "Cross-layer Congestion Control, Routing and Scheduling Design in Ad Hoc Wireless Networks", Proceedings of the 25th IEEE INFOCOM, Barcelona, 2006.

[29] Chen, M. and Zakhor, A., "Rate Control for Streaming Video over Wireless", Proceedings of 23rd IEEE INFOCOM'04, Hong Kong, March 7-11, 2004.

[30] Chen, S., Shen, B., Wee, S., and Zhang, X., "Designs of high quality streaming systems", Proceedings of IEEE INFOCOM'04, Hong Kong, March 7-11, 2004.

[31] Chen, S., Wang, H., and Shen, B., "Segment-based Proxy Caching for Internet Streaming Media Delivery," IEEE Multimedia Magazine, September 2005.

[32] Chen, Zhenzhong, Ngana, King N., and Zhao, Chengji, "Improved Rate Control for MPEG-4 Video Transport Over Wireless Channel", Signal Processing: Image Communication 18 (2003) 879–887.

[33] Cheung, Gene Tan, Wai-Tian and Yoshimura, Takeshi, "Double Feedback Streaming Agent for Real-Time Delivery of Media Over 3G Wireless Networks", IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004.

[34] Cheung, Gene Tan, Wai-Tian and Yoshimura, Takeshi, "Real-Time Video Transport Optimization Using Streaming Agent Over 3G Wireless Networks", IEEE Transactions on Multimedia, Vol. 7, No. 4, August 2005.

[35] Chou, C.-T., and Shin, K. G. "Analysis of Combined Adaptive Bandwidth Allocation and Admission Control in Wireless Networks", Proceedings of 21st IEEE INFOCOM 2002, New York, June 23-27, 2002.

[36] Chung, J. W., "Congestion Control for Streaming Media", Ph.D. Thesis, Worcester Polytechnic Institute, October 2005.

[37] Cleveland, W. S., "LOWESS: A Program for Smoothing Scatterplots by Robust Locally Weighted Regression", The American Statistician, 35, 54, 1981.

[38] Cleveland, W. S., "Robust Locally Weighted Regression and Smoothing Scatterplots". J. Amer. Statist. Assoc., 74, 829–836, 1979.

[39] Consumer Electronics Association, “Video over IP”, Digital America Magazine, http://www.ce.org/publications/books_references/digital_america/video/video_over_ip.asp , last access in September 2005.

[40] Cranley, N., Murphy, L., Perry, P., “User-Perceived Quality-Aware Adaptive Delivery of MPEG-4 Content”, Proceedings of the ACM NOSSDAV 2003, June 1-3, 2003, Monterey, California, USA.

[41] Crovella, M. E. & Bestravos, A., “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes”, IEEE Trans. Networking, 5(6), Dec. 1997.

[42] Cui, Y. and Nahrstedt, K., “Layered Peer-to-Peer Streaming”, Proceedings of the ACM NOSSDAV’03, June 1–3, 2003, Monterey, California, USA.

[43] Dai, M., Loguinov, D., and Radha, H., “A Hybrid Wavelet Framework for Modeling VBR Video Traffic”, IEEE International Conference on Image Processing (ICIP), October 2004.

[44] de Cuetos, P. and Ross, K.W., “Adaptive rate control for streaming stored fine-grained scalable video”, Proceedings of The 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), Pages: 3 – 12, Miami, Florida, USA, 2002.

[45] De Cuetos, P., and Ross, K., “Unified Framework for Optimal Video Streaming”, Proceedings of the 23rd IEEE INFOCOM, Hong Kong, March 2004.

[46] de Cuetos, P., Reisslein, M., and Ross, K., “Evaluating the Streaming of FGS–Encoded Video with Rate–Distortion Traces”, Institut Eurécom Technical Report, RR–03–078, June 2004.

[47] de Cuetos, P., Seeling, P., Reisslein, M., Ross, K. W., “Comparing the Streaming of FGS Encoded Video at Different Aggregation Levels: Frame, GoP, and Scene”, International Journal of Communication Systems, Vol. 18, No. 5, pp. 449-464, 2005

[48] Dennis, J.D., “A Performance Test of a Run-Based Adaptive Exponential Smoothing” Production and Inventory Management 19: 43–46, 1978.

[49] Dias, Kelvin L., Fernandes, Stenio F. L., Sadok, Djamel F. H., “A Call Admission Control Scheme for Next Generation Wireless Networks”, Telecommunications Magazine, Brazilian Institute of Telecommunications, 2004.

[50] Dias, Kelvin L., Fernandes, Stenio F. L., Sadok, Djamel F. H., “Predictive Call Admission Control for All-IP Wireless and Mobile Networks”, IFIP/ACM Latin America Networking Conference (LANC 2003), La Paz, Bolivia October 3 – 5, 2003.

[51] Downey, A. B., "Evidence for Long-Tailed Distributions in the Internet", ACM SIGCOMM IMW 2001, Nov. 2001.

[52] Duffield, N. G., Ramakrishnan, K.K., and Reibman, A. R., "SAVE: An Algorithm for Smoothed Adaptive Video Over Explicit Rate Networks", Networking, IEEE/ACM Transactions on, Vol.6, Iss.6, Oct 1998, Page(s): 717-728.

[53] Falk, A., and Katabi, D., "Specification for the Explicit Control Protocol (XCP)", IETF, draft-falk-xcp-00.txt (work in progress), October 2004.

[54] Fernandes, S. F. L., et. al., "Estimating Properties of Flow Statistics using Bootstrap", 12th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Netherlands, 2004.

[55] Fernandes, Stenio F. L.; Kamienski, Carlos A.; Sadok, Djamel F. H., "Accurate and Fast Replication on the Generation of Fractal Network Traffic Using Alternative Probability Models", Conference on Performance and Control of Next Generation Communication Networks, Part of the SPIE International Symposium ITCom 2003, 7-11 September 2003 in Orlando, FL, USA.

[56] Floyd, S., "HighSpeed TCP for Large Congestion Windows", RFC 3649, Experimental, December 2003.

[57] Floyd, S., "HighSpeed TCP for Large Congestion Windows", RFC 3649, Experimental, December 2003.

[58] Floyd, S., and Henderson, T., "The NewReno Modifications to TCP's Fast Recovery Algorithm", IETF RFC 2582, April, 1999.

[59] Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", In IEEE/ACM Transactions on Networking, 1(4):397–413, Aug. 1993.

[60] Floyd, S., Handley, M., Padhye, J. and Widmer, J. "Equation-Based Congestion Control for Unicast Applications", In Proceedings of the ACM SIGCOMM, 2000.

[61] Floyd, S., Handley, M., Padhye, J., and Widmer, J., "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 3448, Proposed Standard, January 2003.

[62] Fonseca, N. L. S., Mayor, G. S. & Neto, C. A. V., "On the Equivalent Bandwidth of Self-Similar Sources", ACM Trans. Modeling and Computer Simulation, 10(2), p. 104-124, 2000.

[63] Futemma, S., Yamane, K., Itakura, E., "TFRC-based Rate Control Scheme for Real-time JPEG 2000 Video Transmission", Proc. of the IEEE Consumer Communications & Networking Conference – CCNC 2005, Las Vegas, Nevada, USA.

[64] Gaivoronski, A., “Implementation of Stochastic Quasigradient Methods”, in Numerical Techniques for Stochastic Optimization, Springer Series in Computational Mathematics, Springer-Verlag, Berlin, pages 313-352, 1988.

[65] Galluccio, L., Licandro, F., Morabito, G., and Schembra, G., “An Analytical Framework for the Design of Intelligent Algorithms for Adaptive-Rate MPEG Video Encoding in Next-Generation Time-Varying Wireless Networks”, Selected Areas in Communications, IEEE Journal on, Volume 23, Issue 2, Page(s):369 – 384, Feb 2005.

[66] Gan, T., Ma, K-K., and Zhang, L., “Dual-Plan Bandwidth Smoothing for Layer-Encoded Video”, IEEE Transactions on Multimedia, Vol. 7, No. 2, April 2005.

[67] Gençai, R., Selçuk, F., and Whitcher, B., “An Introduction to Wavelets and Other Filtering Methods in Finance and Economics”, Academic Press, San Diego, USA, 2002.

[68] George, A. and W. B. Powell, “Adaptive Stepsizes for Recursive Estimation with Applications in Approximate Dynamic Programming”, Techincal Report, CASTLE Laboratory, Department of Operations Research and Financial Engineering, Princeton University, USA, January 24, 2005.

[69] Gotz, D. and Mayer-Patel, K., “A General Framework for Multidimensional Adaptation”, Proceedings of the 12th annual ACM international conference on Multimedia, October Pages: 612 - 619, New York, USA, 2004.

[70] Graps, A., “An Introduction to Wavelets”, in IEEE Computational Science and Engineering, Vol. 2, Num. 2, IEEE Computer Society, 1995.

[71] Green, P. J. and Silverman, B. W., “Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach”, Chapman and Hall, 1994.

[72] Grieco, L. A. and Mascolo, S., “Adaptive Rate Control for Streaming Flows over the Internet”, ACM Multimedia Systems Journal, 9(6):517–532, June 2004.

[73] Grossglauser, M., Bolot, J.C., “On the Relevance of Long-Range Dependence in Network Traffic”, IEEE Trans. Network, Vol. 7, p.629-640, Oct. 1999.

[74] Grossglauser, M.; Keshav, S.; Tse, D.N.C., “RCBR: a simple and efficient service for multiple time-scale traffic”, Networking, IEEE/ACM Transactions on, Vol.5, Iss.6, Dec 1997, Pages:741-755.

[75] Guo, M., and Ammar, M. H., “Scalable Live Video Streaming To Cooperative Clients Using Time Shifting And Video Patching”, Proceedings of the IEEE INFOCOM 2004, March 2004.

[76] Hamilton, J. D., “Time Series Analysis”, Princeton University Press, ISBN: 0691042896, Princenton, NJ, 1994.

[77] Hamming, R.W., “Digital Filters”, 2nd ed., Englewood Cliffs, NJ, Prentice-Hall, 1983.

[78] Handley, M., Floyd, S., Pahdye, J., and Widmer, J. “TCP Friendly Rate Control (TFRC)”, Protocol Specification. RFC 3448, Proposed Standard, January 2003.

[79] Haratcherev, I., Taal, J., Langendoen, K., Lagendijk, R., and Sips, H., “Optimized Video Streaming over 802.11 by Cross-Layer Signaling”, IEEE Communication Magazine, vol. 44, no. 1, Jan. 2006, pp. 115-121.

[80] Harvey, A.C., “Time Series Models”, The MIT Press; 2nd edition, Massachussets, MA, ISBN: 0262082241, 1993.

[81] Hassan, M., and Jain, R., "High Performance TCP/IP Networking: Concepts, Issues, and Solutions,"Prentice-Hall, 2003.

[82] Hastie, T. J. and Tibshirani, R. J., “Generalized Additive Models”, Chapman and Hall, 1990.

[83] He, G. and Hou, J. C., “On Exploiting Long Range Dependence of Network Traffic in Measuring Cross Traffic on an End-to-end Basis”, Proceedings of the 22nd IEEE INFOCOM, San Francisco, 2003.

[84] Hollot, C., Misra, V., Towsley, D., and Gong, W.-B. “A Control Theoretic Analysis of RED,” in Proceedings of the 20th IEEE INFOCOM, Apr. 2001, pp. 1510-1519.

[85] Horn, U., Stuhlmuller, K., Link, M., and Girod, B., “Robust Internet Video Transmission Based on Scalable Coding and Unequal Error Protection”, Signal Processing: Image Communication, vol. 15, pp. 77–94, 1999.

[86] Huang, J., Krasic, C., Walpole, J., and Feng, W.-C., "Adaptive Live Video Streaming by Priority Drop", IEEE International Conference on Advanced Video and Signal Based Surveillance (IEEE AVSS), July 2003.

[87] Huang, P., Feldmann, A., Willinger, W., “A Non-Intrusive, Wavelet-Based Approach to Detecting Network Performance Problems”, Proceeding of ACM SIGCOMM Internet Measurement Workshop 2001, San Francisco, November 2001.

[88] Hussain, I., and Bains, K., “An Explicit Rate ABR Algorithm for New-generation ATM Switches”, International Journal of Network Management, Vol. 9, 323 – 338 (1999)

[89] In-Stat, Multimedia Market Alert, http://www.instat.com/newmk.asp?ID=1430, September 2005.

[90] ISO/IEC 21000-7:2004, “Information Technology - Multimedia Framework - Part 7: Digital Item Adaptation”, 2004.

[91] Jacobson, V., “Congestion avoidance and control”, In Symposium Proceedings on Communications Architectures and Protocols, Stanford, California, United States, August 16 - 18, SIGCOMM '88, Ed. ACM Press, New York, NY, 314-329, 1988.

[92] Jain, R., “The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling”, John Wiley & Sons, Inc., 1991.

[93] Jain, R., Chiu, D., and Hawe, W., "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems", DEC Research Report TR-301, September 1984.

[94] Jeong, H.-D. J., McNickle, D. & Pawlikowski., K. “Fast Self-Similar Teletraffic Generation Based on FGN and Wavelets”. IEEE ICON’99, Sept. 1999.

[95] Jiang, H. and Dovrolis, C., “Why is the Internet Traffic Bursty in Short Time Scales?”, In Proceedings of the ACM SIGMETRICS’05, June 6–10, 2005, Banff, Alberta, Canada.

[96] Jiang, H., Dovrolis, C., “Why is the Internet Traffic Bursty in Short Time Scales?” ACM SIGMETRICS Performance Evaluation Review, v.33 n.1, June 2005.

[97] Jiang, J. and Papavassiliou, S., “Enhancing Network Traffic Prediction and Anomaly Detection via Statistical Network Traffic Separation and Combination Strategies”, Computer Communications, 2005, pp. 1–12.

[98] Jin, C., Wei, D. X., and Low, S. H., “FAST TCP: Motivation, Architecture, Algorithms, Performance”, In Proceedings of the 23rd IEEE INFOCOM, Hong Kong, March 2004.

[99] Jin, C., Wei, D. X., and Low, S. H., “Modeling and Stability of FAST TCP”, In Proceedings of the 24th IEEE INFOCOM, Miami, FL, March 2005

[100] Jin, C., Wei, D. X., Low, S. H., Buhrmaster, G., Bunn, J., Choe, D. H., Cottrell, R. L. A., Doyle, J. C., Feng, W., Martin, O., Newman, H., Paganini, F., Ravot, S., and Singh, S., “FAST TCP: From Theory to Experiments”, IEEE Network, 19(1):4-11, January/February 2005.

[101] Johari, R., and Tan, D., “End-to-End Congestion Control for the Internet: Delays and Stability,” IEEE/ACM Transactions on Networking, vol. 9, no. 6, December 2001.

[102] Kalampoukas, L., Varma, A., Ramakrishnan, K.K. “Explicit Window Adaptation: A Method to Enhance TCP Performance”, INFOCOM ’98, 17th Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings of the IEEE, Vol.1, Mar-2 Apr 1998, Pages:242-251 vol.1., 1998.

[103] Kamienski, C. A., et al., “Simulating the Internet: Applications in Research and Education”, XXI JAI’2002 (in Portuguese), Jul. 2002.

[104] Kang, S.-R., Zhang, Y., Dai, M., and Loguinov, D., "Multi-layer Active Queue Management and Congestion Control for Scalable Video Streaming", Proceedings of the IEEE 24th International Conference on Distributed Computing Systems, Tokyo, Japan, March 2004.

[105] Kang, S-R., Zhang, Y., Dai, M. and Loguinov, D., "Multi-layer Active Queue Management and Congestion Control for Scalable Video Streaming", Proceedings of the IEEE International Conference Distributed Computing Systems, March 2004.

[106] Karnik, A. and Kumar, A., "Performance of TCP Congestion Control With Explicit Rate Feedback", Networking, IEEE/ACM Transactions on, Volume: 13 , Issue: 1, Feb. 2005, page(s): 108 – 120.

[107] Katabi, D., "Decoupling Congestion Control and Bandwidth Allocation Policy With Application to High Bandwidth-Delay Product Networks", PhD Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (MIT), March 2003.

[108] Katabi, D., "XCP's Performance in the Presence of Malicious Flows", 2nd International Workshop on Protocols for Fast Long-Distance Networks, February 2004.

[109] Katabi, D., Handley, M. and C. Rohr, "Internet Congestion Control for Future High Bandwidth-Delay Product Environments", ACM Computer Communication Review Proceedings of the SIGCOMM 2002 Symposium, August 2002.

[110] Katabi, Dina & Falk, Aaron, "Specification for the Explicit Control Protocol (XCP)", Work-in-Progress, Network Working Group, Internet-Draft, Expires: April 17, 2005.

[111] Kaufhold, Gerry, "The Four Golden Rules of Digital Video Distribution", *White Paper Based on In-Stat's Industry Leading Analysis,* September 2005, Available at http://www.in-stat.com.

[112] Kelly, C. T., "Engineering Flow Controls for the Internet", Ph.D. Thesis, University of Cambridge, 2004.

[113] Kelly, F.P., Maulloo, A.K. and Tan, D.K.H., "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," Journal of the Operational Research Society, 49, 1998.

[114] Kelly, T., "Scalable TCP: Improving Performance in High Speed Wide Area Networks", SIGCOMM Computer Communication Review 33, 2 (Apr. 2003), 83-91.

[115] Kesten, H., "Accelerated Stochastic Approximation", Annals of Mathematical Statistics 29:41-59, 1958.

[116] Khan, S., Peng, Y., Steinbach, E., Sgroi, M., and Kellerer, W., "Application-Driven Cross-Layer Optimization for Video Streaming over Wireless Networks", IEEE Communication Magazine, vol. 44, no. 1, Jan. 2006, pp. 122-130.

[117] Kim, M.S., Kim, T., Shin, YJ, Lam, S. S., Powers, E. J., "A Wavelet-Based Approach to Detect Shared Congestion", SIGCOMM Comput. Commun. Rev. 34, 4 (Aug. 2004), 293-306.

[118] Kim, T. and Ammar, M.H.," Optimal quality adaptation for scalable encoded video", Selected Areas in Communications, IEEE Journal on, Vol.23, Iss.2, Feb. 2005, Pages: 344- 356.

[119] Kim, Y. G., Kim, J., and Kuo, C. C. J., "TCP-Friendly Internet Video Smooth and Fast Rate Adaptation and Network-Aware Error Control", IEEE Transactions on Circuits and Systems for Video Technology, February 2004.

[120] Kim; Y-G., Kim; J-W., Kuo, C.-C.J., "TCP-friendly Internet video with smooth and fast rate adaptation and network-aware error control", Circuits and Systems for Video Technology, IEEE Transactions on, Vol.14, Iss.2, Feb. 2004, Pages: 256- 268

[121] Klaue, J., Rathke, B., and Wolisz, A., "EvalVid – A Framework for Video Transmission and Quality Evaluation", Proceedings of the 13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, USA, Sept. 2003.

[122] Koenen, R., "MPEG-4 Overview - (V.21 – Jeju Version)", ISO/IEC JTC1/SC29/WG11 N4668, March 2002.

[123] Krasic, C. and Walpole, J., "Priority-Progress Streaming for Quality-Adaptive Multimedia", ACM Multimedia Doctoral Symposium, Ottawa, Canada, October 2001.

[124] Krasic, C., Walpole, J., and Feng, W.-C., "Quality-Adaptive Media Streaming by Priority Drop", 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2003), June 2003.

[125] Ksentini, A., Naimi, M., and Guéroui, A., "Toward an Improvement of H.264 Video Transmission over IEEE 802.11e through a Cross-Layer Architecture", IEEE Communications Magazine • January 2006, pp 107-114.

[126] Kumazoe, K., Kouyama, K., Hori, Y., Tsuru, M., and Oie, Y., "Can High-Speed Transport Protocols be Deployed on the Internet?: Evaluation Through Experiments on JGNII", The 4th International Workshop on Protocols for Fast Long-Distance Networks (PFLDNet 2006), February 2nd and 3rd, Nara, Japan, 2006.

[127] Kunniyur, S., and Srikant, R., “Analysis and Design of an Adaptive Virtual Queue Algorithm for Active Queue Management”, IEEE ACM Transactions on Networking, pp. 286-299, April 2004.

[128] Kunniyur, S., and Srikant, R., “End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN marks,” Proceedings of IEEE INFOCOM, March 2000.

[129] Kuzmanovic, A., “The Power of Explicit Congestion Notification”, In Proceedings of the SIGCOMM’05, August 21–26, 2005, Philadelphia, Pennsylvania, USA.

[130] Lakshman, T.V., Mishra, P.P., Ramakrishnan, K.K., “Transporting compressed video over ATM Networks with Explicit-Rate Feedback Control”, Networking, IEEE/ACM Transactions on, Vol.7, Iss.5, Oct 1999, Page(s): 710-723.

[131] Lakshman, T.V., Ortega, A., and Reibman, A.R., “VBR video: tradeoffs and potentials”, Proceedings of the IEEE Volume 86, Issue 5, May 1998 Page(s):952 - 973

[132] Lee, T. K., Zukerman, M., and Addie, R. G., “Admission Control Schemes for Bursty Multimedia Traffic”, IEEE INFOCOM 2001, Apr. 2001.

[133] Leland, W., Taqqu, M., Willinger, W., & Wilson, D., “On the Self-Similar Nature of Ethernet Traffic”, IEEE Trans. Networking, 2(1), Feb. 1994.

[134] Li, Jung-Shian, and Liang, Jing-Zhi, “A novel core-stateless ABR-like congestion avoidance scheme in IP networks”, International Journal of Communication Systems Volume 18, Issue 5 , June 2005, Pages 427 – 447.

[135] Li, K., Krasic, C., Walpole, J., Shor, M., and Pu, C., "The Minimal Buffering Requirements of Congestion Controlled Interactive Multimedia Applications", 8th International Workshop on Interactive Distributed Multimedia Systems (iDMS 2001), Lancaster, UK, September 2001.

[136] Li, L. and Lee G., “DDoS Attack Detection and Wavelets”, Telecommunication Systems, Vol. 28, No. 3-4. (March 2005), pp. 435-451.

[137] Li, Qiong and Schaar, Mihaela van der, “Providing Adaptive QoS to Layered Video Over Wireless Local Area Networks Through Real-Time Retry Limit Adaptation”, IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004.

[138] Li, W., “Overview of Fine Granularity Scalability in MPEG-4 Video Standard”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 3, p. 301-318, March 2001.

[139] Li, W., “Overview of Fine Granularity Scalability in MPEG-4 Video Standard,” IEEE Transactions on Circuits and Systems for Video Technology, March 2001.

[140] Licandro, F., Schembra, G., “A Rate/Quality Controlled MPEG Video Transmission System in a TCP-Friendly Internet Scenario”, Proceedings of the 12th International Packet Video Workshop, April 2002, Pittsburgh PA, USA.

[141] Litwin, L., "FIR and IIR digital filters," *Potentials, IEEE* , vol.19, no.4pp.28-31, Oct/Nov 2000.

[142] Liu, J., Shu, Y., Zhang, L., Xue, F. & Yang, O. “Traffic Modeling Based on ARIMA Models”, IEEE 1999 Canadian Conference on Electrical and Computer Engineering, May 1999.

[143] Liu, Jiangchuan, Chu, Xiaowen, Xu, Jianliang, “Proxy Cache Management for Fine-Grained Scalable Video Streaming”, Proceedings of IEEE INFOCOM'04, Hong Kong, March 7-11, 2004.

[144] Loguinov, D. and Radha, H., “End-to-End Internet Video Traffic Dynamics: Statistical Study and Analysis”, Proceedings of the 21st IEEE INFOCOM, New York, 2002.

[145] Loguinov, D., “Adaptive Scalable Internet Streaming”, PhD Thesis, City University of New York, Jul 2002.

[146] Low, S. H., Lachlan, L. H. A., and Wydrowski, B. P., “Understanding XCP: Equilibrium and Fairness”, In Proceedings of the IEEE INFOCOM, Miami, FL, March 2005.

[147] Lu Xin; Wang Ke; Dou Huijing, "Wavelet Multifractal Modeling for Network Traffic and Queuing Analysis," Computer Networks and Mobile Computing, 2001. Proceedings. 2001 International Conference on , vol., no.pp.260-265, 2001

[148] Ma, S. and Ji, C., “Modeling Heterogeneous Network Traffic in Wavelet Domain”, IEEE/ACM Trans. Netw. 9, 5 (Oct. 2001), 634-649.

[149] Macnicol, J., Arnold, J., Frater, M., “Scalable Video Coding by Stream Morphing”, Circuits and Systems for Video Technology, IEEE Transactions on, Vol.15, Feb. 2005.

[150] Magnaghi, A., Hamada, T., Katsuyama, T., “A Wavelet-Based Framework for Proactive Detection of Network Misconfigurations”, In *Proceedings of the ACM SIGCOMM Workshop on Network Troubleshooting: Research, theory and Operations Practice Meet Malfunctioning Reality* (Portland, Oregon, USA, September 03 - 03, 2004). NetT '04. ACM Press, New York, NY, 253-258.

[151] Mallat, S.G., “A Theory for Multiresolution Signal Decomposition: The Wavelet Representation”, Pattern Analysis and Machine Intelligence, IEEE Transactions on, Vol.11, Iss.7, Jul 1989, Pages:674-693.

[152] Mathis, M., Mahdavi, J., Floyd, S., and Romanow, A., “TCP Selective Acknowledgment Options,” IETF RFC 2018, October 1996.

[153] May, M., Bolot, J., Diot, C., and Lyles, B., “Reasons not to deploy RED”, IWQoS '99, Seventh International Workshop on, pages: 260-262, London, UK, 1999.

[154] Mirozahmedov, F. and Uryasev, S. P., “Adaptive Stepsize Regulation for Stochastic Optimization Algorithm”, Zurnal Vicisl. Mat. I. Mat. Fiz., Issue 23, Vol. 6, pages 1314–1325, 1983.

[155] Nichols, J., Claypool, M., Kinicki, R., and Li, M., “Measurements of the Congestion Responsiveness of Windows Streaming Media”, Proceedings of the 14th International Workshop on Network and Operating Systems Support for Digital Audio and Video, Cork, Ireland, Pages: 94 – 99, 2004.

[156] Optibase, White Paper, A Guide to MPEG-4, www.m4if.org, 2004.

[157] Padhye, J. Model-based Approach to TCP-Friendly Congestion Control. Ph.D. Thesis, University of Massachusetts at Amherst, Mar. 2000.

[158] Padhye, J., Firoiu, V., Towsley, D. and Kurose, J. “Modeling TCP Throughput: A Simple Model and its Empirical Validation”, Proceedings of the ACM SIGCOMM, 1998.

[159] Padhye, J., Kurose, J., Towsley, D., and Koodli, R. “A Model Based TCP-Friendly Rate Control Protocol”. In Proceedings of the ACM NOSSDAV, 1999.

[160] Paganini, F.; Zhikui Wang; Doyle, J.C.; Low, S.H., "Congestion control for high performance, stability, and fairness in general networks," Networking, IEEE/ACM Transactions on , vol.13, no.1pp. 43- 56, Feb. 2005

[161] Papagiannaki, K. , Taft, N., Zhang, Z., Diot, C., “Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models”, In IEEE Infocom, San Francisco, U.S.A., April, 2003

[162] Papagiannaki, K., Cruz, R., and Diot C. “Network Performance Monitoring at Small Time Scales”, In Proceeding of the ACM Internet Measurement Conference (IMC) , Miami, U.S.A., October, 2003.

[163] Papagiannaki, K., Moon, S., Fraleigh, C., Thiran, P., and Diot, C. “Measurement and Analysis of Single-Hop Delay on an IP Backbone Network”, In IEEE Journal on Selected Areas in Communications. Special Issue on Internet and WWW Measurement, Mapping, and Modeling., vol.21, no.6, August, 2003

[164] Papagiannaki, K., Thaft, N., and Diot C. “Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models”, In Proceedings of the IEEE INFOCOM 2003 Conference, San Francisco, U.S.A., April, 2003.

[165] Papagiannaki, K., Thaft, N., Zhang, Z., and Diot C, “Long-Term Forecasting of Internet Backbone Traffic”, in IEEE Transactions on Neural Networks. Special Issue on Adaptive Learning Systems in Communication Networks., vol. 16, no. 5, September, 2005.

[166] Park, K., Kim, G., and Crovella, M. E., “On the Effect of Traffic Self-similarity on Network Performance,” SPIE International Conference on Performance and Control of Network Systems, Nov. 1997.

[167] Pereira, F., and Burnett, I., “Universal Multimedia Experiences for Tomorrow,” IEEE Signal Processing, Special Issue on Universal Multimedia Access, vol. 20, no. 2, Mar. 2003, pp. 63-73.

[168] Phelan, T., “Strategies for Streaming Media Applications Using TCP-Friendly Rate Control”, Internet Draft, Expires: May 2006, October 2005.

[169] Postel, J., “Transmission Control Protocol – DARPA Internet Program Protocol, Specification”, IETF RFC 793, September 1981.

[170] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery B. P., “Numerical Recipes in C: The Art of Scientific Computing”, 2nd Ed., ISBN 0-521-43108-5, Cambridge University Press, Cambridge, MA, 2002.

[171] Qiu, D and Shroff, N.B., "Queueing properties of feedback flow control systems," Networking, IEEE/ACM Transactions on , vol.13, no.1pp. 57- 68, Feb. 2005

[172] R Development Core Team, “R: A Language And Environment For Statistical Computing”, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, http://www.r-project.org, 2006.

[173] Radha, H. and Chen, Y., “Fine-Granular Scalable Video for Packet Networks”, in *Packet Video Workshop,* New York: Columbia Univ., Apr. 1999.

[174] Radha, H. and Wu, M., “Overlay and peer-to-peer multicast with network-embedded FEC”, Image Processing, International Conference on, ICIP '04, Vol. 3, Page(s):1747 – 1750, 2004.

[175] Radha, H. M., Schaar, M. van der and Chen, Y., “The MPEG-4 Fine-Grained Scalable Video Coding Method for Multimedia Streaming Over IP”, IEEE Transactions On Multimedia, Vol. 3, No. 1, March 2001.

[176] Radha, H. M., van der Schaar, M., and Chen, Y., “The MPEG-4 Fine-Grained Scalable Video Coding Method for Multimedia Streaming Over IP”, IEEE Transactions On Multimedia, Vol. 3, No. 1, , p. 53-69, March 2001.

[177] Radha, H. Schaar, M. van der and Karande, S., “Scalable Video Transcaling for the Wireless Internet”, EURASIP Journal on Applied Signal Processing (JASP) - Special

Issue on Multimedia over IP and Wireless Networks, vol. 24, no. 2, pp. 265 - 379, February 2004.

[178] Radha, H., Chen, Y., Parthasarathy, K., and Cohen, R., "Scalable Internet Video Using MPEG-4", *Signal Processing: Image Communication*, no. 15, pp.95–126, September 1999.

[179] Ramakrishnan, K., Floyd, S. and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001

[180] Rangaswami, R., Dimitrijevic, Z., Chang, E., And Chan, S-H G., "Fine-Grained Device Management in an Interactive Media Server", IEEE Transactions on Multimedia, Vol. 5, No. 4, December 2003.

[181] Ranjan, P., Abed, E.H., and La, R.J., "Nonlinear Instabilities in TCP-RED", IEEE Transactions on Networking, vol. 12, no. 6, pp. 1079–1092, December 2004.

[182] Reidi, R., Crouse, M., Ribeiro, V., and Baraniuk, R., "A Multifractal Wavelet Model with Application to Network Traffic", IEEE Trans. Inf. Theory, vol.45, no.3, pp.992–1018, April 1999.

[183] Rejaie, R., and Reibman, A., "Design Issues for Layered Quality-Adaptive Internet Video Playback," in Proceedings of the Workshop on Digital Communications, Taormina, Italy, September 2001, pp. 433–451.

[184] Rejaie, R., Estrin, D., and Handley, M., "Quality Adaptation for Congestion Controlled Video Playback over the Internet," in Proceedings of ACM SIGCOMM, Cambridge, September 1999, pp. 189–200.

[185] Rejaie, R., Handley, M. and Estrin, D. "An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet." In Proceedings of the IEEE INFOCOMM, 1999.

[186] Rhee, I., and Xu, L., "Limitations of Equation-based Congestion Control", In Proceedings of the SIGCOMM'05, August 21–26, 2005, Philadelphia, Pennsylvania, USA, 2005.

[187] Rhee, I., Ozdemir, V. and Yi, Y. TEAR: "TCP Emulation at Receivers – Flow Control for Multimedia Streaming", Technical Report, Department of Computer Science, NCSU. Apr. 2000.

[188] Ribeiro, V.J., Zhang, Z.-L., Moon, S., and Diot, C., "Small-time Scaling Behavior of Internet Backbone Traffic", Computer Networks, Volume 48, Issue 3, June 2005, Pages 315-334.

[189] Ross, S. M., “Introduction to Probability Models”, Academic Press Inc., 5^{a} ed., London, 1993.

[190] S. Floyd, J, Kempf, “IAB Concerns Regarding Congestion for Voice Traffic in the Internet”, March 2004, RFC 3714.

[191] Sahinoglu, Z. & Tekinay, Sirin, “On Multimedia Networks: Self-Similar Traffic and Network Performance”, IEEE Comm. Mag., 37(1), Jan. 1999.

[192] Salehi, J.D.; Zhi-Li Zhang; Kurose, J.; Towsley, D., “Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing”, Networking, IEEE/ACM Transactions on, Vol.6, Iss.4, Aug 1998, Pages:397-410.

[193] Salim, H., Nandy, B., and Seddigh, N., “A proposal for Backward ECN for the Internet Protocol (IPv4/IPv6)”, internet draft draft-salim-jhsbnns-ecn-00.txt, work-in-progress, June 1998.

[194] Sarnoff JND Vision Model, T1A1.5 Working Group Document no. 97-612, ANSI T1 Standards Committee, 1997.

[195] Schaar, M. van der and Andreopoulos, Y., “Rate-Distortion-Complexity Modeling for Network and Receiver Aware Adaptation”, IEEE Transactions on Multimedia, Vol. 7, No. 3, June 2005.

[196] Schierl, T., Wiegand, T., “H.264/AVC Rate Adaptation for Internet Streaming”, Proceedings of the 14th International Packet Video Workshop (PVW 2004), University of California, Irvine, USA, December 2004.

[197] Seeling, P. and Reisslein, M., “The Rate Variability-Distortion (VD) Curve of Encoded Video and its Impact on Statistical Multiplexing”, IEEE Transactions on Broadcasting, September 2005.

[198] Seeling, P., de Cuetos, P., and Reisslein, M. “Fine Granularity Scalable (FGS) Video: Implications for Streaming and a Trace-Based Evaluation Methodology”, IEEE Communications Magazine, Vol. 43, No. 4, p. 138-142, April 2005.

[199] Seeling, P., Reisslein, M., and Kulapala, B., “Network Performance Evaluation Using Frame Size and Quality Traces of Single-Layer and Two-Layer Video: A Tutorial”, IEEE Communications Surveys & Tutorials, 3rd. Quarter, 2004.

[200] Setton, E., Noh, J., and Girod, B., “Rate-Distortion Optimized Video Peer-to-Peer Multicast Streaming”, Workshop on Advances in Peer-to-Peer Multimedia Streaming, ACM Multimedia 2005, November, 2005, Singapore.

[201] Shakkottai, S., Kumar, A., Karnik, A. and Anvekar, A., "TCP Performance Over End-To-End Rate Control and Stochastic Available Capacity", Networking, IEEE/ACM Transactions on, Volume: 9 , Issue: 4, Aug. 2001 , page(s): 377 – 391

[202] Shakkottai, S., Rappaport, T. and Karlsson, P., "Cross-Layer Design for Wireless Networks," IEEE Communication Magazine, vol. 41, no. 10, Oct. 2003, pp. 74–80.

[203] Shensa, M. J., "The Discrete Wavelet Transform: Wedding the À Trous and Mallat Algorithms", Signal Processing, IEEE Transactions on, Vol.40, Iss.10, Oct 1992, Pages:2464-2482.

[204] Shin, J., Kim, J.W. and Kuo, J. C.-C., "Quality of Service Mapping Mechanism for Packet Video in Differentiated Services Network", IEEE Trans. On Multimedia (Special Issue on Multimedia over IP), vol. 3, no. 2, pp. 219-231, June 2001.

[205] Sikora, T., "MPEG Digital Video Coding Standards," *Digital Electronics Consumer Handbook*, McGraw Hill, 1997.

[206] Smith, J. R., "MPEG-21 Digital Item Adaptation: Enabling Universal Multimedia Access", IEEE Multimedia, January-March 2004.

[207] Smith, J.O., "Introduction to Digital Filters", CCRMA, University of Stanford, Draft Version, available on-line at http://www-ccrma.stanford.edu/~jos/filters/, 2002.

[208] Srikant, R. "The Mathematics of Internet Congestion Control", Birkhauser, 2004.

[209] Srikant, R., "Models and Methods for Analyzing Internet Congestion Control Algorithms." In Advances in Communication Control Networks in the series "Lecture Notes in Control and Information Sciences (LCNCIS)," Springer-Verlag, 2004.

[210] Sripanidkulchai, K., Maggs, B., and Zhang, H., "An Analysis of Live Streaming Workloads on the Internet", ACM Internet Measurement Conference, 2004

[211] Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," IETF RFC 2001, January 1997.

[212] Stockhammer, Thomas, Jenkac, Hrvoje and Kuhn, Gabriel, "Streaming Video Over Variable Bit-Rate Wireless Channels", IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004.

[213] Stoev, S., Taqqu, M. S., Park, C., and Marron, J.S., "On the Wavelet Spectrum Diagnostic for Hurst Parameter Estimation in the Analysis Of Internet Traffic", Computer Networks 48, (2005) 423-445.

[214] Tan, D. and Zakhor, A. "Real-time Internet Video Using Error Resilient Scalable Compression and TCP-friendly Transport Protocol". IEEE Transactions on Multimedia, May 1999.

[215] Taylor, James W., "Smooth transition exponential smoothing", Journal of Forecasting, John Wiley & Sons Ltd., Vol. 23, Issue 6, pp. 385-404, 2004.

[216] The Network Simulator – ns-2, www.isi.edu/nsnam/ns/.

[217] Trigg, D. W., and Leach, D. H., "Exponential Smoothing with an Adaptive Response Rate", Operational Research Quarterly, vol. 18, Issue 1, pp. 53-59, 1967.

[218] Trigg, D., "Monitoring a Forecasting System", Operations Research Quarterly, Vol. 15, Issue 3, pages 271–274, 1964.

[219] Tukey, J. W., "Exploratory Data Analysis", Reading Massachusetts: Addison-Wesley, 1977.

[220] Van Beek, P.; Smith, J.R.; Ebrahimi, T.; Suzuki, T.; Askelof, J.; "Metadata-driven Multimedia Access", Signal Processing Magazine, IEEE, Volume 20, Issue 2, Page(s):40 – 52, March 2003.

[221] Veres, A., Kenesi, Zs., Molnar, S. & Vattay, G., "On the Propagation of Long-Range Dependence in the Internet", SIGCOMM´2000, August 2000.

[222] Vetro, A. and Timmerer, C., "Digital Item Adaptation: Overview of Standardization and Research Activities", IEEE Transactions on Multimedia, Vol. 7, No. 3, June 2005.

[223] Video Transport over Wireless Networks Harinath Garudadri Phoom Sagetong Sanjiv Nanda, ACM Multimedia'04, October 10–16, 2004, New York, New York, USA.

[224] Vieron, J., and Guillemot, C., "Real-Time Constrained TCP-Compatible Rate Control for Video Over the Internet", Multimedia, IEEE Transactions on, Volume 6, Issue 4, Aug. 2004 Page(s):634 – 646.

[225] Vojnovic, M. and Le Boudec, Jean-Yves, "On the Long-Run Behavior of Equation-Based Rate Control", IEEE/ACM Transactions on Networking, Vol. 13, No. 3, June 2005.

[226] Vojnovic, M. Le Boudec, J-Y., "On the long-run behavior of equation-based rate control", Proceedings of the ACM SIGCOMM, August 2002, Pittsburgh, USA, pp 103-116.

[227] Wakamiya, N., Miyabayashi, M., Murata, M. and Miyahara, H., "MPEG-4 Video Transfer with TCP-Friendly Rate Control", Proceedings of the 2nd International Workshop on QoS in Multiservice IP Networks (QoS-IP 2003), Vol. LNCS2601, Milano, pp. 539-550, February 2003.

[228] Wang, B., Kurose, J., Shenoy, P., and Towsley, D., “Multimedia Streaming via TCP: An Analytic Performance Study”, Proceedings of the SIGMETRICS/Performance’04, June 12–16, p. 406-407, New York, NY, USA, 2004.

[229] Wang, B., Sen, S., Adler, M., and Towsley, D., “Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution”, IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004.

[230] Wang, J., Li, L., Low, S. H., and Doyle, J. C., “Cross-Layer Optimization in TCP/IP Networks”, IEEE/ACM Trans. Network 13, (3), Jun. 2005, 582-595.

[231] Wang, X., Meditch, J. S., “Adaptive Wavelet Predictor to Improve Bandwidth Allocation Efficiency of VBR Video Traffic”, Computer Communications, Volume 22, Volume 22, Number 1, 15 January 1999

[232] Wang, Y., and Zhu, Q., “Error Control and Concealment for Video Communications: A Review,” Proceedings of the IEEE, vol. 86, no. 5, pp.974–997, May 1998.

[233] Wang, Z. and Bovik, A. C. “Why is Image Quality Assessment so Difficult?,” in Proceedings of the IEEE International Conference Acoustics, Speech, and Signal Processing, May 2002.

[234] Wang, Z., Banerjee, S., and Jamin, S., “Media-Friendliness of a Slowly-Responsive Congestion Control Protocol,” Proceedings of ACM NOSSDAV, Cork, Ireland, 2004.

[235] Weber, S., and de Veciana, G., “Network Design for Rate Adaptive Media Streams”, Proceedings of 22nd IEEE INFOCOM, San Francisco, March 2003.

[236] Wei, D. X., Jin, C., Low, S. H., and Hegde, S., “FAST TCP: Motivation, Architecture, Algorithms, Performance”, IEEE/ACM Trans. on Networking, to appear in 2007.

[237] Welzl, M., “Router Aided Congestion Avoidance with Scalable Performance Signalling”, Proceedings of KiVS 2005, Kaiserslautern, Germany, March, 2005.

[238] Welzl, M., “Scalable Performance Signalling and Congestion Avoidance”, Ph.D. Thesis, Institute of Computer Science, University of Innsbruck, December 2003.

[239] Welzl, M., Mühlhäuser, M., “Scalability and Quality of Service: a Trade-off?”, IEEE Communications Magazine Vol. 41 No. 6, June 2003, pp. 32-36.

[240] Welzl, Michael, “Scalable Performance Signalling and Congestion Avoidance”, Kluwer Academic Publishers), August 2003.

[241] Whybark, D. C., “A Comparison of the Surprising Conclusions cited by Adaptive Forecasting Techniques”, Logistics and Transportation Review, 8 (3), pp. 13-26, 1972.

[242] Wikipedia contributors, "Finite impulse response", Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/w/index.php?title=Finite_impulse_response&oldid=40454566 (accessed March 2, 2006).

[243] Willinger, W., Alderson, D., and Li, L., “A Pragmatic Approach to Dealing with High Variability in Network Measurements”, ACM Internet Measurement Conference (IMC) – 2004.

[244] Willinger, W., Taqqu, M.S., Sherman, R. & Wilson, D.V., “Self-Similarity Through High-Variability: Statistical Analisys of Ethernet LAN Traffic at the Source Level”, IEEE Trans. Networking, 5(1), p. 71-86, 1997.

[245] Xia, Y., Subramanian, L., Stoica, I., and Kalyanaraman, S., “One More Bit Is Enough”, In Proceedings of the SIGCOMM’05, August 21–26, 2005, Philadelphia, Pennsylvania, USA, 2005.

[246] Xu, L. and Helzer, J., “Media Streaming via TFRC: An Analytical Study of the Impact of TFRC on User-Perceived Media Quality”, Proceedings of IEEE INFOCOM 2006, Barcelona, Spain, April, 2006.

[247] Yu, F., Zhang, Q., Zhu, W., and Zhang, Y.-Q., “Qos-Adaptive Proxy Caching For Multimedia Streaming Over the Internet”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 3, March 2003

[248] Zhang, E., Towsley, D., and Wileden, J., “Towards Interoperable Multimedia Streaming Systems”, Proceedings of Packet Video Workshop, 2002

[249] Zhang, Q., Zhu, W., and Zhang, Y.-Q., “Resource Allocation for Multimedia Streaming over the Internet,” IEEE Transactions on Multimedia, vol. 3, no. 3, pp. 339–335, September 2001.

[250] Zhang, Y., and Henderson, T. R., “An Implementation and Experimental Study of the eXplicit Control Protocol (XCP)”, Proceedings of the IEEE INFOCOM 2005 Conference, March 2005.

[251] Zhang, Y., and Loguinov, D., “Oscillations and Buffer Overflows in Video Streaming under Non-Negligible Queueing Delay”, Proceedings of ACM NOSSDAV 2004, Cork, Ireland, June 2004.

[252] Zhang, Yongguang and Mohin, Ahmed, “The Problem of Explict-Control-Based Transport Protocols in Wireless Networks”, International Workshop on Wireless and Industrial Automation (WIA'05), 11th IEEE Real-Time and Embedded Technology and Applications Symposium, March7-10, 2005, San Francisco, California.

[253] Zhao, L. Shin, J., Kim, J. Kuo, J. C-C., “Constant quality rate control for streaming MPEG-4 FGS video,” in Proc. IEEE International Symposium on Circuits and Systems (ISCAS)`2002, Scottsdale, AZ, May 2002.

[254] Zhao, L., Kim, J. W. and Kuo, J. C.-C., “MPEG-4 FGS video streaming with constant-quality rate control and differentiated forwarding,” in Proc. SPIE Visual Communications and Image Processing `2002, San Jose, CA, Jan. 2002.