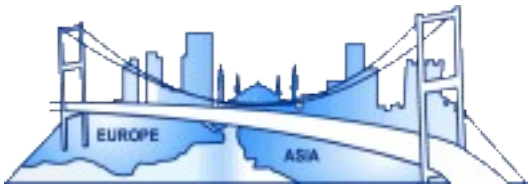
Proceedings of

MULTICOMM'06

The First Multimedia Communications Workshop: State of the Art and Future Directions

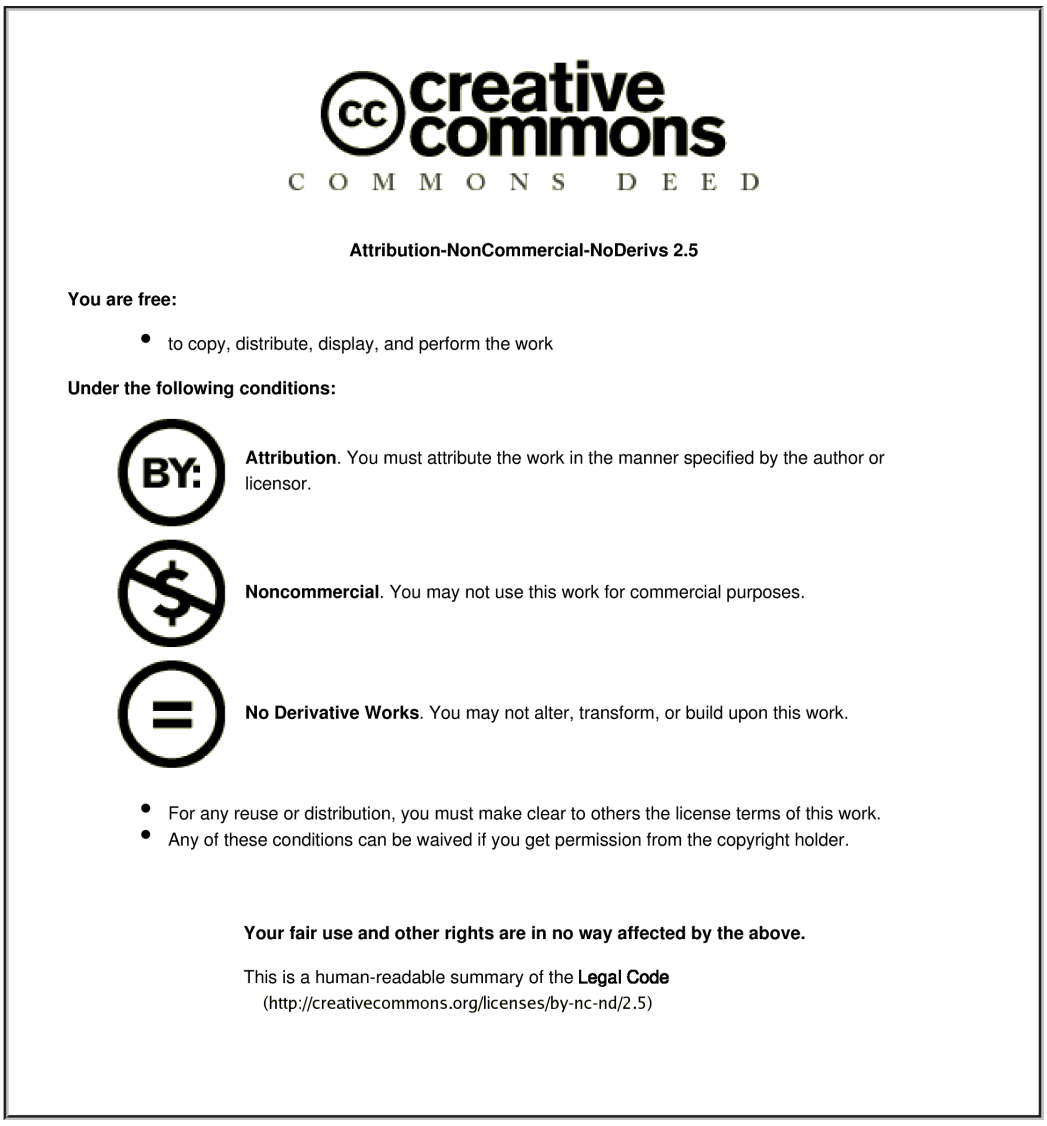
A Workshop of the
IEEE International Conference on
Communications 2006



June 11, 2006, Istanbul, Turkey

Sponsored by

IEEE Communications Society
Multimedia Communications Technical Committee



creative commons

COMMONS DEED

Attribution-NonCommercial-NoDerivs 2.5

You are free:

- to copy, distribute, display, and perform the work

Under the following conditions:

BY: **Attribution**. You must attribute the work in the manner specified by the author or licensor.

Noncommercial. You may not use this work for commercial purposes.

No Derivative Works. You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

This is a human-readable summary of the **Legal Code**
(http://creativecommons.org/licenses/by-nc-nd/2.5)

Message from the General Chair

Welcome to the First Multimedia Communications Workshop!

The multimedia revolution is undoubtedly taking place. Never in history, in fact, humanity has had the opportunity of accessing so much audio and video content, from such a large variety of sources and for purposes so diverse that include both pure entertainment and scholarly pursuits.

The multimedia revolution, however, is still far from being over; many issues remain open to debate and competing visions of the future continue to vie for supremacy. It is, therefore, the right time to discuss what is the future of multimedia delivery, quantitatively addressing questions such as the role of streaming in an increasingly download-based digital world, peer-to-peer multimedia communications, multimedia over ad hoc mobile networks, DVB-H, packet videoconferencing, video sensor networks, 3D television, car-to-car and car-to-roadside signal communications, and other evolving and emerging applications.

The goal for this workshop is to offer a unique, focused opportunity to gather and discuss with peers the state of the art as well as the most interesting potential new directions of multimedia communications.

We would like to thank all the authors who responded to the Call for Papers, regardless of whether their papers have been included in this workshop or not due to space limitations. We would also like to acknowledge the contribution of many experts who participated in the review process, and provided helpful and valuable suggestions to the authors on improving the content and presentations of the articles.

We whole heartedly thank the ICC 2006 Organizing Committee for their support, and in particular Dr. Abbas Yongacoglu for his kind and constant support. We also thank the IEEE Multimedia Communications Technical Committee and in particular Dr. Heather Yu for her advice, energy and relentless encouragement from the very early stages of this initiative.

We hope you will enjoy Multicomm 2006!

Juan Carlos De Martin
Multicomm 2006 General and Technical Program Chair
Politecnico di Torino, Italy

Technical Program

9:00-10:00 KEYNOTE SPEECH

State of the Art and Future Directions in 3DTV
Levent Onural *(Bilkent University)*

10:30-12:15 MORNING SESSION

WS02.1 - $PSNR_{r,f}$: Assessment of Delivered AVC/H.264 Video Quality over 802.11a WLANs with Multipath Fading
Jing Hu, Sayantan Choudhury, and Jerry D. Gibson *(University of California, Santa Barbara, USA)*

WS02.2 - Error Propagation After Concealing a Lost Speech Frame
Christian Hoene *(University of Tübingen, Germany)*, Ian Marsh *(KTH Stockholm, Seden)*, Günter Schäfer *(University of Illmenau, Germany)*, and Adam Wolisz *(Technical University of Berlin)*

WS02.3 - A Performance Study of VoIP Applications: MSN vs. Skype
Wen-Hui Chiang, Wei-Cheng Xiao, and Cheng-Fu Chou *(National Taiwan University, Taiwan)*

WS02.4 - BMC: A Two-stage Switch Architecture for High Performance Multimedia Communication
Yang Xu, Bin Liu, Beibei Wu, and Wei Li *(Tsinghua University, China)*

13:30-15:15 AFTERNOON SESSION

WS02.5 - Rate Adaptation for Buffer Underflow Avoidance in Multimedia Signal Streaming
Matteo Petracca, Fabio De Vito, and Juan Carlos De Martin *(Politecnico di Torino, Italy)*

WS02.6 - Dissemination of Dynamic Multimedia Content in Networked Virtual Environments
Tolga Bektas *(Bilkent University, Turkey)*, Farzad Safaei *(University of Wollongong, Australia)*, Iradj Ouveysi *(University of Melbourne, Australia)*, and Osman Oguz *(Bilkent University, Turkey)*

WS02.7 - First Video Streaming Experiments on a Time Driven Priority Network
Mario Baldi, Guido Marchetto *(Politecnico di Torino, Italy)*

WS02.8 - Open Issues in P2P Multimedia Streaming
Djamal-Eddine Meddour *(France Telecom R&D, France)*, Mubasher Musthaq, Toufik Ahmed *(University of Bordeaux, France)*

15:45-17:00 FINAL PANEL

Multimedia Communications: Future Directions
Panelists:
Juan Carlos De Martin *(Politecnico di Torino, Italy)*
Alex Gelman *(Panasonic Research, USA)*
Jerry Gibson *(University of California, Santa Barbara, USA)*
Levent Onural *(Bilkent University, Turkey)*
Oguz Sunay *(Koc University, Turkey)*

Keynote Speech
State of the Art and Future Directions in 3DTV

Keynote speaker: *Levent Onural, Bilkent University*

Although stereoscopic 3D technologies have been as old as their 2D counterparts, the level of acceptance of 3D viewing by the consumers, and therefore, by the industry has been pretty low. A reason for this is the viewing discomfort associated with stereoscopic approaches. Currently available 3D displays are still stereoscopic; improvements allow elimination of special eye-wear, and enhance the quality by tracking observer coordinates. On the other hand, high-end 3D displays, with no viewer discomfort, must be based on coherent and non-coherent holographic techniques, including integral imaging; such displays are still experimental, and acceptable quality can only be achieved after the underlying electronics and optics technologies evolve.

Even though the user-end of the 3DTV or any kind of 3D visual system is the display, the end-to-end operation of such systems naturally involve many more functional components. Capture of 3D scenes, their abstract representation, compression and coding, transmission, and finally the display are the major components. And there are many alternative techniques for each such component. A common capture technique utilizes multiple cameras. The captured moving 3D scenes are represented using computer graphics techniques. Due to redundancy in information content, significant compression of captured 3D data is possible. Transport of 3D video data may be constructed on prior 2D techniques, but the discriminating features and constraints impose a different set of priorities, and thus a different approach, to the transport problem.

3DTV field is a multidisciplinary area, and a successful cooperation of researchers from optics, electronics, signal processing, computer graphics, telecommunications, and other fields are essential for success. Applications of the technology is vast.

Levent Onural was born in Izmir, Turkey in 1957. He received the B.S. and M.S. degrees in electrical engineering from Middle East Technical University, Ankara, Turkey, and the Ph.D. degree in electrical and computer engineering from State University of New York at Buffalo in 1985. He was a Fulbright scholar between 1981 and 1985. After a Research Assistant Professor position at the Electrical and Computer Engineering Department of State University of New York at Buffalo, he joined the Electrical and Electronics Engineering Department of Bilkent University, Ankara, Turkey, where he is a Professor at present.

His current research interests are in the areas of image and video processing with emphasis on very low bit rate video coding, texture modeling, non-linear filtering, holographic TV and signal processing aspects of optical wave propagation. He has published more than 100 papers and received about 500 citations. He and his team have contributed to MPEG-4 activities through COST211 Analysis Model. Currently, he is the Coordinator of EC funded 3DTV Project.

Dr. Onural is a senior member of IEEE. He was the general co-chair of IEEE 2000 International Conference on Acoustics Speech and Signal Processing ICASSP'2000. Dr. Onural was the Director of IEEE Region 8 (Europe, Africa and Middle East), and a member of IEEE Board of Directors, which is the highest board of IEEE, between 2001-2003. He was a member of IEEE Assembly in 2001-2002. He served as the 2003 Secretary of IEEE, and he was a member of IEEE Executive Committee in 2003. Levent Onural was nominated by the IEEE Board of Directors to the position of 2005 IEEE President-elect (2006 IEEE President); he is the first person from outside of North America nominated for this position in 120 years of history of IEEE. Levent Onural is a recipient of IEEE Third Millenium Medal. He is currently an associate editor of IEEE Transactions on Circuits and Systems for Video Technology

Multicomm 2006 Workshop Organization

General and Technical Program Chair: Juan Carlos De Martin, Politecnico di Torino, Italy

Publication Chair: Antonio Servetti, Politecnico di Torino, Italy

Publicity Chair: Enrico Masala, Politecnico di Torino, Italy

Technical Program Committee: Toufik Ahmed, University of Bordeaux, France
Reha Civanlar, Koc University, Turkey
Enrica Filippi, ST Microelectronics, Italy
Pascal Frossard, EPFL, Switzerland
Jerry Gibson, UC Santa Barbara, USA
Enrico Masala, Politecnico di Torino, Italy
Alan McCree, MIT Lincoln Laboratory, USA
Michela Meo, Politecnico di Torino, Italy
David Miller, Penn State University, USA
Antonio Servetti, Politecnico di Torino, Italy
Oguz Sunay, Koc University, Turkey
Murat Tekalp, Koc University / University of Rochester
Heather Yu, Panasonic Research, USA

Sponsored by

Table of Contents

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

$PSNR_{r,f}$: Assessment of Delivered AVC/H.264 Video Quality over 802.11a WLANs with Multipath Fading

Jing Hu, Sayantan Choudhury and Jerry D. Gibson
Department of Electrical and Computer Engineering
University of California, Santa Barbara, California 93106-9560
Email: {jinghu, sayantan, gibson}@ece.ucsb.edu

***Abstract*— Emerging as the method of choice for compressing video over WLANs, the AVC/H.264 standard is a suite of coding options and parameters whose values are to be chosen for specific videos and channel conditions. We investigate the delivered quality of AVC/H.264 coded video across the video characteristics, the quantization parameter (QP), the group of picture size (GOPS), the payload size (PS), PHY data rate in 802.11a, and average channel signal to noise ratio (SNR). We show that the delivered quality of a coded video sequence varies tremendously across the frames per channel realization, and across different channel realizations of the same PHY data rate at the same average channel SNR. The performance also varies across different average channel SNRs and combinations of codec parameters. We propose a statistical video quality indicator $PSNR_{r,f}$ defined as peak SNR (PSNR) achieved by $f\%$ of the frames in each one of the $r\%$ of the realizations. We study the correspondence between $PSNR_{r,f}$ and perceptual video quality through a subjective experiment and employ $PSNR_{r,f}$ to assess video communications performance under various channel conditions.**

I. Introduction

Recently there has been a significant interest in using packetized video over WLANs. The assessment of the delivered video quality is critical for designing, evaluating and improving, in a cross-layer manner, the video compression schemes, the physical layer (PHY) configuration and the 802.11 protocols and access schemes. Perceptual quality measurement of video sequences has been a very active research area but no universally effective objective metric has been standardized [1]. The objective metrics that have been proposed are computationally very expensive. The measurement of video quality is made even more complicated by packet losses in WLANs with frequency selective multipath fading and the packet loss concealment schemes embedded in the video codecs.

The Advanced Video Coding (AVC) standard, designated ITU-T H.264 and MPEG-4 Part 10, offers a coding efficiency improvement by a factor of two over previous standards and its network abstraction layer (NAL) transports the coded video data over networks in a more "network-friendly" way [2]. Because of these two features, the AVC/H.264 standard is emerging as the method of choice for video coding over WLANs.

In this paper, we investigate the performance of AVC/H.264 coded video for IEEE 802.11a WLANs in a frequency selective multipath fading environment. The AVC/H.264 standard is a suite of coding options and there are many important choices of parameters to be made for communication over wireless LANs with the IEEE 802.11 protocols and access schemes. Therefore we code several video sequences using combinations of parameter values for the three dominant parameters in the codecs: group of picture sizes (GOPSs), quantization stepsizes which are indexed by quantization parameters (QPs), and video payload sizes (PSs). An extensive set of packet loss realizations are generated for a physical layer (PHY) data rate of 6 Mbps, different average channel SNRs (3.5 dB for bad channel, 5 dB for average channel, 7 dB for good channel at 6 Mbps), and two PSs (small–100 bytes and large–1100 bytes). A small set of tests for additive white Gaussian noise (AWGN) channel are also conducted for comparison. Three different videos coded using combinations of GOPSs (10, 15, 30, 45 frames), QPs (26 for refined quantization and 30 for coarse quantization) and PSs are processed based on the packet loss patterns generated by the channel. In the medium access control (MAC) layer of IEEE 802.11, a cyclic redundancy check (CRC) is computed over the entire packet, and if a single bit error is detected, the packet is discarded. For data, a retransmission would be requested, however, for video we do not request a retransmission, but rely on packet loss concealment.

We show that the delivered quality of a coded video sequence varies tremendously across the frames per channel realization, and across different channel realizations of the same PHY data rate at the same average channel SNR. Therefore average bit error rate (BER) or packet error rate (PER) is not a good choice for designing adaptation schemes (Section III).

We propose a statistical video quality indicator $PSNR_{r,f}$ as PSNR achieved by $f\%$ of the frames in each one of the $r\%$ of the realizations. This quantity has the potential to capture the performance loss due to damaged frames in a particular video sequence ($f\%$), as well as to indicate the probablity of

This work was supported by the California Micro Program, Applied Signal Technology, Dolby Labs, Inc. and Qualcomm, Inc., by NSF Grant Nos. CCF-0429884 and CNS-0435527, and by the UC Discovery Grant Program and Nokia, Inc..

a user experiencing a specified quality over the channel ($r\%$). The percentage of realizations also has the interpretation of what percentage out of many video users would experience a given video quality. We study the correspondence between $PSNR_{r,f}$ and perceptual video quality through a subjective experiment and compare $PSNR_{r,f}$ to the average PSNR across all the frames and channel realizations (Section IV). We employ $PSNR_{r,f}$ to assess the delivered video quality in each average channel condition. AWGN channels are also tested for comparison (Section V).

II. Background

A. Video quality asssessment

The methods of measuring perceptual video quality are usually divided into two categories: subjective measurements and objective measurements. Subjective video quality measurements have been conducted under standardized International Telecommunication Union (ITU) Recommendations ITU-T P.910 [3] and ITU-R BT.500 [4]. Subjective measurements involve a huge number of experiments on human subjects so they are expensive and time-consuming. The most commonly used objective video quality metric is the mean squared error (MSE) or equivalently the PSNR of the distorted videos. A number of sophisticated objective video quality metrics have been proposed in the past few years based on the lower order processing of human vision systems (HVS) [1], [5], [6]. These sophisticated objective metrics focus on quantifying the quality degradation due to the artifacts caused by compression and therefore they correlate to human perception more precisely than PSNR.

However for video over WLANs, the quality degradation in the video encoder is overwhelmed by the quality degradation caused by the possible packet losses in the wireless channel, even though the losses are concealed to some extent in the decoders. If for a single frame, the PSNR of the compressed signal is known and it is also known that the reconstructed frame without errors has acceptable video quality for the application, the PSNR of the frame reconstructed at the decoder after transmission through the channel can be a useful indicator of performance. However when the PSNRs vary significantly across the frames in a video sequence, which we will show is the case for delivered video with packet losses, the assessment of the overall quality of this video sequence is unclear. Furthermore in the scenario when the quality a video user experiences is not deterministic or the scenario when multiple users are using the same channel, the assessment of the channel in terms of the delivered video quality has not been studied.

B. Choices in AVC/H.264 codecs

Figure 1 is a simplified diagram of a typical AVC/H.264 encoder, with the options for the major schemes and parameters presented in the callout blocks. Some of these options are new in AVC/H.264 such as “9 intra-frame prediction modes” and “different block sizes”, while others are inherited from the older standards but with refinements. Each video sequence has its unique properties and the codec parameters must be chosen accordingly. For example, in Table I we show that the average PSNR, source bit rate and intra-predicted frame and inter-predicted frame sizes are quite different for three different video sequences at two values for QP. These videos are coded using AVC/H.264 reference software [7] JM10.1 with GOPS = 90 frames, frame rate = 15 frames per second (fps), 5 reference frames, and no packet loss. This suggests that to derive an indicator of delivered AVC/H.264 video quality, a collection of video sequences needs to be coded using combinations of different values for the codec parameters.

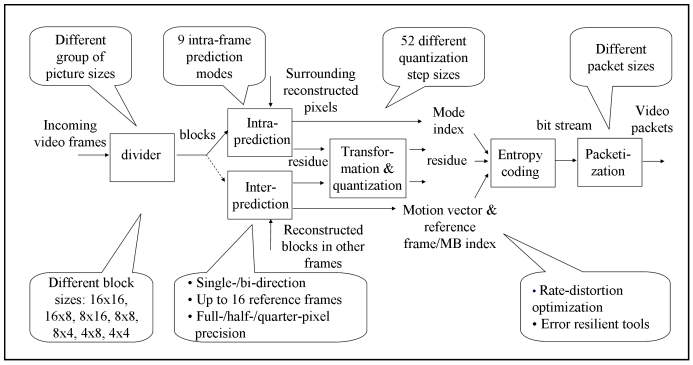


Fig. 1. Simplified diagram of AVC/H.264 encoder with different coding options and parameters

TABLE I
AVC/H.264 CODEC PERFORMANCE OF THREE DIFFERENT VIDEO SEQUENCES

Video	silent.cif		paris.cif		stefan.cif	
Typical application	video conference		news broadcast		sports broadcast	
QP	26	30	26	30	26	30
Average PSNR	36.69	34.22	36.59	33.45	36.69	33.47
Bit rate (kbps)	169.5	97.8	373.5	218.9	1396.8	404.6
I frame size (bytes)	13945	8826	19886	14390	30432	15978
Average of P frame size (bytes)	1272	725	2924	1683	11429	3230
Variance of P frame size (bytes)	412	254	322	219	1544	625

C. Link adaptation in IEEE 802.11a

The IEEE 802.11a wireless systems operate in the 5 GHz Unlicensed National Information Infrastructure (U-NII) band. It uses twelve 20 MHz channels from the U-NII lower-band (5.15-5.25 GHz), U-NII mid-band (5.25-5.35 GHz) and U-NII upper-band (5.725-5.825 GHz) with the first 8 channels dedicated for indoor use. Each 20 MHz channel is composed of 52 subcarriers, with 48 being used for data transmission and the remaining 4 used as *pilot carriers* for channel estimation and phase tracking needed for coherent demodulation. The 802.11a PHY provides 8 modes with varying data rates from 6 to 54 Mbps by using different modulation and coding schemes as shown in Table II. Forward error correction (FEC) is done

TABLE II
PHY MODES IN IEEE 802.11A

Mode	Modulation	Code Rate	Data Rate	Bytes per Symbol
1	BPSK	1/2	6 Mbps	3
2	BPSK	3/4	9 Mbps	4.5
3	QPSK	1/2	12 Mbps	6
4	QPSK	3/4	18 Mbps	9
5	16-QAM	1/2	24 Mbps	12
6	16-QAM	3/4	36 Mbps	18
7	64-QAM	2/3	48 Mbps	24
8	64-QAM	3/4	54 Mbps	27

by using a rate 1/2 convolutional code and bit interleaving for the mandatory rates and using puncturing for the higher rates. A detailed description of OFDM systems and applications to wireless LANs can be found in [8], [9].

The OFDM physical layer convergence procedure (PLCP) is used for controlling frame exchanges between the MAC and PHY layers. The frame format for the MAC data frame is given in Fig. 2. Each MAC frame or MAC protocol data unit (MPDU) consists of MAC header, variable length frame body and a frame check sequence (FCS). The MAC header and FCS consists of 28 bytes and the ACK is 14 bytes long. The frame body varies from 0-2304 bytes including the RTP/UDP and IP headers. The RTP and UDP overhead for multimedia traffic is 12 and 8 bytes, respectively, and another 20 bytes is added for the IP header. A PLCP Protocol Data Unit (PPDU) is formed by adding a PLCP preamble and header to the MPDU. The PLCP header (excluding the service field) is transmitted using BPSK modulation and rate 1/2 convolutional coding. The six "zero" tail bits are used to unwind the convolutional code, i.e. to reset it to the all zero state, and another 16 bits is used by the SERVICE field of the PLCP header.

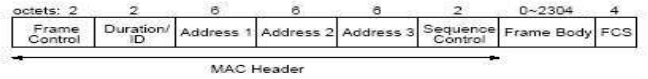
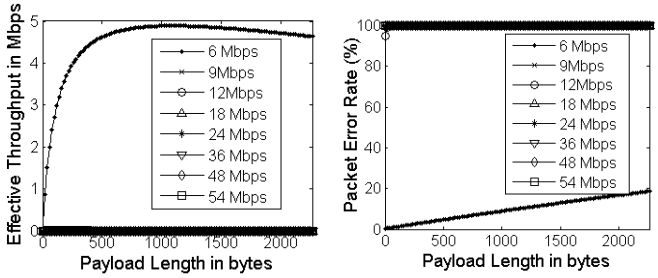


Fig. 2. Frame format of a data frame MPDU

Most link adaptation schemes target data transmission [10], [11], as opposed to voice and video. In [11] the expected effective throughput is expressed as a closed-form function of the data payload length and the selected data transmission rate as a function of channel SNR in AWGN and Nakagami fading environments. A joint selection of data rate and payload length is done to maximize the user throughput without retransmissions. In [12], joint PHY-MAC based link adaptation schemes to maximize throughput and achieve a PER constraint for frequency selective multipath fading channels are proposed. However, the connection between PER and concealed video quality is not taken into account by these link adaptation schemes. The cross-layer adaptation schemes for video communications proposed in [13], [14] model distortion in the video as a function of the average BER or PER of the wireless channels without consideration of the effects of the variation in BER or PER on the video quality and they exclude the different options in the source codecs for adaptation.

III. VIDEO OVER WLAN SETUP

We investigate the performance of AVC/H.264 coded video across the video characteristics, the quantization parameter (QP), the group of picture size (GOPS), the payload size (PS), PHY data rate in 802.11a, and the average channel SNR for multipath fading channels. The wireless channel model used for the multipath fading case is the Nafteli Chayat model [15], which is an important indoor wireless channel model with an exponentially decaying Rayleigh faded path delay profile. The rms delay spread used was 50 nanoseconds which is typical for home and office environments. Each realization of the multipath delay profile corresponds to a certain loss pattern for that fading realization. Figure 3 plots the effective throughput and PER for the different IEEE 802.11a PHY data rates at an SNR of 3 dB for additive white Gaussian noise. One intuitive design is to choose the PS that maximizes the effective throughput, such as, for example, about 1100 bytes in Figure 3(a). However, this optimal PS corresponds to a possibly large PER of 10% in Figure 3(b), which might not yield acceptable video quality. To compare the results of using different PSs, we choose 1100 bytes as the large PS, which is close to the optimal PS for throughput maximization under the conditions in Figure 3, and 100 bytes as the small PS, which yields much lower throughput but also much lower PER.



(a) Throughput at channel SNR 3 (b) PER at channel SNR 3

Fig. 3. Effective throughput and PER for at a SNR of 3 dB for IEEE 802.11a PHY rates

Figure 4 plots the cumulative distribution function (cdf) of PER for 100 byte and 1100 byte packets in a multipath fading environment at average channel SNRs of 3.5 dB, 5 dB and 7 dB when the 6 Mbps PHY data rate is used. It shows that for the same channel SNR and the same PS, the PER of an individual channel realization can range from 0% to 100%, with the 1100 byte packets more likely to be lost than the 100 byte packets. Roughly, at most a 10% packet loss in video can be concealed for acceptable quality. Note from Figure 4 that for a PS of 100 bytes and an average SNR = 7 dB, the average PER across the realizations is 5.5%, but this PER is achieved by only 90% of the realizations. Thus 10% of the realizations will have a higher PER than the average. The cdf of PER for 100 byte packets and 6 Mbps PHY data rate in an AWGN environment at a channel SNR of 0.5 dB is also plotted. It shows that the average PER of an AWGN channel is much lower than that of a multipath fading channel even at a much poorer channel SNR. Also the variation of the PER of an AWGN channel is significantly lower as we can see that all PERs of the AWGN channel in this figure vary only from 1% to 3%.

We are mainly concerned with real-time two-way video-

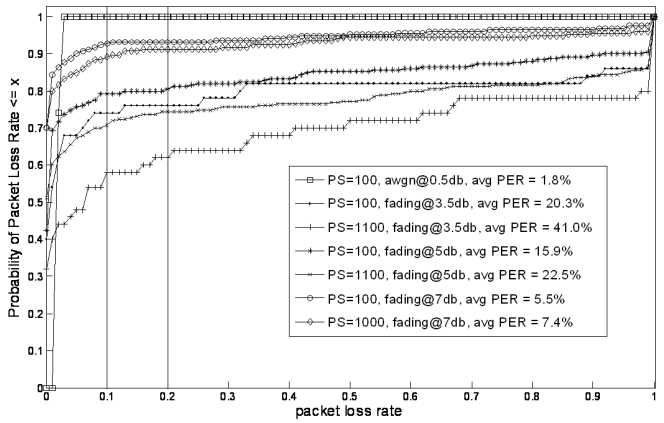


Fig. 4. Cumulative distribution function (cdf) of packet error rate of different channels in AWGN and multipath fading environments for 100 byte and 1100 byte packets and PHY data rate as 6 Mbps

conferencing in which round-trip delay of video needs to be less than 500 ms and the coding complexity needs to be low. Therefore the Baseline Profile with forward-only inter-frame prediction is chosen in the simulations and we are interested in not requiring any retransmissions. 90 frames of each of three videos, silent.cif, paris.cif and stefan.cif are processed at 15 fps and the number of reference frames is fixed as 5. The latest version of AVC/H.264 reference software [7] JM10.1 is used, including its packet loss concealment implementation. The three dominant parameters – QP, GOPS and PS are tested for different values. QP dominates the quantization error and has a major effect on the coded video data rate. GOPS determines the intra-frame refresh frequency and plays an important role when there is packet loss. PS is the parameter that is carried forward from the source to the PHY layer. The remainder of the adjustable parameters in Figure 1: the intra-mode, block size and inter-frame prediction precision are optimally chosen in the encoder to yield the minimum source bit rate. 250 packet loss patterns are generated for each of the investigated combinations of average channel SNR, video PS and PHY data rate.

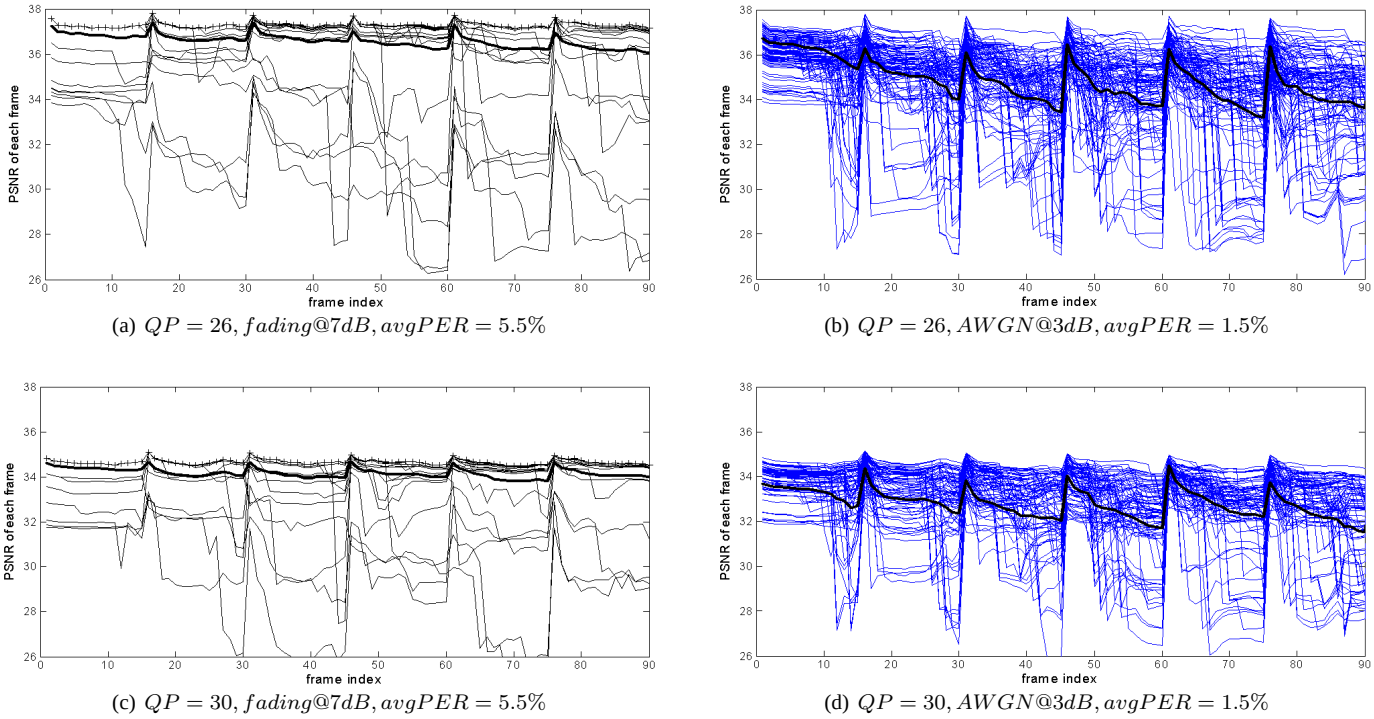
We obtain a PSNR for each frame and each packet loss pattern, for a combination of the codec parameters. Figure 5 plots the PSNRs of each frame of the video silent.cif coded at QP = 26 and 30, GOPS = 15, PS = 100 for 100 realizations of multipath fading channel of average SNR 7 dB and AWGN channel of SNR 3 dB, respectively, when PHY data rate 6 Mbps is used. The thick lines in each plot represent the average PSNRs across the 100 realizations. It is clear that even for the same video, coded using the same parameters for the same average channel SNR, the quality of concealed video in terms of PSNR varies significantly across different realizations. This is typical for all of the videos and parameters we tested. PSNRs also can vary dramatically from one frame to another in the same processed video sequence. From Figure 4 we know that for the multipath fading channel about 70% of the realizations have no packet loss. These realizations overlap and form the lines marked with "+" in Figure 5(a) and 5(c). For the AWGN channel each realization has similar PERs. However, because of the prediction employed in video coding, it is shown in Figures 5(b) and 5(d) that the realizations of similar PER can generate completely different concealed video quality. The AWGN channel with a smaller SNR does not deliver better video quality than the multipath fading channel. This suggests that neither the average PER, nor the average PSNR across all the frames and all the realizations, is a suitable indicator of the quality a video user experiences and therefore these quantities should not serve as the basis for developing or evaluating video communications schemes for WLANs.

IV. Definition of $PSNR_{r,f}$ and its correspondence to perceptual quality

In this section we propose a statistical PSNR based measure $PSNR_{r,f}$ which is defined as the PSNR achieved by $f\%$ of the frames in each one of the $r\%$ of the realizations. This definition is based on two observations that are recognized by researchers in this area [6]: 1) the frames of poor quality in a video sequence dominate human viewers' experience with the video; 2) When the PSNRs are higher than a threshold, increasing PSNR does not correspond to an increase in perceptual quality that is already excellent at the threshold. Only PSNR of the luminance component of the video sequences are considered and the peak signal amplitude picked in this paper is 255 due to 8 bit precision in the video codecs.

Parameter r captures the reliability of a channel and can be set as a number between 75% to 100% according to the desired consistency of the user experience. To study the correlation between $PSNR_{r,f}$ and the perceptual quality of videos and to find a suitable range for the parameter f, a subjective experiment is designed and conducted. Stimulus-comparison methods [4] are used in this experiment, where two video sequences of the same content were presented to the subjects side by side and were played simultaneously. The video on the left is considered to be of perfect quality while the video on the right is compressed and then reconstructed with possible packet loss and concealment. Three naive human subjects are involved in this experiment. They are asked to pick a number representing the perceptual quality of the processed video compared to the perfect video from the continuous quality scale shown on the left end of Figure 6. 50 video pairs were tested and 20% of them appear twice in this experiment to test the consistency of the subjects' decisions.

Figure 6 plots the opinion scores given by the three subjects. We find the best linear fit of average PSNRs across all the frames for each video tested and $PSNR_{r,f}$ with f ranging between 0.5 to 0.99, according to minimum mean square error. The best fits for average PSNR and $PSNR_{r,f=90\%}$ are plotted in Figure 6. As seen from these plots $PSNR_{r,f=90}$ correlates significantly better than average PSNR to the perceptual quality for all three videos. Average PSNR underestimates the quality at high quality level and overestimates the quality at low quality level. This is because average PSNR treats all frames equally, so at high quality level, only a few frames with relatively lower quality bring down the average PSNR but do not affect the perceptual quality. While at low quality level, there are frames with extremely bad quality while the average PSNR is still quite high. This subjective experiment



(a) $QP = 26, fading@7dB, avgPER = 5.5\%$

(b) $QP = 26, AWGN@3dB, avgPER = 1.5\%$

(c) $QP = 30, fading@7dB, avgPER = 5.5\%$

(d) $QP = 30, AWGN@3dB, avgPER = 1.5\%$

Fig. 5. PSNRs of each frame of the video silent.cif coded at GOPS = 15, PS = 100 for 100 realizations of multipath fading channel of average SNR 7 dB and AWGN channel of SNR 3 dB respectively, when PHY data rate 6 Mbps is used. The thick lines in each plot represent the average PSNRs across the 100 realizations which are represented by the other lines.

implies that $PSNR_{r,f}$ can serve as an effective video quality measure before more sophisticated perceptual quality measuring methods come along, and that f should be set around 90% for medium video frame rates, such as 15 fps used in this paper.

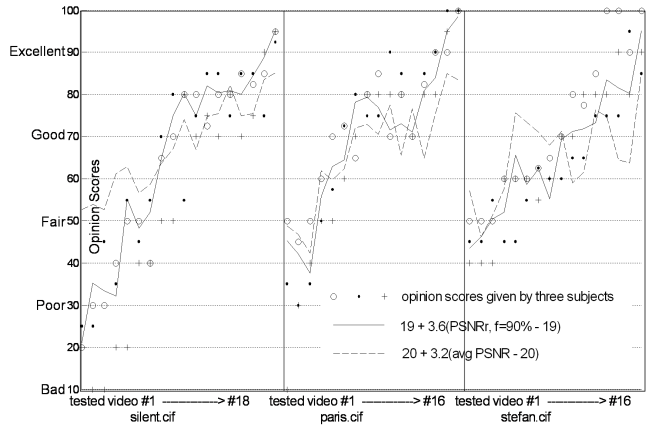


Fig. 6. Scale and results of subjective experiment

V. Discussions

$PSNR_{r,f}$ has the potential to capture the performance loss due to damaged frames in a video sequence ($f\%$), as well as to indicate how often a user, in multiple uses of the channel, would experience a specified quality ($r\%$). Figure 7 plots $PSNR_{r,f}$ for the four plots in Figure 5, with fixed $r = 85\%$, PHY data rate = 6 Mbps, channel SNR = 7 dB over the multipath fading channel, PS = 100 bytes, GOPS = 15 and the video silent.cif. The average PSNRs displayed in this figure are calculated across all the frames of all realizations. This figure shows clearly the delivered quality guaranteed for 85% of the users for different percentage of the frames. Even though the AWGN channel in this plot has a lower channel SNR than the fading channel, from Figure 5 we can see that the AWGN channel at 3 dB has an average PER of 1.5%, which is much lower than that of the fading channel at 7 dB, 5.5%. Note that the 85% realizations that are chosen for different values of f are not always the same, and therefore in our definiton the parameter r has certain dependence on the parameter f.

Figure 8 shows $PSNR_{r,f}$ for different videos, with fixed f = 80%, PHY data rate = 6 Mbps, average channel SNR = 7 dB and QP = 26, GOP = 10 and PS = 100. This figure shows that even though the average PSNRs across all the frames and realizations for all the videos at both PSs are between 32 dB to 36 dB, which imply good perceptual quality, the PSNRs achieved by 80% of the frames in 90% of the realizations are less than 26 dB for the multipath fading channel which corresponds to poor quality. With all the parameters kept as the same, stefan.cif, which is a video of a tennis player playing tennis, is the most difficult to conceal. Silent.cif which is a head-and-shoulders video is the easiest to conceal and paris.cif with two people talking to each other falls in between the other two videos in terms of motion content and performance with packet loss concealment. Some insights into comparing

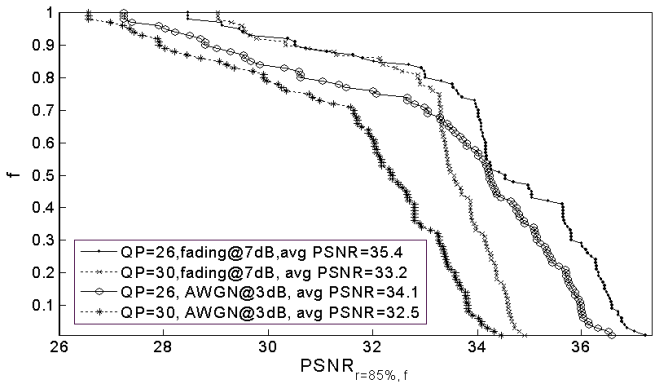


Fig. 7. Comparing $PSNR_{r,f}$ for different QPs and channel conditions, with fixed r=85%, PHY data rate = 6 Mbps, average channel SNR = 7 dB, PS = 100, GOPS = 15 and the video processed is silent.cif

AWGN and multipath fading channels are also provided by this plot. Since the fading channel delivers a certain percentage of the videos without any packet loss, its performance is always better than that of a comparable AWGN channel up to a threshold value for r, about 70% in this specific case. On the other hand there are also very bad realizations for the fading channel. As can be seen from Figure 4, about 8% of the realizations for PS = 100, fading channel at 7 dB have PLR greater than 20%. Returning to Figure 8, when r is greater than 92%, the performance of AWGN channel is definitely better than a comparable multipath fading channel. When r falls between 70% and 92%, i.e., when the fading channel realizations have PLR greater than 0% but less than 20% from Figure 4, we can see in Figure 8 that as r increases, the quality of delivered video over the fading channel decays faster than that over the AWGN channel. The interplay of the coding parameters on the processed video quality are discussed in [16].

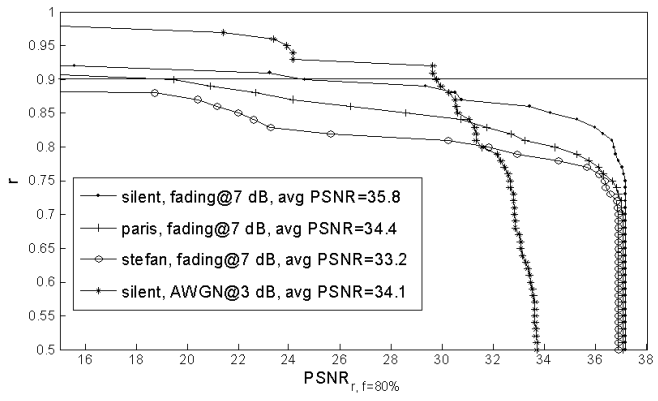


Fig. 8. Comparing $PSNR_{r,f}$ for different videos and PSs, with fixed $f = 80\%$, PHY data rate = 6Mbps, channel SNR = 7dB and QP = 26, GOP = 10

VI. Conclusions and Future Work

In this paper we investigate the delivered quality of AVC/H.264 coded video across the video characteristics, the quantization parameter (QP), the group of picture size (GOPS), the payload size (PS), PHY data rate in 802.11a, and average channel signal to noise ratio (SNR), for AWGN and multipath fading channels. We show that for the same video coded using the same parameters for the same average channel SNR, the quality of concealed video varies significantly across different realizations. The PSNRs also vary from one frame to another in the same processed video sequence. Neither the average PER nor the average PSNR across all the frames and all the realizations, is a suitable indicator of the quality a video user experiences and therefore they should not serve as the basis for video communications quality assessment.

We define a statistical video quality indicator $PSNR_{r,f}$ as PSNR achieved by $f\%$ of the frames in each one of the $r\%$ of the realizations. We show that $PSNR_{r,f}$ agrees consistently with perceptual video quality through a subjective experiment. We employ $PSNR_{r,f}$ to evaluate video communications performance under various channel conditions and to select the best combination of codec parameters at certain desired consistency of video user experience.

Future work will include more subjects in the subjective experiment to construct a nonlinear relationship between the opinion scores and $PSNR_{r,f}$.

References

[1] "The quest for objective methods: Phase II, final report," *Video Quality Experts Group, http://www.its.bldrdoc.gov/vqeg/*, Aug. 2003.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, July 2003.

[3] I.-T. R. P.910, *Subjective video quality assessment methods for multimedia applications*, Std.

[4] "Methodology for the subjective assessment of the quality of television pictures," *ITU-R Recommendation BT.500*, 2002.

[5] T. N. Pappas and R. J. Safranek, "Perceptual criteria for image quality evaluation," *Handbook of Image & Video Processing (A. Bivok eds.), Academic Press*.

[6] Z. Wang, H. R. Sheikh, and A. C. Bovik, "Objective video quality assessment," *The Handbook of Video Databases: Design and Applications (B. Furht and O. Marqure, eds.), CRC Press*, pp. 1041–1078, Sept. 2003.

[7] "H.264/AVC software coordination - reference software JM10.1," *http://iphome.hhi.de/suehring/tml/*, 2006.

[8] R. van Nee and R. Prasad, *OFDM for Wireless Multimedia Communications*. Artech House, Jan 2000.

[9] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*. Sams, December 2001.

[10] D. Qiao, S. Choi, and K. G. Shin, "Goodput analysis and link adaptation for IEEE 802.11a wireless LANs," *IEEE Trans. on Mobile Computing (TMC)*, vol. 1, no. 4, 2002.

[11] S. Choudhury and J. Gibson, "Payload length and rate adaptation for throughput optimization in wireless LANs," *To appear in IEEE Vehicular Technology Conference (VTC)*, May 2006.

[12] ——, "Joint PHY/MAC based link adaptation for wireless LANs with multipath fading," *To appear in Wireless Communication and Networking Conference (WCNC)*, April 2006.

[13] M. van der Schaar, S. Krishnamachari, S. Choi, and X. Xu, "Adaptive cross-layer protection strategies for robust scalable video transmission over 802.11 WLANs," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1752–1763, Dec. 2003.

[14] X. Zhu, E. Setton, and B. Girod, "Congestion-distortion optimized video transmission over Ad Hoc networks," *EURASIP'05*.

[15] N. Chayat, "Tentative criteria for comparison of modulation methods," *IEEE P802.11-97/96*, Sept. 1997.

[16] J. Hu, S. Choudhury, and J. D. Gibson, "H.264 video over 802.11a wlans with multipath fading: Parameter interactions and delivered quality," *submitted to Globecom*, Nov. 2006.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

Error Propagation After Concealing a Lost Speech Frame

Christian Hoene
University of Tübingen
Germany
hoene@ieee.org

Ian Marsh
KTH Stockholm
Sweden
ianm@sics.se

Günter Schäfer
University of Illmenau
Germany
guenter.schaefer@tu-illmenau.de

Adam Wolisz
Technical University of Berlin
Germany
awo@ieee.org

***Abstract*— Depending on the content of speech frames, the quality impairment after their loss differs widely. In previous publications we described an off-line measurement procedure to determine the loss impairment – *the importance* – of single speech frames. We showed that knowing the importance of frames can enhance the transmission performance of VoIP telephones significantly if only important frames are transmitted.**

Here we study to what extend the importance can be calculated at real-time: The loss impairment is due to the imperfect packet loss concealment (PLC) and also due to error propagation (EP). EP originates from the desynchronisation of the decoder's internal state and cannot be calculated at real-time. We developed a measurement method to determine the effect of the imperfect PLC and the temporal progression of the error propagation. The results show the trade-off between algorithmic delay and the accuracy of real-time importance calculation: A good frame classification needs to look ahead 20-40 ms in order to calculate the importance precisely.

I. Introduction

Packet losses significantly decrease the quality of voice communications. Usually, packet loss rate and speech quality are considered to be closely related. However, this ignores the fact that speech frames differ significantly. For example, it is well known that speech transmission can be interrupted during silence because silent speech frame have a minor impact on the quality of speech transmission. Active speech frames differ, too: Human speech generates two types of sounds: voiced and unvoiced. Voiced sounds have a regular pattern and usually high energy (e.g. "a","o", ...). Unvoiced sounds have a random nature (e.g. "h","sh", ...). Actually, one third of all active frames can be dropped while maintaining speech intelligibility [1]. However, only the right, more precise, the irrelevant frames are allowed to be dropped. Identifying irrelevant or important frames is a non-trivial task. Parts of this problem are addressed in this paper.

If a speech frame is lost, the receiver tries to extrapolate the last successful received frame to limit the impact of the lost frame. Such algorithms are known as packet loss concealment. Nowadays, they are often standardized and part of the decoder. A lost frame causes the current speech period to become distorted as the receiver's PLC cannot fully reconstruct the lost frame. Thus, the concealed frame differs from the sent frame and hence introduces a *loss distortion.*

Low-rate speech coders that transmit only signal differences suffer from an additional effect: If a frame is lost, the decoder becomes desynchronized [2]. If the internal state of the decoder does not match the encoder's state, the decoding of the following frames is affected and an additional distortion is introduced. We refer to this effect as *error propagation* (Figure 1). This effect is well known from digital, compressed TV and video transmissions. A transmission error causes the video signal to be distorted for a long period that can even last multiple video frames.

In [3] we presented a off-line method of how to determine the impact of an individual frame's loss – called the *importance of a frame*. It considers both loss distortion and error propagation. Here we extend this method to quantify the impact of the loss distortion and temporal progression of error propagation by studying common narrow-band speech codecs. Our results show that the frame following the loss contains the larger amount of the error propagation.

This results are important for the development of a real-time algorithm to classify speech frames: We can measure the amount of loss distortion at real-time [4]. But we cannot foresee the amount of error propagation because it depends on the following speech, which has not be spoken yet. Thus, a perfect frame classification must know the future. Any algorithm which does not know the future or cannot predict the amount of error propagation is less precise. So to say, this work shows the maximal achievable accuracy of real-time classification algorithms.

This paper is structured as follows: We start with a background and related work section. Then, we describe error propagation in narrow-band speech codings. Next, we present our measurements on quantifying the amount of error propagation. Finally, we conclude.

II. Related work and Background

A. Speech quality

The perceived quality of a telephony call can be measured with subjective tests. Humans evaluate the quality of service according to a standardized quality assessment process [5]. Often the quality is described by a *mean opinion score (MOS)* value, which ranges from 1 (bad) to 5 (excellent). More precisely, values which origin from passive human test results are called MOS-Listening Quality Subjective (MOS-LQS). In listening-only tests usually speech samples are used, which have a lengths ranging from 6 to 12 s. Listening-only tests

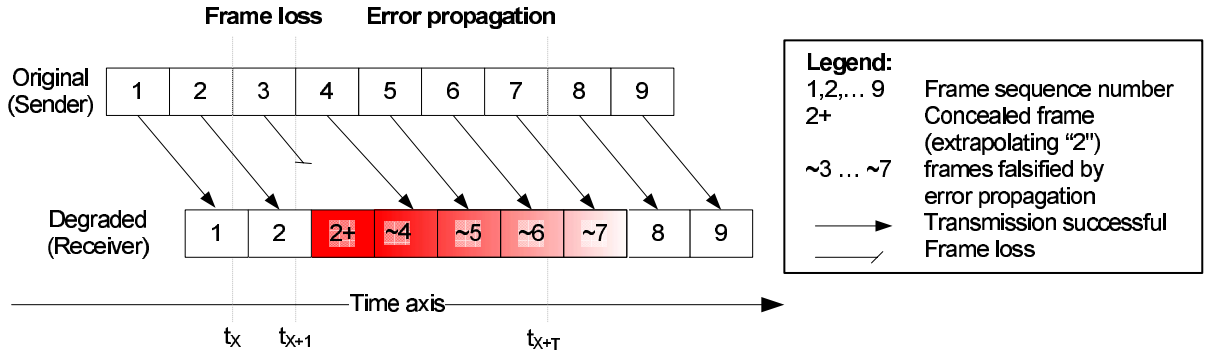


Fig. 1. Consequences of losing a frame.

are time consuming because many subjects have to be asked. Thus, in the last few years considerable effort has been made to develop instrumental measurement tools, which predict human rating behaviour.

The *Perceptual Evaluation of Speech Quality (PESQ)* algorithm predicts human rating behaviour for narrow band speech transmission [6]. It compares an original speech fragment with its transmitted and thus degraded version to determine an estimated MOS-LQO (Listening Quality Objective) value. Benchmark tests of PESQ have yielded an average correlation of R=0.935 with human based, subjected tests.

The correlation coefficient is often used to compare speech quality scores of human and instrumental predictions (e.g. PESQ). A value of R=1 would be a perfect match between both score sets, whereas R=0 means no correlation at all. A positive behaviour of correlation means that it is not influenced by linear scaling or adding an offset: Any linear regression applied to sets of measurement data does not change the value of R at all.

B. Real-time classification of speech frames

Petr et al. [7] suggested a method to mark speech frames containing background noise with the lowest priority. The next higher priority is assigned to voiced speech segments, which are not at the beginning of the voiced sounds. The next higher priority is assigned to non-initial fricative (e.g. the "ch" in the German word Bach). All other frames including the initial voiced and fricative speech segments are marked with the highest priority.

De Martin [8] has proposed an approach called Source-Driven Packet Marking, which controls the priority marking of speech packets in a DiffServ network. If packets are assumed to be perceptually critical, they are transmitted in a premium traffic class. All other packets are sent using the best-effort traffic class. The author describes a packet-marking algorithm for the ITU G.729 codec. For each frame, it computes the expected perceptual distortion, as if the speech frame were lost, under the assumption that no previous speech frames were lost. First, only speech frames with at least a minimal level of energy are considered to be marked as premium. Next, the marking algorithm takes the coding parameters (e.g. the gain, linear prediction filter, codebook indexes) and computes the parameters that would be computed by the concealment algorithm if the packet was lost. It then compares both parameter sets – the original and the concealed – in order to compute the perceptual quality degradation in case of loss.

Petracca and De Martin [9] presented a classification of AMR frames. Their analysis-by-synthesis distortion evaluation algorithm calculates the spectral distortion in dB for the LP coefficients, the percentage difference for the long-term prediction coefficients and the difference in dB for the codebook gains. If any of these values is above a given threshold, an AMR frame is marked as premium. De Martin's frame classifications do not consider any error propagation effects.

Sanneck et al. [10] analyzed the temporal sensitivity of VoIP flows if they are encoded with μ-law PCM and G.729: Single losses in PCM flows have a small sensitivity to the current speech properties. Multiple consecutive losses have a higher impact on the quality degradation than single, isolated losses. The concealment performance of G.729, on the other hand, largely depends on the change of speech properties. If a frame is lost shortly after an unvoiced/voiced transition, the internal state of the decoder might be de-synchronized for up to the next 20 following frames.

Rosenberg et al. [2] measured the length of desynchronisation after losing a G.729 frame. Our work extends these initial results, includes other codecs and enhances the accuracy of the measurement procedure.

C. The Importance of Individual Speech Frames

In [3] Hoene et al. describe an off-line measurement procedure, which measures the impact of loss on speech quality and quantifies the importance of frames. They used this method in an extensive experiment effort evaluating more than two million different, deliberately simulated packet and frame losses. Hereby they considered the most common standardized, narrow-band speech codecs and concealment algorithms, which are Adaptive Multi-Rate (AMR), G.711 plus Annex I[1], and G.729. Also, they validated their method with formal listening-only tests [11].

[1]Its "frame" length is set to 10 ms.

In [1] Hoene et al. developed an quality metric to describe the importance of speech frame or VoIP packets. Under many conditions this metric shows an additive property of equality. Thus, is it possible to to give a statement like “frame A and frame B are as important as frame C” or “frame A is three times more important than frame B”. The metric’s definition is given as: *The importance of frame losses is the difference between the quality due to coding loss and the quality due to coding loss plus frame losses, multiplied by the length of the sample.* The following equation describes how to calculate the importance. For a given sample s that has a length of $t(s)$, a given codec implementation c, and a loss event described with e $\mathrm{MOS}(s,c)$ describes the speech quality due to coding loss, and $\mathrm{MOS}(s,c,e)$ describes the speech quality due to coding as well as frame loss.

$$\begin{gathered} Imp(s,c,e) = (cl - c) \cdot t(s) \\ \text{with } cl = (4.5 - MOS(s,c,e))^2 \\ \text{and } c = (4.5 - MOS(s,c))^2 \end{gathered} \tag{1}$$

In this paper, we extend the off-line measurement procedure and use (1) to quantify the importance.

III. Real-time packet classification

To control the transmission of speech frames, their importance should be known at transmission time. For example, in addition to the encoding of speech, the sender could calculate the importance of each speech frame. This leads to the question, is it possible to predict the importance of speech frames at transmission time? In general, the consequences of packet loss can be split into two effects (Figure 1):

First, the lost frame is concealed at the receiver, which causes a distortion if the concealment does not perfectly predict the frames content. In the illustration this refers to frame 3 (transmitted) and frame 2+ (concealed). The encoder knows the original and degraded speech segment. It can also predict the behaviour of the decoder in case of loss, as the decoder’s concealment algorithm is known (since it is standardized). In principle, the encoder can therefore calculate the impact of imperfect concealment.

The second effect of packet loss is due to error propagation. After a frame loss the internal state of the concealment algorithm is desynchronized. The impact of error propagation cannot be known at the time of transmission because the length of error propagation depends on the following speech content. In case of interactive telephony the following speech has not yet been spoken. Thus, predicting the importance of a speech frame at run-time will always be falsified by the effect of error propagation.

In Figure 2, we display how long it takes until synchronisation of the decoder is achieved. We measure desynchronisation lengths for the ITU G.729 coding, which last up to 650 ms.

To demonstrate the impact of imperfect packet loss concealment and error propagation we plotted the speech signals of a sample segment in Figure 3 for different encoding schemes. Beside the original sample, the figures also contain the encoded/decoded (=degraded) signal, the encoded/lost/decoded/concealed signal, and the difference between those signals. Also, the figures contain the PESQ MOS values to quantify the perceptual impact of coding and concealment degradation.

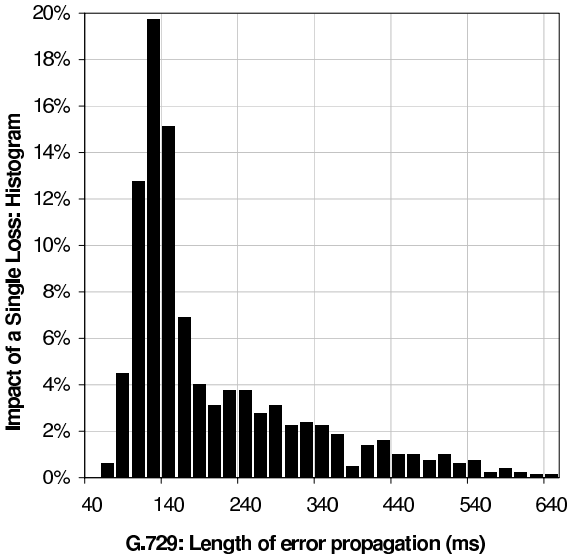


Fig. 2. Histogram of error propagation lengths in case of loss of one G.729 frame. We measured the time until the internal state of the G.729 decoder matches the non-loss state again. The decoders’ post-filter is ignored as it does not synchronise again.

IV. Quantifying error propagation

A. Method

The aim of this paper is to quantify the imperfect concealment and error propagation caused by a single frame loss. The question arises how should the effects be measured? The speech sample could be split into two parts. The first part contains the content until the end of the concealed frame (e.g. frame 1, 2, 2+). The next part contains the remaining content (e.g. frame ~4 to 8). The position of the split is exactly after concealing and decoding the lost frames. Thus, the effect of concealment and the effect of error propagation are separated into two samples. For both samples the degradation can be measured with PESQ and compared to the corresponding samples that do not contain any frame loss. This method is problematic due to two reasons.

First, PESQ judges the speech quality largely different if the sample content differs. Thus, splitting the sample and thus changing the sample’s length introduces a source of error. Instead, the sample content must not be changed.

Second, a hard split between two samples introduces an additional clicking sound, which falsifies the results.

Therefore, we developed the following measurement procedure (see Figure 4). We generate two samples containing first the degraded sample without loss and second, the degraded sample with one frame loss. Then, we mix both samples to

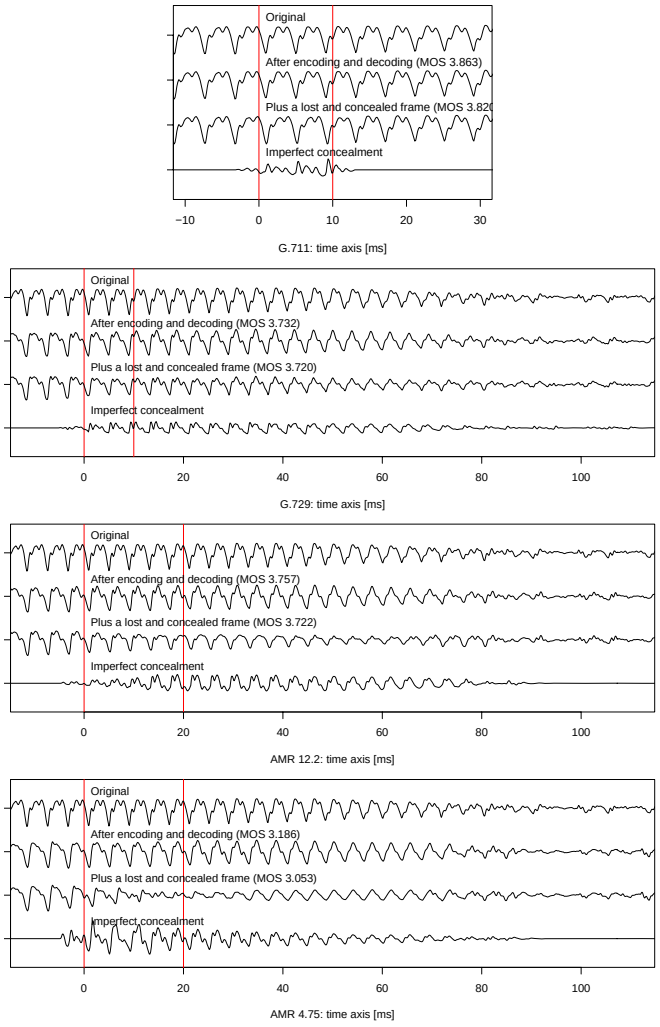


Fig. 3. Speech signals before and after decoding, after loss concealment, and the difference between the decoded and concealed signals. The two vertical lines define the length of the frame loss.

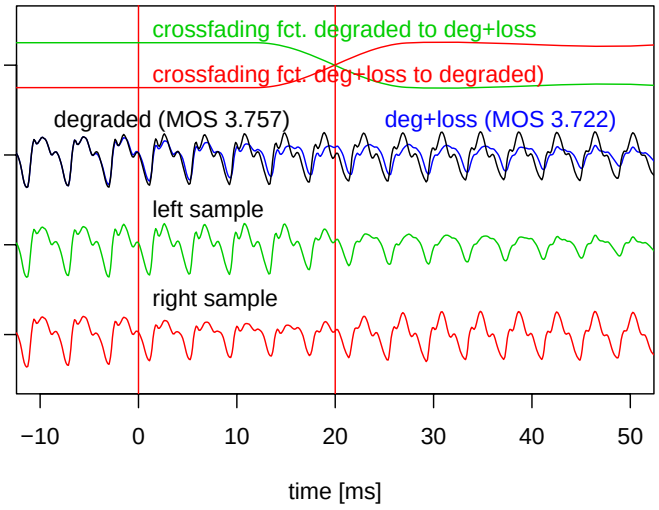


Fig. 4. Splitting the imperfect concealment and error propagation into two different speech samples. The position of the lost frame is marked with two vertical, red lines. Two degraded samples are generated, with loss (blue) and with-out loss (black). Then, to get the impact of PLC we crossfade from the loss to the no-loss sample to produce a new speech sample called “left”. Similar, to get the impact of EP we crossfade from the no-loss to loss sample to produce a new speech sample called “right”.

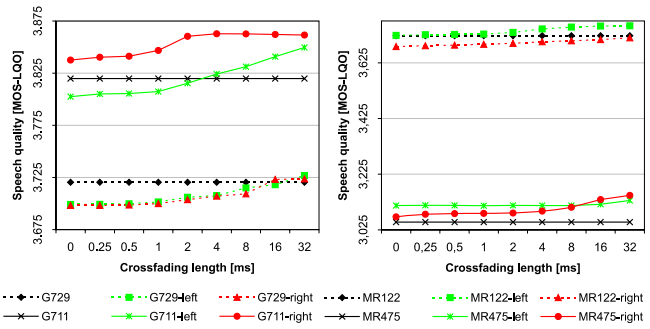


Fig. 5. Impact of crossfading length on speech quality.

produce new samples: We crossfade just after the lost frame (right vertical line in Figure 4). The crossfading function is a cosine curve. Then, two new samples are produced. The first called “left” contains the concealment frame and the second called “right” contains the error propagation. The speech quality of those samples is then measured with PESQ.

This algorithm leads to another question: How long should this crossfading period be? For one test condition containing one frame loss we conducted measurements with varying crossfading lengths (see Figure 5).

The black lines represent the speech quality considering imperfect concealment and error propagation. If the crossfading is done in less than 4 ms, it introduces an addition distortion that lowers the speech quality. However, if the crossfading is too slow, the short effect of a single frame loss is smeared over the left and right samples. Thus, we will use a crossfading length of 4 ms in the following work.

If in addition the split is conducted not only at the end of lost frame (refer to as position 0 ms) but also at positions shortly after the lost frame, we can observe the temporal progression of the error propagation.

B. *Experimental set-up*

In order to study the impact of frame losses on the speech quality, we conducted experiments as depicted in Figure 6 and described in [1], [3]. We used speech recordings, taken from an ITU coded speech database [12] that consists of 832 files, each 8 seconds long, with 16 different speakers, 8 female and 8 male, spoken in four different languages, without any background noise. We chose this database to limit the influence of specific languages [13], speakers, or samples. We chose three common narrow-band-speech-coding algorithms: ITU’s G.711 and G.729, and ETSI’s Adaptive-Multirate (AMR).

We simulated packet losses at different positions within the sample. We varied the coding scheme, the packet loss positions and the sample content, and generated for each test case a

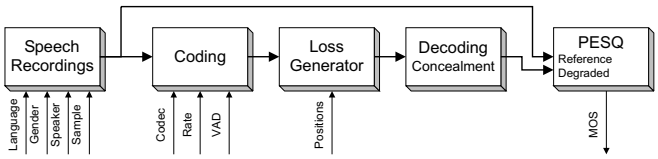


Fig. 6. Measuring one frame loss.

degraded audio. In addition, we split the sample as described above. To assess the speech quality we applied the ITU's PESQ algorithm [6] to calculate a MOS value. Some million PESQ rating results were gathered to achieve a high accuracy for statistical analysis.

C. Results

For each test condition we calculate the distribution of importance values. In Figure 7, we display the importance value of the left and right parts of the loss distortion. Actually, we choose to display the 75% percentile as it is close to the median importance of all active speech frames. In addition, the sum of the left and right importance values are displayed since the importance metric is to some extent additive.

The first graph containing G.729 values shows that this codec has a high amount of error propagation and it takes approximately 80 ms until this effect disappears. The next graph using G.711 is to demonstrate the quality of our measurement procedure as in case of G.711 the error propagation is fixed to a maximal length of at most 3.75 ms [14]. It shows that our measurement procedure does not split perfectly both distortion effects, but has an inaccuracy of 0–10 ms. Last, the values for AMR coding are shown. The amount of error propagation is small and disappears after 20–40 ms.

D. Analysis

Coming back to the main question of this paper: How well can the importance of a speech frame can be predicted in real-time? As a performance metric we will calculate the Pearson's correlation coefficient of the offline, reference importance values and the left, right, and both (left+right) importance values.

In Figure 8 we display the correlation to compare the importance value sets of the reference (offline), left, right, and both measurements. If only the imperfect concealment is considered to calculate importance values, the performance for G.729 coding is R=0.72, for G.711: R=0.91, and for AMR: R=0.73. If in addition the next frame after the concealed frame is considered, the performance increases to G.729: R=0.92, G.711: R=1.00, and AMR: R=0.97. Given the precision of our measurement procedure (R=0.94, [11]), the later results are almost perfect.

V. Conclusions

Given the knowledge of packet importance, we showed that significant performance gains can be achieved if only packets are transmitted with priority that are important. However, the importance of speech frames has to be known precisely, otherwise this performance gains are lost [15]. The importance of a packet can be measured both off-line and in real-time. A measurement procedure that identifies the impact of a single frame loss offline has already been developed and has been verified with formal listening-only tests in previous publications.

In this paper, we studied how the importance can be measured in real-time. This is difficult, as the importance values partially depend on the amount of error propagation which is not known at the time of transmission. Waiting for the next frame before calculating the importance value significantly increases the accuracy of the importance predictions. The enhancement comes at the cost of an increased algorithmic delay. A good compromise is a look ahead of 20 to 40 ms to minimize error propagation effects.

References

[1] C. Hoene, S. Wiethoelter, and A. Wolisz, "Calculation of speech quality by aggregating the impacts of individual frame losses," in *Thirteenth International Workshop on Quality of Service (IWQoS 2005)*, Passau, Germany, June 2005.

[2] J. Rosenberg, "G.729 error recovery for internet telephony," Columbia University Computer Science, Prof. H. Schulzrinne, New York, NY, Tech. Rep. CUCS-016-01, Dec. 2001.

[3] C. Hoene, B. Rathke, and A. Wolisz, "On the importance of a VoIP packet," in *ISCA Tutorial and Research Workshop on the Auditory Quality of Systems*, Mont-Cenis, Germany, Apr. 2003.

[4] F. D'Agostino, E. Masala, L. Farinetti, and J. De Martin, "A simulative study of analysis-by-synthesis perceptual video classification and transmission over diffserv IP networks," in *IEEE International Conference on Communications (ICC '03)*, vol. 1, 2003, pp. 572–576.

[5] *Methods for subjective determination of transmission quality*, Recommendation P.800, ITU-T Std., Aug. 1996.

[6] *Perceptual Evaluation of Speech Quality (PESQ), an Objective Method for End-To-End Speech Quality Assessment of Narrowband Telephone Networks and Speech Codecs*, Recommendation P.862, ITU-T Std., Feb. 2001.

[7] D. Petr, J. DaSilva, L.A., and V. Frost, "Priority discarding of speech in integrated packet networks," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 5, pp. 644–656, 1989.

[8] J. C. De Martin, "Source-driven packet marking for speech transmission over differentiated-services networks," in *Proceedings of the IEEE International Conference on Audio, Speech and Signal Processing*, Salt Lake City, UT, May 2001, pp. 753–756.

[9] M. Petracca, A. Servetti, and J. C. De Martin, "Voice transmission over 802.11 wireless networks using analysis-by-synthesis packet classification," in *First International Symposium on Control, Communications and Signal Processing*, Hammamet, Tunesia, Mar. 2004, pp. 587–590.

[10] H. Sanneck, N. Tuong, L. Le, A. Wolisz, and G. Carle, "Intra-flow loss recovery and control for VoIP," in *Ninth ACM international conference on Multimedia (MULTIMEDIA '01)*. New York, NY: ACM Press, 2001, pp. 441–454. [Online]. Available: citeseer.nj.nec.com/article/sanneck01intraflow.html

[11] C. Hoene and E. Dulamsuren-Lalla, "Predicting performance of PESQ in case of single frame losses," in *Measurement of Speech and Audio Quality in Networks Workshop (MESAQIN)*, Prague, CZ, June 2004.

[12] *Coded-speech Database*, Recommendation P.Supplement 23, ITU-T Std., Feb. 1998.

[13] D. Goodman and R. Nash, "Subjective quality of the same speech transmission conditions in seven different countries," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '82)*, vol. 7, 1982, pp. 984–987.

[14] *A High Quality Low-Complexity Algorithm for Packet Loss Concealment with G.711*, Recommendation G.711 Appendix I, ITU-T Std., Sept. 1999.

[15] C. Hoene, "Internet telephony over wireless links," Ph.D. dissertation, Technical University of Berlin, TKN, 2005.

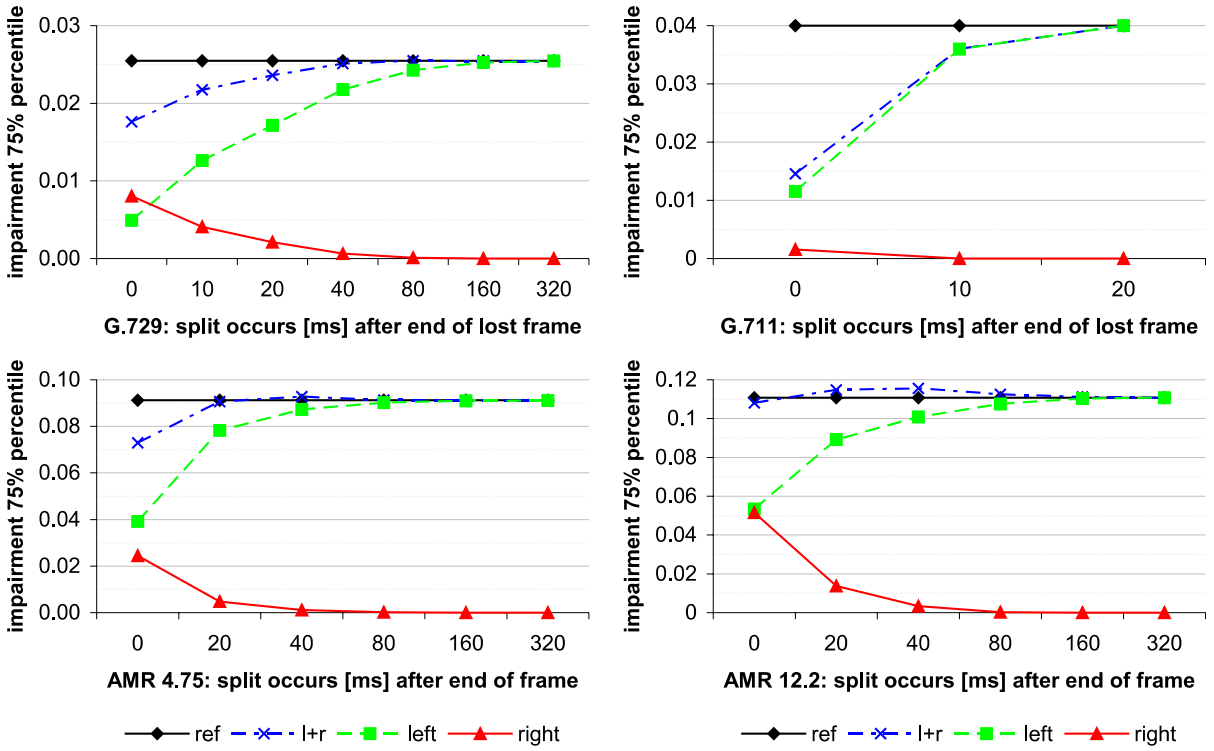


Fig. 7. Temporal impact of error propagation on the importance of frame losses.

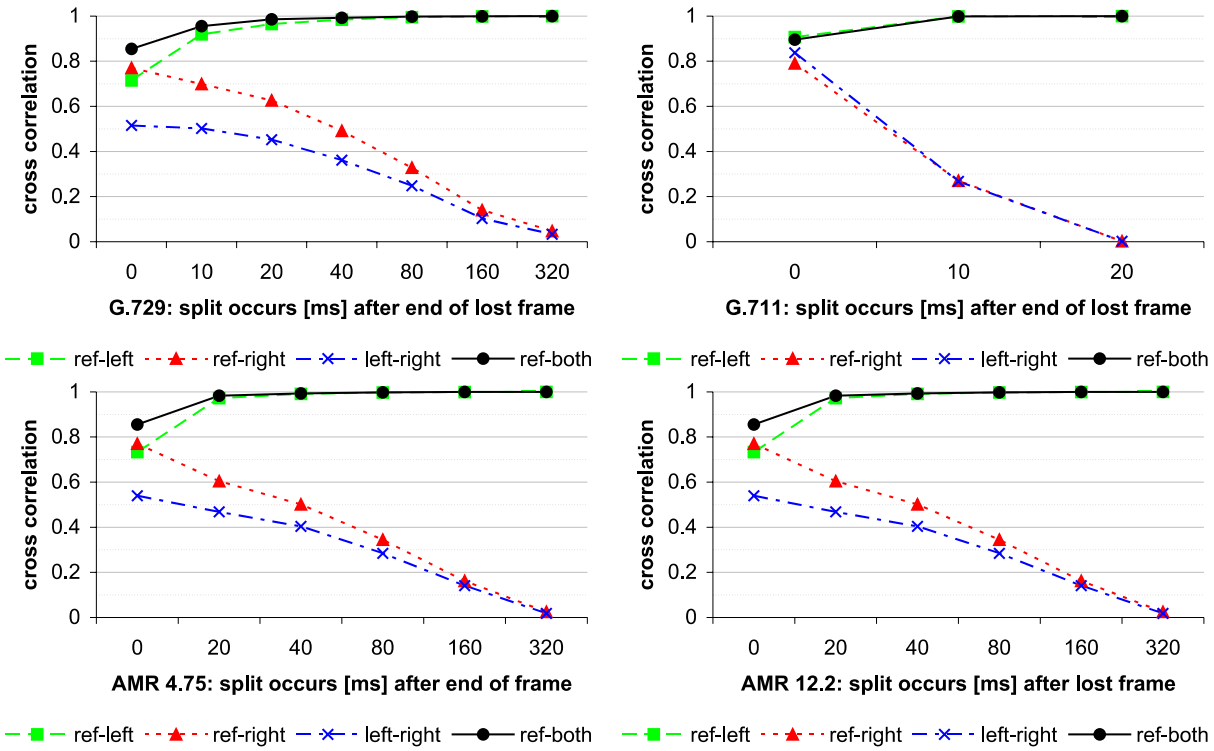


Fig. 8. How well can the frame importance be predicted if some X milliseconds of the following speech is considered in addition to concealment effect? (This result is displayed in the “ref-left” line.)

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

A Performance Study of VoIP Applications: MSN vs. Skype

Wen-Hui Chiang, Wei-Cheng Xiao, and Cheng-Fu Chou
Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
Email: {whchiang, garry, ccf}@cmlab.csie.ntu.edu.tw

Abstract—**Due to the growing demand for VoIP (Voice over Internet Protocol) services, researches on VoIP design have attained more and more attention. Compared with a traditional voice service - PSTN (Public Switched Telephony Network), VoIP is able to provide lower cost and more flexibility. However, there are still many challenging issues to guarantee a consistent and good quality of voice connection over the best-effort Internet. In this work, we use a measurement-based approach to do quantitative evaluation of two most popular VoIP applications, i.e., MSN and Skype. In general, Skype performs better than MSN --- it shows that an up to 47% improvement in the overall throughput and an up to 50% improvement in the MOS. Such performance improvement for Skype is due to its (a) rate control mechanism, (b) voice codec, (c) error-resilience mechanism, and (d) better relaying mechanism to go through NAT servers or firewalls. We believe that this study can be of great use in designing a better voice service in current or next-generation heterogeneous networks.**

Keywords-VoIP; Codec; Skype; MSN; P2P

I. INTRODUCTION

VoIP service is a rapidly emerging technology for voice communication. Different from traditional PSTN, it has several advantages including cost saving, flexibility, and better voice quality. These properties have led to growing demand for development of better VoIP services. Furthermore, recent research works [4][5][6][7] on VoIP design have attained more and more attention.

We note that voice quality is affected by not only bandwidth but also other potential pitfalls. The poor voice quality in PSTN might result from the poor connections or old cables. On the other hand, the voice quality in VoIP is mostly dominated by the characteristics of packet networks such as delay, jitter, and packet loss. Therefore, it is important for us to take characteristics of IP network into consideration when we design a VoIP application.

Skype is able to provide good voice quality under unstable or resource-constrained networks and often outperforms other VoIP applications, e.g., MSN. Now, an interesting problem arises: which mechanism makes Skype superior? Both MSN and Skype provide a variety of functions such as voice calls, instant messaging, audio conferencing, and buddy list. However, their underlying techniques and protocols could be quite different. For example, in [1], they point out that the Skype mechanism, which is used to pass through the NAT, can easily adapt to port constraints on firewalls. Moreover, the wideband codec of Skype results in substantial improvement for the voice quality.

In this work, we use a measurement-based approach to evaluate the performance of MSN and Skype. We first create a voice connection of MSN or Skype between two hosts. In the meanwhile, ethereal is used to collect the online traffic for analysis. To simulate real world traffic, dummynet is used to generate the required bandwidth, add the artificial propagation delay, and provide the specific packet loss rate. Through the experiments, we would like to observe how MSN and Skype react to different network conditions, such as varying bandwidth, different packet loss rates, etc. Moreover, by carefully analyzing the collected data, we can explain which underlying technique, the path selection, or the codec mechanism dominates the performance issues. Hence, we are able to figure out which technique has great impact on the voice quality.

The contributions of this work are as follows. We present a comprehensive study which compares the performance and adaptation characteristics of MSN and Skype. In general, the Skype outperforms MSN. Under public networks, the throughput improvement is at least 13% and the MOS improvement can be up to 50%. In addition, when both hosts are behind NAT servers, the throughput difference is up to 47% or more. Such performance improvement for the Skype application is due to its (a) rate control mechanism, (b) voice codec, (c) error-resilience mechanism, and (d) better relay mechanism for traffic passing NAT servers or firewalls. We believe that such evaluation is important and can be of great help in designing a better voice service across Internet or next-generation heterogeneous networks.

The organization of the paper is as follows. Brief reviews of Skype as well as MSN are given in Section 2. Section 3 describes the details of our measurement-based approach and performance evaluation and comparison. We give some related works in Section 4, and then conclude this paper in Section 5.

II. MEASUREMENT-BASED APPOACHES

In this study, we aim to observe how the performance of VoIP applications is affected by different network conditions. To achieve this goal, we design measurement-based experimental approaches. We use the ethereal to collect voice packets from real VoIP voice connections. Besides, the dummynet is used to simulate the real network conditions such as the bandwidth constraint and events of packet loss. So, we can analyze the collected data and understand the important factors that greatly influence the VoIP design.

A. Experimental Environment

In the following experiments, we use Skype of version 1.3 and MSN of version 7.0. Both Skype and MSN run on Microsoft Windows XP and machine with Intel Pentium 4 1.8 GHz CPU and 256 MB DDR DRAM. In addition, Ethereal of version 0.10.12 is used for packet dumping, and FreeBSD 5.4 is used for dummynet settings.

The experiment environment is shown in Fig. 1. We perform the experiments under two different scenarios. In the first scenario, both clients are assigned public IP addresses. In the second scenario, one is assigned a private IP address while the other is assigned a public IP address. In both scenarios, we use dummynet to simulate required network conditions, such as bottleneck capacity, network delay and events of packet loss. With above mechanisms, we can simulate the real network environment and observe how Skype and MSN react to different network conditions. Ethereal is installed as well to monitor the traffic for analyzing. We also record the conversation in each experiment and the recorded data will be graded based on the perception of voice quality. Note that all experiments were performed from August 15th to August 31st, 2005.

B. Measured Experiments

To observe the effect caused by a single factor in the VoIP service, i.e., to isolate the effect from other factors, we design 3 different experiments. In the first experiment, public IP addresses are assigned to both clients. The second experiment is similar to the first one while we add the background traffic into the network to compete with the voice connection. Different from the first two experiments, in the final one, a public IP and a private IP are respectively assigned to these two clients. We do analyses and compare the performance of Skype and MSN in all these experiments.

1) Direct Connection – Both Clients with Public IP:

This is a simple case in the Internet. Clients with public IP can connect with each other directly. In this case, we focus on the effect of codec and basic design issues such as the rate control and the error recovery mechanisms.

This experiment is performed in LAN. In the beginning, we setup a bottleneck link to observe how Skype and MSN react to the bandwidth-constrained network. In other words, the goal is to examine the rate control mechanism of Skype and MSN. Next, packet loss events are manually generated to test the resilience mechanism; that is, to see whether these applications can maintain the acceptable voice quality with the codec recovery and retransmission mechanisms. To avoid the transient behavior of the environment, we wait for at least 3 minutes between two conjunctive experiments.

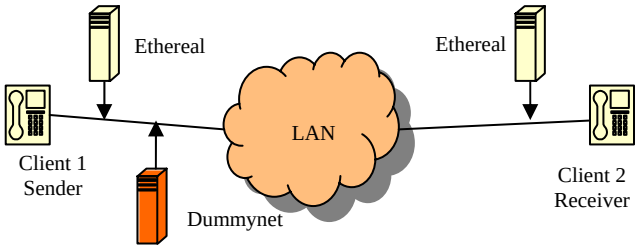


Figure 1. Experimental environment

2) Direct Connection with Background Traffic – TCP-Friendly or not:

In this experiment, we observe that the voice connections compete resources of bandwidth with FTP sessions. The motivation is to examine whether the Skype or MSN voice connection is TCP-friendly or not. In the beginning, we start a FTP session, which occupies most of the link resources. After a while, a Skype or MSN voice connection is then set up to see how it reacts to the FTP session.

3) Connection through NAT – One with Public IP and the other with Private IP:

Owing to the shortage of IP addresses, the number of private network is increasing. Thus, the problem - how to go through NAT - becomes an important issue now. This testing case is similar to the "Direct Connection" experiment while the difference is that one of the clients uses a private IP. Through this experiment, we would like to investigate the impact caused by NAT on the performance of Skype and MSN.

III. EVALUATIONS

A. Experimental Settings

The experiment environment is illustrated in Fig. 1. We do the experiments under various environment settings with path capacity from 32 Kbit/s to unlimited and with path loss rate ranging from 0% to 40%.

B. Performance Metrics

The performance metrics used in our experiments includes 1) throughput, 2) mean, variance and distribution of packet inter-arrival time, and 3) MOS (Mean Opinion Score) [3].

To some extent, throughput can reflect the bandwidth requirement and the achieved voice quality of a VoIP application. For an application, higher throughput often results in better quality. From the distribution of packet inter-arrival time, we can see if the traffic generated by Skype or MSN is bursty or stable. In addition, we invited 20 students to grade the received voice quality by MOS, in which the score is ranged from 1 (unacceptable) to 5 (excellent). The MOS results directly reflect voice quality, which almost determines user's will to use the application.

C. Direct Connection

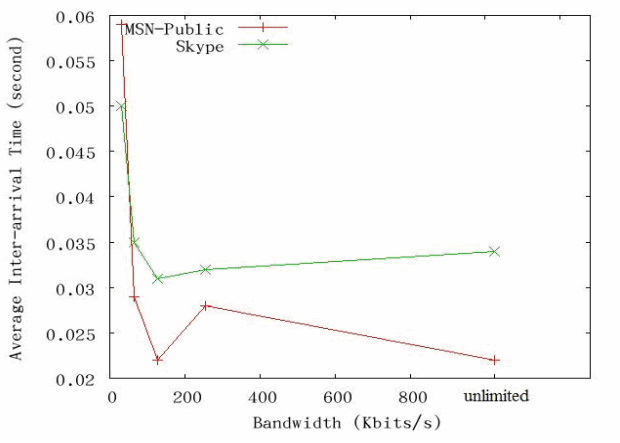
In this experiment, both clients use a public IP to set up a direct voice connection. Specifically, we would like to investigate and compare the performance of (1) the rate control mechanism and (2) the error resilience mechanism in both Skype and MSN applications under different network conditions.

1) Rate Control Scheme – The influence of different bandwidth: The motivation of this experiment is to measure the performance of Skype and MSN under bandwidth-constrained networks. As shown in Fig. 2(a), when the available bandwidth is higher than the requirement of voice data (e.g., 64 Kbits/s or higher), Skype has larger average packet inter-arrival time than MSN does. This is also illustrated in the Fig. 3. For example, as the bottleneck bandwidth is higher than 64Kbit/s, almost 100% of MSN packets have their inter-arrival time less than 50ms while only around 90% of Skype packets do. Of course, as the bottleneck bandwidth is less than the requirement, e.g., 32 Kbit/s, average packet inter-arrival time of both Skype and MSN increases. However, the increasing amount of the inter-arrival time of Skype is smaller than that of MSN. In addition, an interesting and important observation is that no matter what the bottleneck bandwidth is, the variance of packet inter-arrival time in Skype is much smaller than that of MSN as shown in Fig. 2(b) and Fig. 3.

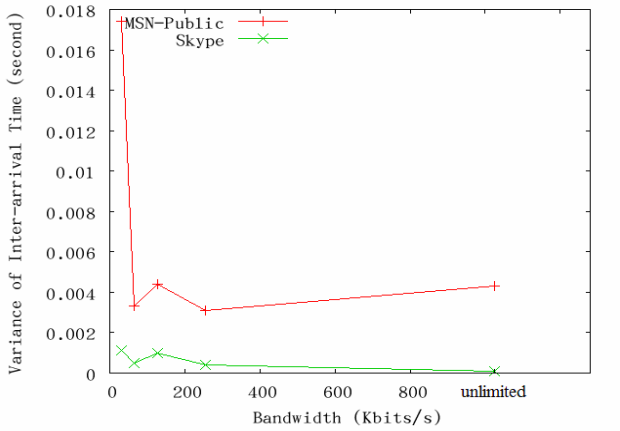
The above observations directly explain why the MOS of Skype is higher than that of the MSN - Skype keeps smooth transmission, which results in the smaller variance of the packet inter-arrival time. Especially, when the available bandwidth is as low as 32 Kbit/s, to maintain a smooth transmission, Skype not only reduces its sending rate but shrinks its packet size from 150 Bytes to 87 Bytes. On the other hand, MSN still keeps its sending rate and packet size. This also explains why the average and variance of packet inter-arrival time in MSN increase as the path capacity drops to 32 Kbit/s. Therefore, the performance improvement of the Skype in Fig. 2(c) is mostly contributed from the better rate control mechanism and voice codec. On the other hand, higher variance in MSN may result from the silence suppression mechanism, which we will investigate later.

2) Error Resilience Mechanism – The Influence of Packet Lost: In this experiment, we want to explore the resilient capability of the VoIP software under a loss-prone network. We use dummynet to introduce packet loss events in the voice connection with loss rate ranging from 0% to 40%. As shown in Fig. 4, no matter what the packet loss rate is, the average packet inter-arrival time of Skype is larger than that of MSN while the variance of the packet inter-arrival time of Skype is much smaller.

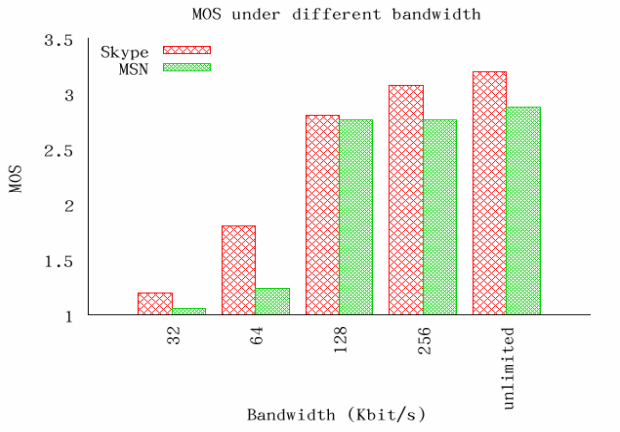
Moreover, as the packet loss rate increases, Fig. 5(a) shows that the throughput of the MSN voice connection decreases as we expect. On the contrary, the throughput of the Skype connection increases a lot as packet loss rate is higher than 10%. This is because Skype will send more packets and increase the packet size from one hundred bytes to two or three hundred bytes when loss events occur. Also, this can illustrate how the error-recovery mechanism works in the



(a) Inter-arrival time



(b) Variance of inter-arrival time



(C) MOS

Figure 2. (a) The average Inter-arrival time (b) The variance of inter-arrival time (c) The MOS information

presence of loss events. On the other hand, based on our collected data, we cannot find any error resilience scheme in the MSN application. This explains why MSN's throughput drops as the packet loss rate increases.

Next, we would like to investigate how good the error-recovery mechanism of Skype is. As shown in the Fig. 5(a)

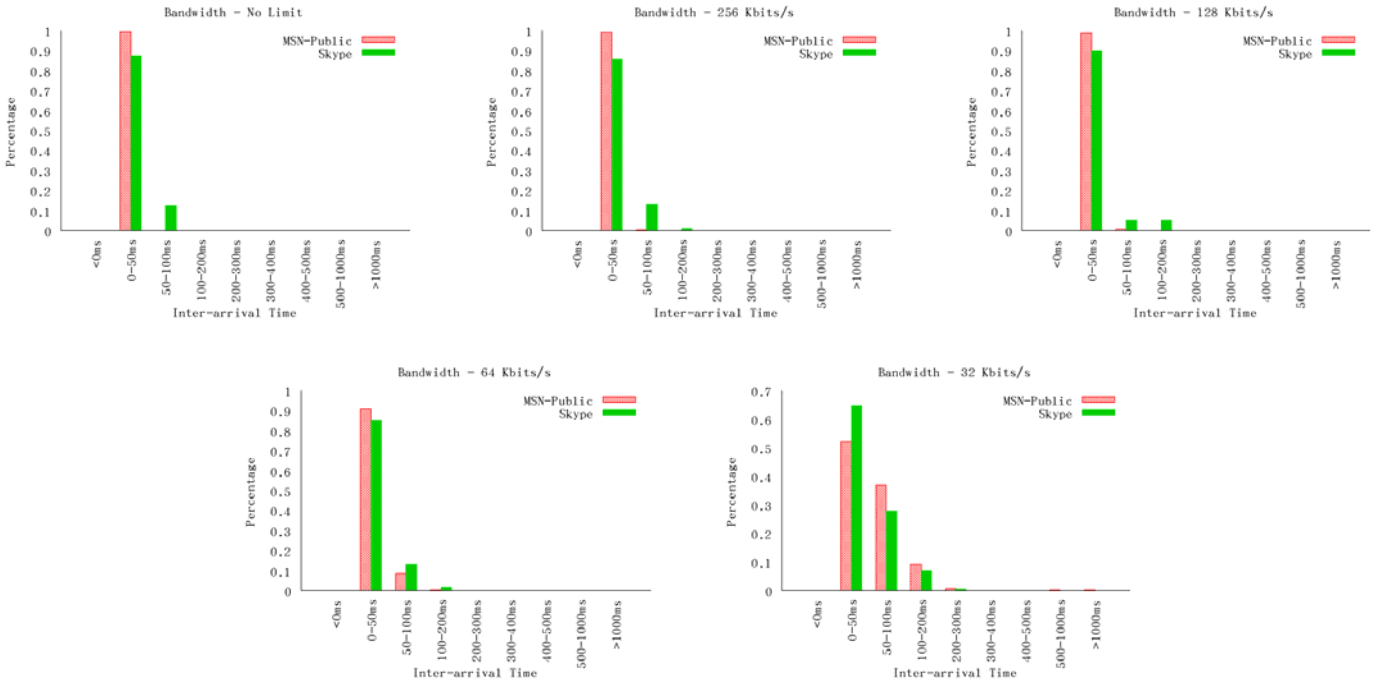


Figure 3. Relation between the inter-arrival time and bottleneck bandwidth

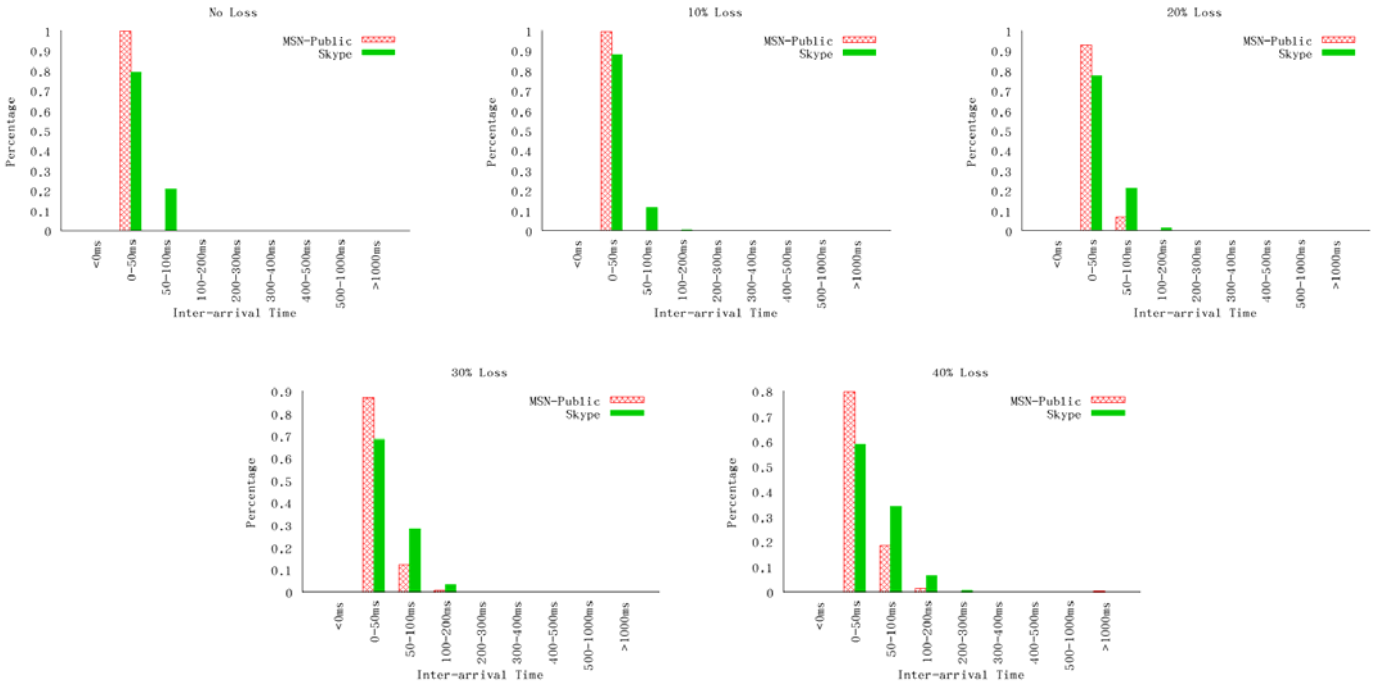
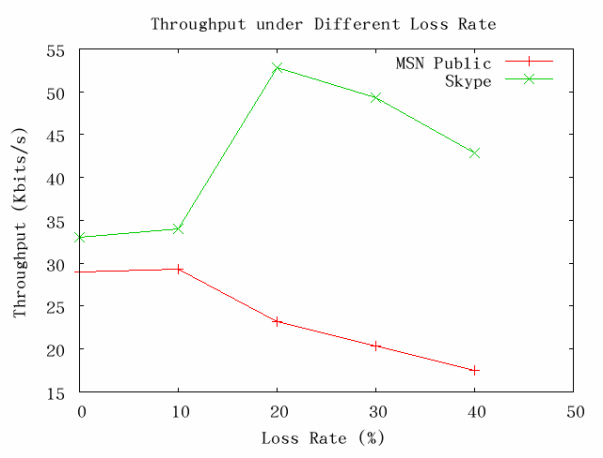


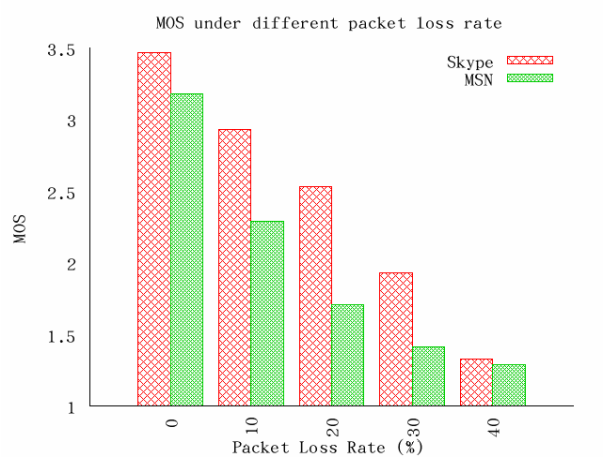
Figure 4. Relation between the inter-arrival time and loss rate

and 5(b), when the packet loss rate is less than 10%, the throughput of Skype increases slightly and the MOS still keeps higher than 3. When the packet loss rate becomes larger than 10%, the throughput of Skype increases a lot while the MOS still decreases. In other words, as the packet loss rate is larger than 10%, the effect of the error-resilience scheme of Skype is not significant and this error-resilience scheme consumes a lot of network resources as well. Therefore, the error-resilience mechanism of Skype works well as the packet loss rate is under 10%. On the other hand, as the packet loss rate is larger than 10%, this error-recovery mechanism might have to be re-designed.

3) Silence Supression: From the collected data, we can find that the silence suppression scheme is supported in the MSN voice service but not in Skype. However, sometimes this scheme might have negative effects on the voice quality.



(a) Throughput under different loss rate



(b) MOS

Figure 5. Experiment results under different packet loss rate: (a) throughput (b) MOS

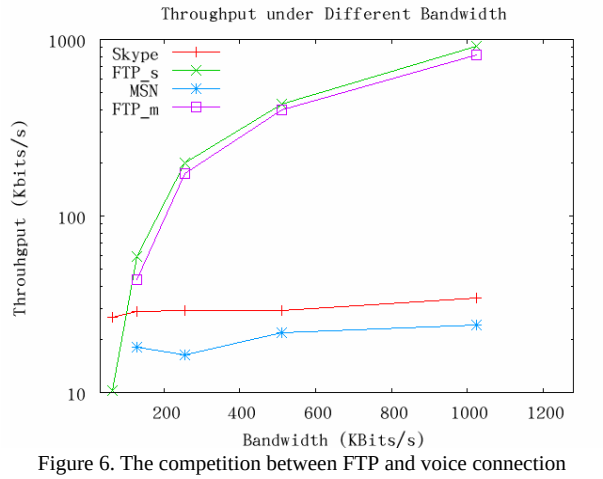


Figure 6. The competition between FTP and voice connection
FTP_s is the FTP traffic which competes with Skype.
FTP_m is the FTP traffic which competes with MSN.

For instance, Skype delivers packets regularly even if the user does not speak. This also explains why the Skype voice service has the low variance of the inter-arrival time of packets.

D. Direct Connection with Background Traffic: TCP-Friendly or not?

This experiment is to examine whether the two VoIP applications have any congestion control mechanism, i.e., we are interested in investigating TCP-friendliness of Skype and MSN. As shown in Fig. 6, as bottleneck bandwidth is larger than 128Kbits/s, the throughput of Skype is around 30Kbit/s and that of MSN is around 20Kbit/s. In this case, we note that the available bandwidth can both meet the demands of FTP and VoIP software. When we limit the bandwidth to 64Kbit/s, the MSN voice connection consumes almost all the bandwidth, which results that we cannot start another FTP connection. Similarly, the Skype connection uses most of the link resources and leaves only 1Kbit/s for the FTP. From the above observations, we can see that both Skype and MSN do not have implemented the congestion control mechanism, i.e., they are not TCP-friendly. The reason why the Skype leave 1Kbit/s for the FTP could be the smooth packet delivery of the Skype connection. Thus, the FTP packets are not dropped at all. On the contrary, the bursty traffic in the MSN connection does not give any chance to the FTP session to transmit a packet. This is why we cannot start another FTP connection simultaneously with the MSN voice connection and do not depict the point for the MSN connection when the bandwidth is 64Kbits/s.

E. Connection Through NAT: The Relay Node Mechanism

According to the research work in [1], we know that the Skype can go through port-restricted NAT directly with the help of relay nodes to start connections. When Skype goes across a UDP-restricted NAT, one of the super nodes will help relay voice packets via a TCP session. In MSN, when the client behind a firewall wants to set up a voice connection, the connection can be indirect, i.e., the voice packets will be relayed through a certain node in the US even if both clients are located in Taiwan. Therefore, the voice quality of the conversation will drop down due to the long route through the US. Table I includes the throughput ratio of Skype to MSN under different network conditions. In every experiment, the throughput of MSN is normalized to 1.0. And then we use this normalization to compute the throughput ratio of Skype to MSN under the same condition. For example, when the voice packets are transmitted through NAT and the link loss rate is 0%, throughput ratio of Skype to MSN is 1.47:1. However, if the voice connection is direct, this ratio is only 1.13:1. From the results in Table I, we can see that Skype always has higher throughput than MSN, and that when the voice packets go through NAT, MSN uses some relay node such that the achieved throughput is even lower than that of direct connection, which can lead to worse voice quality.

TABLE I. Throughput Ratio of Skype to MSN

Type of Connection	Loss Rate (%)				
	0	10	20	30	40
Direct	1.13	1.13	2.25	2.43	2.51
Through NAT	1.47	1.97	3.08	3.67	4.11

To sum up, under a NAT or a firewall network environment, the voice quality of the MSN connection becomes worse since the voice packets are re-routed through a

remote host. On the other hand, the Skype connection can set up its voice packets without the help of super node even under the NAT or the port-restricted firewall network environment. Finally, we note that such relay node mechanism, i.e., going through the NAT or the firewall, has a significant impact on the performance of the VoIP applications.

IV. Related Work

In recent years, VoIP service becomes more and more popular, and there have been many applications that allow users to send voice calls or instant messages to their friends on the Internet. Some well-known applications include Skype, MSN Messenger, Yahoo Messenger, Google Talk, etc. In early days, MSN and Yahoo Messenger mainly focus on delivering instant messages instead of voice call. People can save their friends' information in the buddy list and send messages to them when they are online. However, like Skype, MSN Messenger and Yahoo Messenger are also supporting voice calls now. Moreover, Skype also provides service of calling ordinary phone numbers around the world, which is named "SkypeOut", and service of accepting call from a real phone number, which is named "SkypeIn."

There have been some researches [1][2] discussing the key components of Skype. As in [1], they include a) super node: there are some nodes called the "super node" that help relay voice packets and request when end hosts are behind restricted networks, b) host cache: host cache is a list of super nodes cached in the Windows Registry. It helps a Skype client build connections with others, c) codec: Skype uses some wideband codec to encode voices, and d) NAT and firewall: when clients reside behind NAT or firewall, Skype will try to go through the firewall with a variation of the STUN and TURN protocols.

In this paper, however, we focus on the performance and quality that Skype can reach, compare it to another popular application – MSN Messenger under various network conditions, and then discuss about the metrics that affect the performance of VoIP applications.

V. Conclusion

We compare the most popular VoIP applications, Skype and MSN, to observe how they react to different network conditions. At the beginning, we observe the original behaviors of Skype and MSN in normal network environments. Then, we test the influence of insufficient bandwidth, packet loss events, and the competition with other traffic. We also try to compare the difference between setting the clients behind NAT and exposing them to the Internet.

- In general, we know that MSN sends smaller packets with higher sending rate than Skype. Owing to the effect of silence suppression, MSN has higher variance in packet's inter-arrival time while Skype sends packets in a stable fashion. The packet size of Skype varies with the network condition while that in MSN is almost fixed.
- In Skype, the bandwidth insufficiency has little influence on the inter-arrival time; that is, the variance of inter-arrival time in Skype is very low. Skype also has better performance than MSN in MOS score.
- Skype increases its throughput to recover missed data when suffering from higher loss rate. On the other hand, the throughput of MSN drops when facing the same situation.
- Skype has better mechanisms to go through NAT. This avoids packets traveling long route through relay nodes and decreases the influence of background traffic. MSN does not address this issue much.

From the differences of Skype and MSN, it is obvious that Skype is trying to improve the overall quality via some mechanisms. We believe that these analytical results are helpful if they can be applied to further design in VoIP applications, and then these VoIP applications can have voice connections as stable as PSTN.

References

[1] S. A. Baset and H. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol", Columbia University Technical Report CUCS-039-04, Sept. 2004.

[2] D. Bergstrom, "An Analysis of the Skype VoIP application for use in a corporate environment", Oct. 2004.

[3] ITU-T Recommendation P.800, "Methods for Subjective Determination of Transmission Quality", Aug. 1996.

[4] H Zlatokrilov and H Levy, "Packet Dispersion and the Quality of Voice over IP Applications in IP networks", IEEE INFOCOM, 2004

[5] L. Sun and E. Ifeachor, "New models for perceived voice quality prediction and their applications in playout buffer optimization for VoIP networks," in Proc. ICC, June 2004.

[6] R. Cole and J. Rosenbluth, "Voice over IP performance monitoring," Journal on Computer Commun. Review, vol.31, Apr. 2001.

[7] A. Kansal and A. Karandikar, "Adaptive delay estimation for low jitter audio over Internet," IEEE GLOBECOM, vol.4, pp.2591-2595, 2001.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

BMC: A Two-stage Switch Architecture for High Performance Multimedia Communication

Yang Xu[1], Bin Liu[2], Beibei Wu[3], Wei Li[4]
Department of Computer Science and Technology
Tsinghua University
Beijing 100084, P. R. China

{xy01[1], wbb02[3], li-wei03[4]}@mails.tsinghua.edu.cn, liub@tsinghua.edu.cn[2]

***Abstract*—Output Queueing (OQ) emulated switch is desirable in real-time applications because it can guarantee the lowest average cell delay. The challenging issue in OQ emulated two-stage switch is the extreme complexity of the scheduling operations between the first stage switch and the central shared memories. To overcome this problem, a new switching architecture called Banyan-Memory-Crossbar (BMC) is proposed in this paper. In BMC, the first stage switch is a banyan-like interconnection network, which is used to route the incoming cells from input ports to central memories without occurring arrival and departure conflicts. Several different implementations of banyan network are given in this paper, which are named as RANDBN, F^{2}BN, and SF^{2}BN respectively. Compared with previous methods in OQ emulated two-stage switch, they don't need centralized scheduler, causing the communication and computation overhead reduced dramatically. Theoretical analysis and simulations show that the cell loss rates in RANDBN and SF^{2}BN are 21.31%, while the cell loss rate of F^{2}BN is less than 1%. Delay guarantee capability and such a slim cell loss rate make F^{2}BN practical in real-time applications.**

Index Terms*-- Multimedia Communication, Two stage switches, High-speed, Banyan, Scheduling

I. INTRODUCTION

Switch/Router is the infrastructural equipments of multimedia communication, where cell delay and throughput/drop ratio are two important performance measurements. (For real-time applications, cell delay is more important).

Output Queueing (OQ)[1] switch is known as the ideal switch model because it can guarantee the maximum throughput and the lowest average cell delay (the delay of each cell can also be predicted and easily controlled[2][3]), thus is desirable for high performance multimedia communication. However OQ is extremely hard to implement under high line speed or large port number environments, the reason lies in both the switch fabric and the output queues need to run at N times the line rate, where N is the number of ports.

Therefore several more scalable architectures, which less speedup, were proposed to emulate the behavior of OQ. One of such architectures is so called Combined Input and Output Queueing (CIOQ) [4][5][6]. It has been proven that a CIOQ switch with a speedup of 4 or 2 can exactly emulate an OQ switch by employing specially designed scheduling algorithms, such as MUCFA, CCF, and JPM, but none of these presented algorithms is practical in real routers/switches for their high time complexity.

Besides CIOQ switches, the Switch-Memory-Switch (SMS) architecture provides another way towards emulating OQ[7][8][9][10][11]. In SMS switches, two crossbar fabrics are used in the first and second stage switching respectively, and between them located some shared memories. The main problem in SMS switch is the scheduling in the first stage, where two kinds of conflicts must be avoided: arrival conflict and departure conflict[9][11]. RiPSS, a request-grant-accept based scheduling algorithm, was proposed together with its pipeline version PRiPSS in [8] and [9]. RiPSS needs $(2N+\varepsilon)$ independent memories (where ε is a very small positive number), and can complete the matching with $O(\log^* N)$ rounds of iterations w.h.p in N. Despite the good performance, RiPSS/PRiPSS requires a bipartite graph without departure conflict set up in each time slot, which has the time complexity of $O(N)$ [11]. So RiPSS/PRiPSS is unpractical.

The objective of this paper is to design a new switch architecture optimized for real-time applications, which should have the following characteristics:

- ✓ Guaranteed cell delay performance with acceptable cell loss rate. For multimedia applications, especially real-time applications (such as VoIP and video conferencing), cell delay is the most crucial measurements, while slim cell loss is acceptable.
- ✓ The scheduling in the architecture should be simple enough to be implemented in very high-speed environment.

The rest of this paper is organized as follows. In section II, we introduce a new two-stage switch architecture named Banyan-Memory-Crossbar (BMC), which uses a banyan network to route the incoming cells into shared memories without occurring arrival and departure conflicts, so that emulates a FIFO-based OQ architecture. Section III presents several such banyan networks, whose performance evaluations and simulation results are given in section IV. Finally section V gives the conclusion.

Supported by the NSFC under Grant No. 60373007 and No. 60573121;China-Ireland Science and Technology Collaboration Research Fund (CI-2003-02) and Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20040003048);985 Fund of Tsinghua University(No. JCpy2005054)

II. The Architecture of Banyan-Memory-Crossbar Switches

In this paper, we assume that only fixed sized packets, also called cells, are transferred inside the switch.Time is divided into time slots and one time slot is equal to the transmission time of a cell.

A. How to Emulate an OQ Switch with FIFO Policy

Our goal is to design an OQ emulated switch. Here, we give the definition of OQ emulated switches firstly.

Definition 1: OQ emulated switches (in narrow sense)[5]:

A switch is said to be an OQ emulated switch, if under identical inputs, the departure time of every cell from this switch is exactly identical to that from an OQ switch (The detailed process to compute each cell's departure time can be referred to [8].). If a cell is dropped in an OQ switch, it will be dropped in this OQ emulated switch too.

Definition 2: OQ emulated switches (in broad sense):

A switch is said to be a broad-sense OQ emulated switch, if under identical inputs, the departure time of every cell from this switch adds a fixed delay to that from an OQ switch. If a cell is dropped in an OQ switch, it will be dropped in this OQ emulated switch too.

In the rest of this paper, OQ emulated switches refer to OQ emulated switches in broad sense, if there is no other special notation.

B. Banyan-Memory-Crossbar Architecture

Here, we present a novel switch architecture named Banyan-Memory-Crossbar (BMC) to emulate the OQ switch, which is composed of three parts: distributing banyan network, shared memories, and crossbar fabric. Fig. 1. depicts the architecture of a BMC switch with 8 ports, which is composed of three parts:

1) Distributing banyan network
2) Shared memories
3) Crossbar

Distributing banyan network is a $2N \times 2N$ banyan network, which connects N input ports and $2N$ shared memories (N is assumed to be a power of 2). It consists of $\log_2 N+1$ phases. Each phase contains N 2×2 switch elements (SE) with two states: either cross or bar. The SE located at row j and column (phase) j is marked as $SE_{i,j}$, $(0 \le i \le N-1, 0 \le j \le \log_2 N)$.

Cells are assumed to be transferred from current phase SE to the next phase SE every time slot. So it needs $\log_2 N+1$ time slots for a cell to be transferred from the input ports to the shared memories crossing the banyan network.

Assume there exists a shadow OQ switch with FIFO policy, where the length of each queue is L. To use BMC to emulate the shadow OQ switch, arrival cells must be put into memories satisfying the following two requirements:

(1) Cells arriving simultaneously should be put into different memories.

(2) Cells with the same departure time should also be put into different memories.

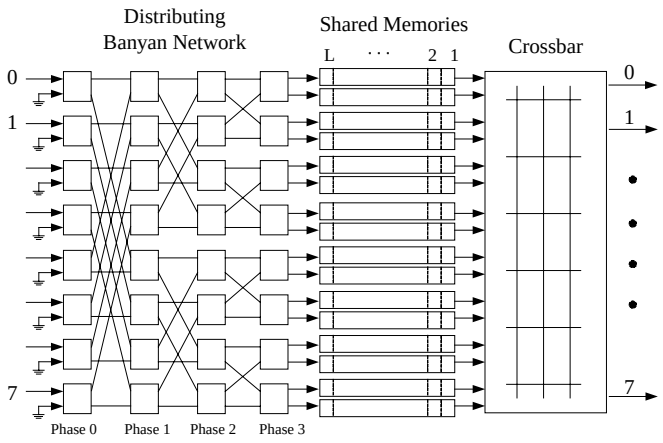


Fig. 1. The architecture of Banyan-Memory-Crossbar switch (an example with N=8).

When the first requirement is satisfied, we say arrival conflicts are avoided, and when the second is satisfied, departure conflicts avoided. For there are at most N cells arriving simultaneously, and at most N cells having the same departure time, it can be easy proven that when $M \ge 2N-1$ (M is the number of shared memories), it is enough to put a new arrival cell into an available memory without conflict[7][10]. In this way, the operations of such a two-stage switch in each time slot can be divided into two parallel processes/jobs:

(1) In the first stage switching, new arrival cells are scheduled into shared memories without arrival and departure conflict.

(2) In the second stage, cells whose departure time is equal to the current time slot are read out from the memories and sent to the corresponding output ports. This job is easy to be done for all these cells are staying in different memories, and destined to different outputs.

So the challenge in BMC mainly lies in the first stage where cells must be scheduled without arrival and departure conflict. In the left of this paper we focus on the scheduling in the first stage switch.

C. Why employing banyan network

The banyan type of interconnection network was originally defined in [13]. It has the property of exactly one path from any input to any output, which can be used to avoid arrival conflict.

Here we make a comparison between traditional conflict-avoiding solutions (such as RiPSS/PRiPSS) and our banyan-based method. In traditional solutions, each memory first finds out which inputs are departure-time compatible to it, and then a bipartite graph without departure conflict is constructed. Then the scheduling algorithm is executed to find a matching without arrival conflict based on that bipartite graph. In these methods, the time complexity to construct a bipartite graph without departure conflict is very expensive ($O(N)$ for each memory), and we call them as *departure-conflict-avoiding-first* method.

Our method, however, utilizes the characteristic of banyan networks that there is exactly one path from any input to any output to avoid the arrival conflict firstly. The remainder work for the banyan network is to find an appropriate method to avoid the departure conflict. We call it *arrival-conflict-*

avoiding-first method. The strongpoint of *arrival-conflict-avoiding-first* methods is that it avoids the computing overhead to construct a bipartite graph without departure conflict, so that it is more practical in high-speed environment.

III. DISTRIBUTING BANYAN NETWORK

In this section, we present several kinds of Banyan network, which are used in BMC switches to avoid the departure conflict.

When a cell arrives at an input port, its departure time is first computed based on the shadow OQ switch[9], and a label will be tagged to its header before it enters the BMC switch. The format of a tagged cell is shown in Fig. 2, where *DP* is the destination port of the cell, with the length of $\log_2 N$, and *DT* presents the departure time of this cell in the shadow OQ switch, with the value denoted as follows.

$$DT=\text{DepartureTime(cell)} \bmod L. \quad (1)$$

Fig. 2. Format of Cells entering the BMC switch.

A. Random Banyan Network

We first use a Random Banyan Network (RANDBN) to solve the departure conflict. Denote the probability of state 'cross' in $SE_{i,j}$ at time slot t as $Pr(SE_{i,j}=Cross, t)$, and state 'bar' as $Pr(SE_{i,j}=Bar, t)$. Then in RANDBN, we let

$$Pr(SE_{i,j}=Cross, t)= Pr(SE_{i,j}=Bar, t)=\frac{1}{2},$$

Where $0 \le i \le N-1,\ 0 \le j \le \log_2 N,\ t \in Z^+ \bigcup \{0\}$.

When a cell arrives at a SE, it will be forwarded away according to the state of SE in a random manner. After traversing all the phases of RANDBN and reaching one certain shared memory, the cell will be accepted if no cell with the same *DT* is in this memory. Otherwise, it will be dropped, and we call this a *departure conflict*.

B. Flip-Flop Banyan Network

It will be later seen in section IV that the cell conflict probability in RANDBN is quite high. To decrease the conflicts, we ask the banyan network remember the path each cell traversed, so that when a new cell arrives the banyan network can try its best to prevent it from reaching a memory that has already been occupied by another cell with the same departure time.

Here, we give some useful definitions firstly, and then introduce a novel Flip-Flop Banyan Network (F^2BN).

Earliest Departure Time ($EDT_i(t)$): in time slot t, to SEs in the i-th phase, the earliest departure time of all possible arrival cells.

Latest Departure Time ($LDT_i(t)$): in time slot t, to SEs in the i-th phase, the latest departure time of all possible arrival cells.

For example in time slot t, the departure times of cells at $SE_{*,0}$ range from $(t+1) \bmod L$ to $(t+L) \bmod L$ (Cells whose departure time is larger than $t+L$ will be dropped in the shadow OQ switch). So $EDT_0(t)= (t+1) \bmod L$, and $LDT_0(t)= (t+L) \bmod L$.

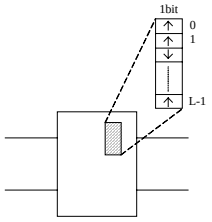


Fig. 3. Forwarding information table in each SE.

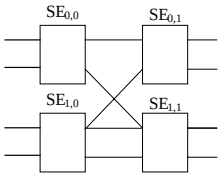


Fig. 4. A 4×4 F^2BN.

More generally, for SEs in phase i,

$$EDT_i(t)= (t+1-i) \bmod L, \quad (2)$$

$$LDT_i(t)= (t+L-i) \bmod L. \quad (3)$$

1) The Structure and Behavior of Each SE

To record the path information, in F^2BN, we give each SE an independent Forwarding Information Table (FIT), and denote FIT at $SE_{i,j}$ as $FIT_{i,j}$. FIT is used to remember the outlet of cells with different *DT*s (between *EDT* and *LDT*), and the sketch map of FIT is shown in Fig. 3.

Each FIT has L entries, and each entry only has 1 bit. We assume '0' presents 'up', and '1' 'down'. In the beginning, all entries in FITs have an initial value, e.g. 'up' or 'down'. When a cell with $DT = t_k$ arrives at $SE_{i,j}$, $SE_{i,j}$ will check the t_k-th entry in $FIT_{i,j}$. If the value of this entry is '0', the cell will be forwarded to the upper line of the SE, otherwise it will be forwarded to the lower line of the SE. After that, the value of the t_k-th entry in $FIT_{i,j}$ will be reversed.

F^2BN has a very useful property:

Theorem 1. Assume N same-departure-time cells arrive at a $2N\times2N$ F^2BN network slot by slot. If there is no other cell arriving, these N cells will be routed definitely to different outlets.

Proof: Let the number of cells traversing $SE_{i,j}$ be $X_{i,j}$.

For simplicity, we only give the proof of N=2 (as shown in Fig. 4).

Because each entry in FIT works as the round-robin manner, so long as $X_{0,1} \le 2$ and $X_{1,1} \le 2$, each outlet will receive no more than one cell.

Similarly, we can get

$$\begin{aligned} X_{0,1} &\le \left\lceil \frac{X_{0,0}}{2} \right\rceil + \left\lceil \frac{X_{1,0}}{2} \right\rceil \\ &\le \frac{X_{0,0}+X_{1,0}}{2}+1 \end{aligned} \quad (4)$$

Since there are at most two cells with the same departure time, $X_{0,0}+X_{1,0} \le 2$. Substitute it into (4), we get

$$X_{0,1} \le 2 \quad (5)$$

In the same way,

$$X_{1,1} \le 2 \quad (6)$$

Combining (5) and (6), it can be concluded that each outlet will receive no more than one cell.

□

Theorem 1 shows the fact that all cells can be routed to different memories if there is no collision along their paths. But actually cells with different departure times coexist in the distributing banyan network, the collision is inevitable when cells travel through the distributing banyan network. Hence two points must be stressed:

(1) In each time slot, there are at most two cells arriving at a certain SE simultaneously. If these two cells are all scheduled to the same outlet, collision will occur. So one of them must be dropped or shifted to the unwilling outlet, in this case we call it a *bump*. In F^2BN no cell will be dropped in the SE, and when a *bump* happens, the cell with low priority will be shifted to the idle outlet, and the corresponding entry in the FIT will not be changed.

(2) Because there are only L entries in each FIT, these entries must be reused every L time slots. In F^2BN, entries in a FIT are all set to the same initial value ('up' or 'down'). At the end of time slot t, the entry whose index is $EDT_j(t)$ in $FIT_{i,j}$ should be reinitialized.

Now, the behavior of each SE in each time slot can be summarized as follows:

(1) Receiving new arrival cells.

(2) Checking FIT, forwarding cells, and updating FIT.

(3) Reinitializing FIT.

The details of step (2) and (3) are described by pseudocode in Fig. 5. It can be seen that only one checking operation, two update operations and one reinitializing operation are needed in each time slot at each SE.

2) The Initial Value and Priority Assignment in F^2BN

In F^2BN, there must be an initial value assignment for each entry of FITs to perform reset, and a priority assignment for each SE to perform the *bump* when two cells arrive at a SE simultaneously.

We divide banyan network into interconnect-groups (ICG). Each group is composed of four SEs, which are interconnected by four wires. As Fig. 6 shows, $SE_{0,0}$, $SE_{0,1}$, $SE_{2,0}$, and $SE_{2,1}$ compose a single ICG. Here, we use the position of the top left SE to mark an ICG. For example, the ICG in Fig. 6 is marked as $ICG_{0,0}$.

In each ICG two left SEs share the inlets of two right SEs. In order to balance the traffic load between the two right SEs, it's better to assign different initial values to the two left SEs. The initial value assignment is explained with pseudocode in Fig. 7, and an example of initial value assignment is shown in Fig. 6.

In the SE, when a bump happens, one of the two cells must be shifted to another outlet according to the cells' priorities. There are several kinds of methods to assign priorities, such as: high priority to upper inlet, high priority to lower inlet, or high priority to random inlet. The first two methods will cause unfairness, and the random method is difficult to implement in hardware. So we assign the priorities still based on ICG, just like the method used in initial value assignment. The priority assignment is described with pseudocode in Fig. 8, and an example is shown in Fig. 6, where the upper inlet of $SE_{0,1}$ and the lower inlet of $SE_{2,1}$ have high priority, the other two inlets of $SE_{0,1}$ and $SE_{2,1}$ have low priority.

3) Simple Flip-Flop Banyan Network

The RANDBN is costly to implement in hardware for the use of random, we can just use a simple version of Flip-Flop Banyan Network (SF^2BN) to mimic a RANDBN. SF^2BN has almost no difference with F^2BN, except that there is only one entry in each FIT, which is shared by cells with different departure times. In SF^2BN, there is no need to reinitialize FIT.

Checking FIT, forwarding cell, and updating FIT
At time t, for all $SE_{i,j}$ in parallel
  Upon cells arrival
  Case: one cell c_1 arrives with $DT=t_1$
    Check the t_1-th entry in FIT, and return $direct_1$
    Forward c_1 to $direct_1$
    Reverse the t_1-th entry in FIT
  Case: two cell c_1, c_2 arrive with $DT=t_1$, t_2 respective, and whose priorities are pri_1, pri_2
    If $pri_1>pri_2$
      Check t_1-th entry in FIT, and return $direct_1$
      Forward c_1 to $direct_1$
      Reverse the t_1-th entry in FIT
      Forward c_2 to another direction
      Update the t_2-th entry in FIT to $direct_1$
    Else
      Check t_2-th entry in FIT, and return $direct_2$
      Forward c_2 to $direct_2$
      Reverse the t_2-th entry in FIT
      Forward c_1 to another direction
      Update the t_1-th entry in FIT to $direct_2$

(a) Checking FIT, forwarding cell, and updating FIT.

Reinitializing FIT
At time t, for all $SE_{i,j}$ in parallel
  Set the $EDT_j(t)$-th entry in $FIT_{i,j}$ to initial value

(b) Reinitializing FIT

Fig. 5. Pseudocode for the behavior of each SE in F^2BN.

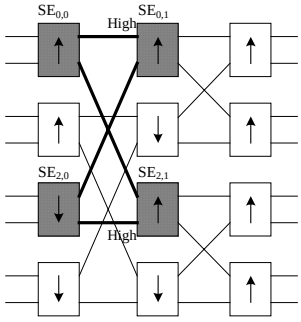


Fig. 6. An interconnect-group in an 8×8 banyan network.

Initial Value Assignment
From phase 0 to phase $\log_2 N-1$
For all ICGs in this phase
  Let the initial value of the top left SE be 'up'
  Let the initial value of the bottom left SE be 'down'
For all SEs in phase $\log_2 N$
  Let the initial value of the SE be 'up'

Fig. 7. Pseudocode for initial value assignment.

```
Priority Assignment
From phase 0 to phase  log2 N − 1
    For all ICGs in this phase
        To the top right SE
            Give high priority to the upper inlet
            Give low priority to the lower inlet
        To the bottom right SE
            Give high priority to the lower inlet
            Give low priority to the upper inlet
For all SEs in phase 0
    Give high priority to the upper inlet
    The lower inlet is left idle
```

Fig. 8. Pseudocode for priority assignment.

IV. Performance Evaluation

Here we analyze the cell loss rate in RANDBN. It is difficult to give the analytic solution of cell loss rate of F^{2}BN currently, so we evaluate its performance through simulations.

A. Theoretical Analyses

Since the state of each SE in the RANDBN is selected randomly, the final destination of a certain cell can be regarded to be distributed uniformly among all of the shared memories. Let R be the probability of a cell reaching one certain memory M_i. For there are $2N$ memories, $R = \frac{1}{2N}$.

Consider there will be at most N cells with the same departure time, we present the ball-bin model, in which each of the N cells is placed in any of the $2N$ independent memories with equal probability of R.

Let $\Pr(M_i = 0)$ be the probability that M_i is empty, and $\Pr(M_i = 1)$ be the probability that M_i is nonempty. Then the average number of nonempty memories is:

$$\text{E(nonempty memories)}=\text{E}(\sum_{i=0}^{2N-1}\Pr(M_i=1))=2N\cdot\Pr(M_i=1).$$

Since $\Pr(M_i = 0) = (1-\frac{1}{2N})^N$ and

$\Pr(M_i = 1) = 1 - \Pr(M_i = 0)$,

$$\text{E(nonempty memories)}= 2N\cdot[1-(1-\frac{1}{2N})^N]. \quad (7)$$

The average number of conflicts is:

$$\text{E(conflicts)}= N - \text{E(nonempty memories)}$$
$$= N - 2N\cdot[1-(1-\frac{1}{2N})^N]. \quad (8)$$

Let c be the probability of conflict, then

$$c = \frac{\text{E(conflicts)}}{N} = 1-2\cdot(1-(1-\frac{1}{2N})^N). \quad (9)$$

When $N \to \infty$, $c = 2e^{-\frac{1}{2}} - 1 \approx 21.31\%$.

B. Experiments

In this subsection, the performance of the banyan network is evaluated in a series of simulations. Since BMC is to emulate OQ switches, we focus on the metric of cell loss rate. There are two factors that might cause a cell loss in BMC. Firstly, bump may occur when two cells select paths on a same SE, and one cell may not be able to select its preferred path. So when a cell reaches the memory, another cell with the same DT may already exist. In this situation, the new arrival cell will be dropped. Secondly, the queue length of the shadow OQ switch is finite, so there's unavoidable congestion caused by multi-cells simultaneously arriving at different inputs addressing to the same output, then drop appears. Much work has been done on the second situation to address the cell loss problem in OQ[1]. Here we just study the cell loss caused by the first factor.

In simulations, two typical traffic patterns are used: Bernoulli i.i.d. uniform traffic and bursty traffic. The simulation progress lasts 1000000 time slots when $N \le 256$, and 200000 time slots when $N \ge 512$.

1) Performance of RANDBN and SF^{2}BN

We first make the study of RANDBN and SF^{2}BN. Denoting ρ as the offered load, we observe the relationship between cell loss rate c and ρ under Bernoulli i.i.d. uniform traffic, with N to be 32 and 1024. In RANDBN, cell loss rate increases linearly as ρ increases, which can be seen from the upper two lines in Fig. 9. The lower two lines depict the case in SF^{2}BN, where cell loss rate increases linearly with ρ above 0.5, and when ρ is below 0.5, the increase becomes much slower. Generally speaking, SF^{2}BN is always better than RANDBN, because using SF^{2}BN the cell distributing can be recorded to a certain extent, and through changing the state of SE, load balancing can be achieved. Fig. 10 shows the relationship between cell loss rate and N in SF^{2}BN and RANDBN when ρ is 1.0. The performances of these two methods are quite close. We can see the simulation results match the theoretic conclusion very well. When ρ is 1.0 and N approaches infinite, both of the two methods can reach the throughput of 78.69%, and the probability of cell conflict is 21.31%.

2) Performance of F^{2}BN

a) Bernoulli uniform i.i.d. Traffic

As depicted in Fig. 11, cell loss rate in F^{2}BN is plotted as a function of ρ with various values of N. When ρ is 1.0, the cell loss rate is between 10^{-2} and 10^{-3}. When ρ decreases, loss rate drops very quickly. With ρ below 0.5, there is no cell loss during the simulation.

On the other hand, with the same value of ρ, cell loss rate decreases as N increases. The reason for such phenomenon is that, anytime with a higher N, the ratio between compatible and unavailable memory number is higher, so that the probability for cell to reach compatible memory is larger.

In F^{2}BN, the cell loss rate is always lower than 1%. This strongly proves that the memorial capability of SE has an effective influence on load balancing cells with the same departure time.

b) Bursty Traffic

We use the same bursty traffic model as in [12], and use two parameters to describe the traffic character: the mean burst length b and the offered load ρ. The probabilities that an active or an idle period will end at a time slot are fixed, which are denoted as p and q respectively. We can express the mean burst length and the offered load as follows.

$$b = \sum_{i=1}^{\infty} ip(1-p)^{i-1} = \frac{1}{p}, \tag{10}$$

$$\rho = \frac{\frac{1}{p}}{\frac{1}{p} + \sum_{i=0}^{\infty} iq(1-q)^{i}} = \frac{q}{q+p-pq}. \tag{11}$$

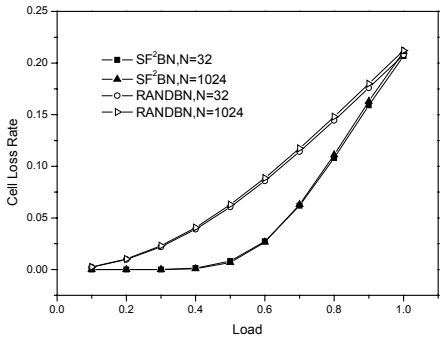


Fig. 9. Cell loss rate in SF^{2}BN and RANDBN under Bernoulli i.i.d. uniform traffic and various values of ρ, N=32, 1024.

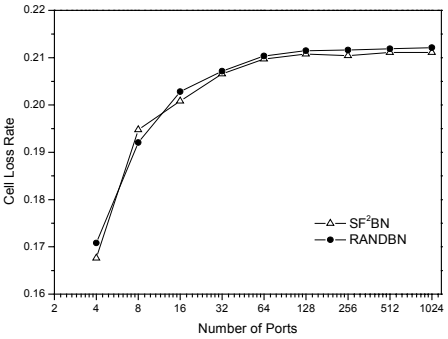


Fig. 10. Cell loss rate in SF^{2}BN and RANDBN under Bernoulli i.i.d. uniform traffic where ρ=1.0.

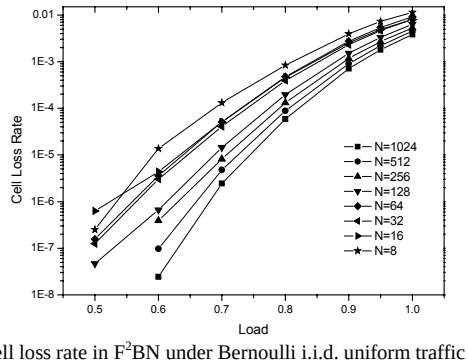


Fig. 11. Cell loss rate in F^{2}BN under Bernoulli i.i.d. uniform traffic and various values of ρ.

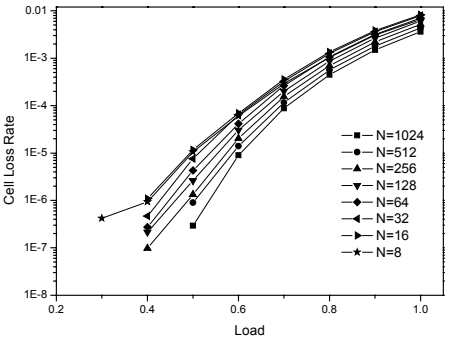


Fig. 12. Cell loss rate in F^{2}BN under the bursty traffic (b=16).

With b to be 16, Fig. 12 depicts the relationship between cell loss rate and ρ when N varies. The loss rate is appreciably greater than that under Bernoulli traffic, and the trends are similar. When ρ is 1.0, cell loss rate is still below 1%.

V. Conclusions

In this paper, we present a new kind of structures using banyan network to fulfill the non-conflicted cell distributing in OQ emulated two-stage switches. Three distribute banyan networks, named RANDBN, SF^{2}BN, and F^{2}BN, are proposed respectively. By recording the outlets of cells with different departure times in SE, load balance of the cells with the same departure time among memories can be perfectly achieved by F^{2}BN. The hardware implementation is quite simple, with just one checking operation for each cell to select a path at each SE.

The performance of F^{2}BN is well enough to achieve 99% throughput under both Bernoulli i.i.d. uniform and bursty traffic. Because BMC is an OQ emulated switch architecture, it can predict and guarantee the cell delay (which equals the delay in OQ switch adding a constant of $\log_2 N$). These excellent features make BMC a quite ideal solution for multimedia communication applications.

Currently, the cell loss rate of F^{2}BN is obtained through simulations. Further work will be done to get an analytic solution of the cell loss rate of F^{2}BN.

References

[1] M. Karol, M. Hluchyj, and S. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch," *IEEE Transactions on Communications*, vol. 35, pp. 1347-1356, 1987.

[2] Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," ACM SIGCOMM Computer Communication Review, vol. 19, pp. 1-12, 1989.

[3] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Transactions on Networking,* vol. 1, pp. 344-357, 1993.

[4] B. Prabhakar and N. McKeown, "On the speedup required for combined *input* and output queued switching," *Automatica*, vol. 35, pp. 1909–1920, Dec. 1999.

[5] S.-T. *Chuang*, A. Goel, N. McKeown, and B. Prabhakar, "Matching output queueing with a combined input/output-queued switch," *IEEE Journal on Selected Areas in Communications,* vol. 17, pp. 1030-1039, 1999.

[6] I. Stoica and H. Zhang, "Exact emulation of an output queueing switch by a *combined* input output queueing switch," in *Proc. IWQoS*, 1998, pp. 218-224.

[7] A. *Prakash*, S. Sharif, and A. Aziz, "An O($\log^2 N$) parallel algorithm for output queuing," in *Proc. IEEE INFOCOM*, 2002, pp. 1623-1629 vol.3.

[8] A. Aziz, A. Prakash, and V. Ramachandran, "A near optimal scheduler for switch-memory-switch routers," in *Proc. Fifteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, 2003, pp. 343-352.

[9] A. *Prakash*, A. Aziz, and V. Ramachandran, "Randomized parallel schedulers for switch-memory-switch routers: Analysis and numerical studies," in *Proc. IEEE INFOCOM 2004*, pp. 2026-2037.

[10] S. Iyer, R. *Zhang*, and N. McKeown, "Routers with a single stage of buffering," *Computer Communication Review,* vol. 32, pp. 251-264, Oct 2002.

[11] Y. Xu, B. Wu, W. Li, B. Liu, "A scalable scheduling algorithm to avoid conflicts in switch-memory-switch routers," *IEEE ICCCN 2005*.

[12] H. J. Chao, "Saturn: A terabit packet switch using dual round-robin," *IEEE Communications Magazine*, vol. 8, No. 12, pp. 78-84, Dec. 2000.

[13] L. R. Goke and G, J. Lipovski, "Banyan networks for partitioning multiprocessor *systems*," *Proc. 1st Annu. Int. Symp. Comput. Architecture*, pp.21-28, Dec. 1973.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

Rate Adaptation for Buffer Underflow Avoidance in Multimedia Signal Streaming

Matteo Petracca, Fabio De Vito, Juan Carlos De Martin
Dipartimento di Automatica e Informatica
Politecnico di Torino
C.so Duca degli Abruzzi 24, Torino, 10129, Italy
email: {matteo.petracca,fabio.devito,demartin}@polito.it

***Abstract*— In media transmission over packet networks, one of the most challenging issues is the avoidance of receiver buffer underflows. Among the approaches proposed to solve this problem, the source rate adaptation is promising, due to the availability of multi-rate encoders.**

In order to succeed, rate-adaptive approaches should take into account not only the network throughput, but also the playout buffer fullness. Using this information, it is possible to determine at which rate the source should encode the stream, in order to avoid underflows.

In this paper, we derive and analyze the expression of the underflow probability; we show that it can be written in closed form as a function of the source rate, the receiver buffer fullness and the channel statistics. In particular, we study the dependency on the media rate, and we show how to achieve infinitesimal underflow probabilities even in presence of a high channel variance. Furthermore, we specialize this formulation for a CBR channel.

In case of channel variations during the playout, the buffered data may not suffice to ensure zero underflow probability; based on the previous formulation, we present an algorithm to recompute the source rate that allows continuous media playout. This algorithm can be safely iterated at every channel throughput change, and proved to be effective in avoiding buffer underflows.

I. Introduction

In recent years, the interest of consumers in services over packet networks experienced a fast growth, mainly due to the wide diffusion of high-capacity wireless and wired access technologies. Along with traditional data services like ftp and web browsing, the increased network throughput, together with the development of highly efficient media coding techniques, made transmission of multimedia contents possible across today's Internet. Applications involving streaming of media resources impose several new challenges, mainly related to the bursty nature of the network. In particular, the transmission of stored video, audio, speech and synchronized text imposes strict bounds on the delivery time and jitter, and at the same time requires low loss rates, to avoid quality degradation in the decoded signal.

To compensate for the variability of network throughput, a certain amount of data needs to be stored within the receiver before starting the playout; this buffer will supply data and avoid media interruptions if the network is unable to deliver packets for a limited interval of time. The time spent for this pre-buffering operation should be, at the same time, long enough to ensure jitter compensation for the entire duration of the playout, and short enough to avoid a long waiting for the user.

In the case of constant bitrate (CBR) media transmission over a constant throughput channel, the computation of the minimum pre-roll time is straightforward [1]; if instead the channel is variable, the buffering time should be longer with respect to the CBR case, to compensate throughput variability, therefore the user may experience a long waiting before enjoying the requested multimedia content. Furthermore, in many cases the channel statistics may be unknown, and this computation may be impossible.

Several solutions for receiver buffer underflow avoidance have been proposed in recent studies. A solution may be buffering for a given pre-roll time, either pre-determined or computed under the hypothesis of constant rate and constant throughput, and then reacting to channel variations whenever changes are detected. This operation may be performed in different ways. One of the possible techniques is *Dynamic Playback Rate* [2] which makes use of a threshold for data buffering. If data already buffered exceed the threshold, the media is played at its natural speed; if instead the amount of data buffered is below, the playback rate is reduced proportionally. A similar buffer management technique is the *Adaptive Media Playout* (AMP) [3]–[5] in which the client varies the playback rate, increasing and decreasing its speed, according to the network throughput (and consequently to the playout buffer fullness). In this sense, it extends Dynamic Playback Rate. Both approaches may experience problems if the buffer continues decreasing for a long time, since it is not possible to vary the playout speed beyond a certain limit without introducing excessive perceptual distortion. Techniques to forecast the channel capacity and delay, and to adapt the playout speed in advance, have been studied in [6], [7],

Other buffer underflow avoidance methods are based on the determination of the optimal buffer size. In [8], a buffer dimensioning technique to avoid buffer underflows is presented in the case of variable media rate and variable channel capacity. The problem was also studied, by means stochastic processes, for queues in a packet network [9]; if the arrivals and departure processes are known, then it is possible to evaluate the optimal buffer allocation. The management of source and receiver

This work was partially supported by Motorola Electronics S.p.A. MDB Development Center, Torino, Italy and by Centro Supercalcolo Piemonte (CSP), Torino, Italy

buffer dimensioning in case of a VBR media can be found also in [10].

Another family of techniques behaves as follows. The pre-buffering time is set to a given value, and in case the amount of data present within the receiver is insufficient, they force the source to change the coding rate. In [11], rate recomputing is performed considering both channel delay and loss rate; receiver buffer fullness is taken into account using a system of thresholds. In [12], the source rate is controlled according to the buffer fullness and channel throughput, for transmission over TCP.

In this paper, we propose a source media rate adaptation algorithm. It stores data during a pre-roll period which is determined under the hypothesis of a constant channel throughput; in case of channel variation, the media rate is recomputed taking into account both the new channel rate and the playout buffer fullness at the time of the event. This algorithm selects the new source media rate as the one which guarantees no buffer underflows (and consequently no media freezing), supposing that no further channel variation will occur; however, this recomputing can be safely reiterated at every network throughput change. We show that the buffer underflow probability can be extremely high in case of non-reactive approaches; to demonstrate this, we derive the analytical expression of the underflow probability, supposing channel samples as independent and identically distributed (i.i.d.).

The paper is organized as follows. In Section II, we describe the mathematical framework of the problem, and derive the expression of the buffer underflow probability at a generic time during the playout; we study how this probability is affected by modifications of the source rate. We further specialize this formulation to the constant channel throughput case. In Section III, we present an algorithm for rate selection, based on the above theory, according to the channel throughput and buffer fullness. We show the behavior of this algorithm for some throughput patterns in Section IV. Finally, we draw conclusions in Section V.

II. Problem formulation

Under the hypothesis of constant channel throughput C^* and constant media rate R^*, and considering $C^* < R^*$, data buffering at receiver is necessary to ensure a continuous playout; the buffering time t_B can be simply computed (see, e.g., [1]) as:

$$t_B \geq t_D \left(R^*/C^* - 1 \right), \tag{1}$$

where t_D indicates the duration of the stream to be played. If the receiver allows a pre-buffering period of this length, the underflow probability for time points $t \leq t_D$ is zero.

If the channel throughput diminishes and the media rate is not adapted accordingly, then the data buffered may not suffice to ensure continuous playout. The exact time position of the buffer underflow depends on the particular evolution of the channel and it can be located only if the channel throughput value is exactly known at each instant. Instead, if the specific time evolution is unknown but the statistic characterization of the channel is available, it is possible to compute the expression of the buffer underflow probability as a function of time.

In the development of this study, we suppose the time to be divided into slots of duration δ_t; the generic time instant t can be defined as

$$t = n_t \delta_t, \tag{2}$$

where n_t indicates the slot number. Similarly, we define the buffering time as

$$t_B = n_B \delta_t \tag{3}$$

and the media duration as

$$t_D = n_D \delta_t. \tag{4}$$

We suppose the media rate and the channel throughput to remain constant within each time slot, which is a reasonable assumption if the time δ_t is small (e.g., in the order of milliseconds).

After the pre-roll time t_B, the playout starts and the buffer is full up to a level of B_s bits. Under the hypothesis of average channel throughput μ_C during pre-roll, its value is

$$B_s \simeq \mu_C n_B \delta_t. \tag{5}$$

During slots $\in [n_B, n_B + n_D]$, information is taken from this buffer at constant rate R^* bps, while the arrival rate follows a stochastic process $C \sim f_C(x, n_t)$. After the playout starts, at a generic time slot n_t, the buffer fullness is:

$$B(n_t) = B_s - R^*(n_t - n_B)\delta_t + \delta_t \sum_{k=1}^{n_t - n_B} C(x,k), \tag{6}$$

where the random variable $C(x,k)$ is the stochastic process sampled at the given time slot k. According to the *central limit theorem*, and considering $C(x,k)$ independent and identically distributed (later, *i.i.d.*), it is:

$$\sum_{k=n_B}^{n_t} C(x,k) \sim \mathcal{N}((n_t - n_B)\mu_C, \sqrt{n_t - n_B}\sigma_C). \tag{7}$$

The probability of experiencing an underflow before a given slot is the probability of having less than zero bits within the buffer at any moment before t:

$$P_u(n_t) = P\left(B(n_t) < 0\right). \tag{8}$$

By definition, the buffer cannot contain a negative number of bits; the above condition represents the probability that the media playout requires, before a given time slot, more data than the amount contained in the buffer. In formulas, the probability of this event is:

$$\begin{aligned} P_u(n_t) \quad & \simeq P(B_s - R^*(n_t - n_B)\delta_t + \\ & + (\sqrt{n_t - n_B}\sigma_C Z + (n_t - n_B)\mu_C)\delta_t < 0), \end{aligned} \tag{9}$$

where $Z \sim \mathcal{N}(0,1)$. It is straightforward to obtain the mathematical expression of this quantity:

$$
\begin{aligned}
& P_u(n_t) = \\
= \;& P\left(Z < \frac{R^*(n_t - n_B) - B_s/\delta_t - \mu_C(n_t - n_B)}{\sqrt{n_t - n_B}\sigma_C}\right) \\
= \;& \Phi_Z\left(\frac{R^*(n_t - n_B) - B_s/\delta_t - \mu_C(n_t - n_B)}{\sqrt{n_t - n_B}\sigma_C}\right) \\
= \;& \frac{1}{2}\left(1 + erf\left(\frac{(R^* - \mu_C)(n_t - n_B) - B_s/\delta_t}{\sqrt{2}\sqrt{n_t - n_B}\sigma_C}\right)\right) \quad (10)
\end{aligned}
$$

where $\Phi_Z(x)$ is the cumulative density function (*cdf*) of $Z \sim \mathcal{N}(0,1)$.

To obtain the probability density function of the buffer underflow (indicated in the following as $p_u(n_t)$), it is sufficient to derive Formula (10). Indicating with $g(n_t)$ the argument of the erf function, it is:

$$
\begin{aligned}
p_u(n_t) &= \frac{\mathrm{d}P_u(n_t)}{\mathrm{d}n_t} \\
&= \frac{\mathrm{d}P_u(g(n_t))}{\mathrm{d}g(n_t)}\frac{\mathrm{d}g(n_t)}{\mathrm{d}n_t} \\
&= \frac{e^{-g^2(n_t)}}{\sqrt{\pi}}\frac{\mathrm{d}g(n_t)}{\mathrm{d}n_t}. \quad (11)
\end{aligned}
$$

The complete expression can be written easily from Formula (11).

To ensure that the buffer fullness does not reach zero at any time during transmission, it is sufficient to have p_u infinitesimal at each time slot $\in [n_B, n_B + n_D]$. To better clarify how to obtain this condition, we show some examples.

Let the initial (constant) rate of a media source be 100 kbps, and the channel throughput have statistics μ_C=80000 kbps and σ_C=20000. We suppose the sequence has a duration of 90 s, and from Formula (1) we estimate a pre-roll time $t_B = 22.5$ s. Under these conditions, the media stops the playout at time $t_E = t_B + t_D = 112.5$ s. Figure 1 shows the buffer underflow probability p_u around the point t_E.

It can be seen that, for $R^* = 100$ kbps, the buffer underflow probability starts being non-infinitesimal around time $t = 108$ s; this means that it is highly probable to have a freeze in the last 4.5 s of the playout. Figure 1 also shows the plots of $p_u(n_t)$ for rates in the range $R^* \in \{98.5, 99.0, 99.5\}$ kbps; we vary only the media rate, while the pre-buffering time is kept constant to 22.5 s. The effect of a reduction in the media rate is that the average point of $p_u(t)$ is shifted in the future, and therefore the buffer underflow probability for a generic instant $t \leq t_E$ diminishes. For $R^* = 98.5$ kbps, the buffer underflow probability is always infinitesimal inside the playout interval.

A. The CBR channel case

To further show the consistency of this formulation, we will compute the expression of the underflow probability as a function of time in the simplest case, when the channel is constant at rate C^*. This result cannot be obtained directly from Formula (11), since it corresponds $\sigma_C = 0$ and this quantity appears at denominator.

In this case, with constant media rate R^* and if we set the pre-roll time according to Formula (1), we showed that the underflow probability will be a delta function centered in t_E, since the first and only underflow event will be placed exactly at the end of the playout.

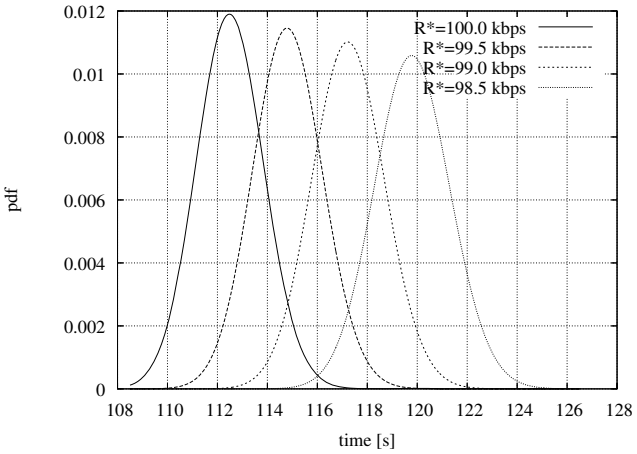


Fig. 1. Buffer underflow probability as function of time, for different values of R^*; the channel throughput statistics are $\mu_C = 80000$ and $\sigma_C = 20000$, the media lasts 90 s; pre-buffering time is set to 22.5 s and the media playout stops at 112.5 s.

Formula (10) needs some manipulation to be meaningful. If the channel is constant at rate C^*, then $f_C(x) \sim \delta(x - C^*)$; its variance is zero, the average is C^* and the argument of the *erf* is infinite. In particular, according to the sign of the numerator:

$$
\begin{aligned}
& \frac{(R^* - C^*)(n_t - n_B) - B_s/\delta_t}{\sqrt{n_t}\sigma_C} = \\
& = \begin{cases} +\infty \text{ if } (R^* - C^*)(n_t - n_B) - B_s/\delta_t \geq 0 \\ -\infty \text{ if } (R^* - C^*)(n_t - n_B) - B_s/\delta_t < 0 \end{cases} \quad (12)
\end{aligned}
$$

Given the definition of the error function erf, the effect on P_u becomes:

$$
P_u(n_t) = \begin{cases} 1 \text{ if } (R^* - C^*)(n_t - n_B) - B_s/\delta_t \geq 0 \\ 0 \text{ if } (R^* - C^*)(n_t - n_B) - B_s/\delta_t < 0 \end{cases} \quad (13)
$$

Remembering that in this case it is $B_s = C^* n_B \delta_t$ the condition to determine the sign of the infinity becomes:

$$
\begin{aligned}
& (R^* - C^*)(n_t - n_B) - C^* n_B < 0 \Rightarrow \\
& \Rightarrow (R^* - C^*) n_t - R^* n_B < 0 \Rightarrow \\
& \Rightarrow n_t < \frac{n_B R^*}{R^* - C^*} \quad (14)
\end{aligned}
$$

Subtracting n_B at both members:

$$
n_t - n_B < n_B \frac{C^*}{R^* - C^*}. \quad (15)
$$

We set the number of buffering slots at the minimum as defined in Formula (1):

$$
n_B = n_D\left(R^*/C^* - 1\right) = n_D \frac{R^* - C^*}{C^*}. \quad (16)
$$

Substituting in Formula (14), we get:

$$n_t - n_B < n_D \Rightarrow n_t < n_B + n_D. \quad (17)$$

This means that:

$$P_u(n_t) = \begin{cases} 1 \text{ if } n_t \geq n_D + n_B \\ 0 \text{ if } n_t < n_D + n_B \end{cases} \quad (18)$$

Since $P_u(n_t)$ is a step function, with step in $n_D + n_B$, its derivative is

$$p_u(n_t) = \delta(n_t - (n_D + n_B)), \quad (19)$$

which represents the slot in which the transmission ends. This means that the buffer underflow probability is zero in all of the slots before the media playout end time, and this coincides with the result we expected.

III. Proposed rate adaptation algorithm

We showed that, if the channel varies over time and the media rate remains constant, the probability of having an underflow can be computed in closed form.

The proposed formulation involves four quantities, the channel average throughput μ_C and standard deviation σ_C, the media rate R^* (supposed constant) and the buffering slots n_B. The first two, related to the channel, are not dependent on the application and can only be measured; the media rate and the buffering time can be properly set to avoid buffer underflows. Using the same pre-roll time and channel characteristics, a decrease in the media rate is helpful in diminishing the underflow probability for time slots during the playout; an increase in the rate produces the opposite effect.

We also showed that, if the channel throughput is considered constant, the pdf of the underflow probability degenerates into a delta function; as in the case of variable channel, modifying the media rate it is possible to drive the peak of this delta outside the playout interval.

Until now, we considered that the channel characteristics (μ_C and σ_C) remain constant during the experiments. In this Section, we show that it is possible to apply the same formulation, if a channel variation is detected, at any point in time during the playout. The pre-roll duration is set supposing the channel will remain stationary during pre-roll and transmission; if for any reason the average channel throughput diminishes, or its variance increases significantly, and the media rate is not adapted accordingly, the quantity of buffered data may not suffice to ensure continuous playout. In this case, it is necessary to recompute the media rate *during* the playout.

In the following, we will suppose a CBR channel throughput; extension to the case with variance can be similarly derived and involves a heavier notation.

Since the computation is made my means of the formulation of Section II, the new rate at which the source should encode the stream takes into consideration both the new channel throughput value and the buffer fullness. When the channel change is detected, the quantities μ_C and σ_C are measured; the buffer fullness is known at the receiver and the remaining parameter R^* can be determined by means of Formula (11).

Packets already buffered at the receiver are not discarded, but the request for a new media rate is immediately sent to the source; therefore, since several seconds of data may be present within the buffer, the effects of a rate change are not immediate in the playout quality. The media rate must be requested at the time t_{req}, but will be effective under the decoder point of view at time t_{pl}, after all the packets at the old rate are played; for this reason, the buffer level to be used in our formulation is the one at time t_{pl}, which needs to be predicted.

Supposing the channel throughput will not change again during the interval $t_{diff} = t_{pl} - t_{req}$, then the buffer at time t_{pl} can be computed from the buffer at t_{req}, by means of a linear relationship:

$$B(t_{pl}) = B(t_{req}) - (R_{old} - C_{new}) * t_{diff}, \quad (20)$$

since data will be taken from the buffer at the old media rate R_{old} and added at the new channel throughput C_{new}. The rate which should be used to avoid buffer underflows after the time t_{pl} is given by:

$$R_{new} = C_{new} + \frac{B(t_{req}) - (R_{old} - C_{new}) * (t_{diff} + t_{RT})}{t_D + t_B - t_{pl} - t_{RT}}. \quad (21)$$

Formula (21) means that, to ensure the buffer goes to zero at the playout end time, the new rate R_{new} should be equal to the new channel throughput C_{new} plus a margin given by the quantity of buffered data at the moment the packets at the new rate will be played ($B(t_{req}) - (R_{old} - C_{new}) * (t_{diff} + t_{RT})$) divided by the time remaining until the end of the playout time ($t_D + t_B - t_{pl} - t_{RT}$). The quantity t_{RT} represents the *round trip time*, and takes into account the propagation delay of the new rate request. In the following, this parameter will be set to zero for simplicity.

The new rate is the one that forces the peak of p_u at time t_E, under the assumption that no more channel throughput changes will occur until the end of the playout; if other changes happen, the above approach can be iterated and the rate modified again.

Formula (21) may be used also to increase the rate in case the channel throughput grows. In this case, the new rate allows better quality; the rate is again chosen to ensure the end of the transmission coincident with the end of the playout.

The delay introduced between the new rate request and its effect at the decoder is not critical if changes occur in the final part of the sequence, since t_{diff} reflects the quantity of already buffered data, which is smaller at the end of the playout. The only limitation to the successful application of this technique is when the channel changes at a time distance smaller than t_{RT} from the end.

IV. Results

To test the correctness of the proposed rate adaptation system, we performed several buffer simulations. We present here the behavior of the adaptive system, along with the results obtained with a non-adaptive approach.

A. Scenario I, short channel degradation during playout

In the first scenario, transmission is performed over a channel which diminishes its rate for some seconds during the media playout. The channel starts at 400 kbps, it drops

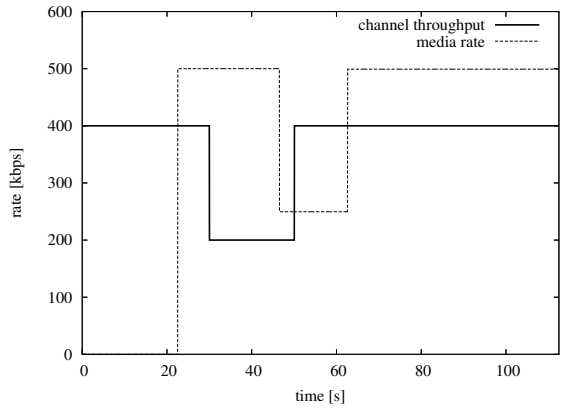


Fig. 2. Channel throughput and playout media rate obtained with the proposed rate control method; the channel diminishes for some seconds during the playout.

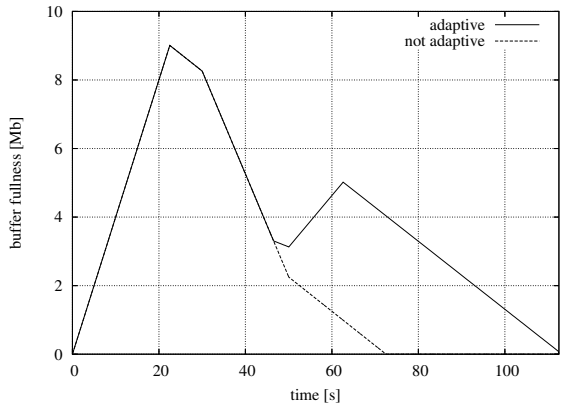


Fig. 3. Buffer evolution for the setting of Figure 2.

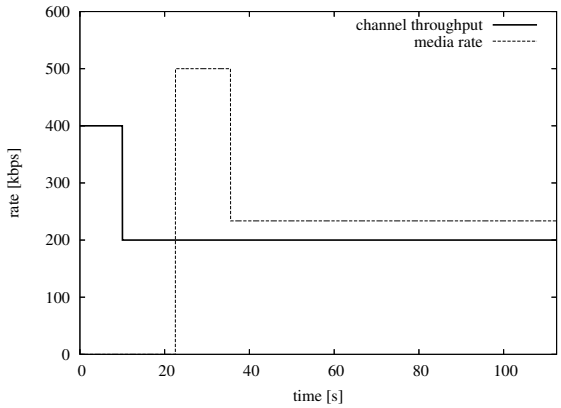


Fig. 4. Channel throughput and playout media rate obtained with the proposed rate control method; the channel throughput diminishes during the pre-roll and remains low until the end of the playout.

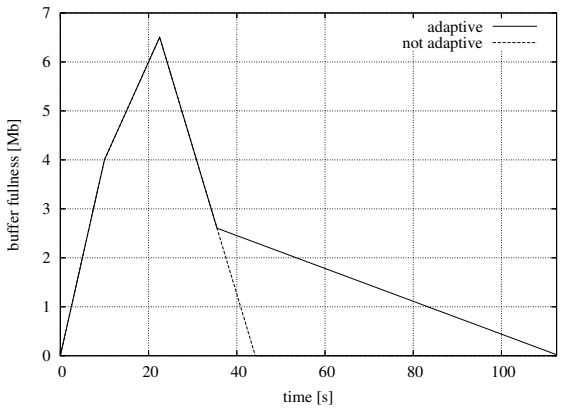


Fig. 5. Buffer evolution for the setting of Figure 4.

at 200 kbps after 30 s, and grows again to 400 kbps after 20 s; after this, it remains constant, as shown in Figure 2. The initial media rate is set to $R^* = 500$ kbps; from Formula (1), the pre-roll time is $t_B = 22.5$ s. This period is included in the interval in which the channel remains constant at 400 kbps, therefore no channel throughput changes occur during pre-roll. In Figure 2, the media rate seen at the decoder is shown. It remains zero during the buffering time, then it evolves adapting to the channel conditions.

The rate remains constant at 500 kbps for a certain time after the channel throughput changes; this is the effect of the packets already buffered at that rate. At the time the channel drops ($t = 30$ s), the buffer contains packets for 16 s; this quantity is equivalent to t_{diff} of Formula 21. When all of those packets are played, the decoder starts playing the part of the stream received at lower rate; according to the formulation, the new rate must be $R_{new} = 249.5$ kbps. When the throughput increases again, there are packets for $t_{diff} = 12$ s in the buffer. After this period, the rate increases again, and this time the rate value is set to $R_{new} = 499$ kbps.

The buffer fullness evolution under these conditions is shown in Figure 3. The buffer grows until $t_B = 22.5$ s at constant rate; at $t = 30$ s, the channel throughput changes and the buffer level starts decreasing fastly, because packets arrive at smaller rate while the decoder pops packets at the same speed as before. At $t = 46$ s, the decoder starts consuming information at the new rate, and the buffer decrease slows down. While the decoder is running at $R^* = 249.5$ kbps, the channel increases again, and information is received faster than it is consumed; in this interval, the buffer fullness grows up again, and this allows the buffer to avoid decoder starvation until the end. Note that we suppose that the source has always enough packets stored to always transmit at the channel rate throughput.

For comparison, Figure 3 shows the curve of buffer fullness in the case of a non-adaptive media rate, which remains constant at $R^* = 500$ kbps; in this case, the buffer goes in underflow well before the media playout end time.

B. Scenario II, permanent channel degradation during pre-roll

In the second scenario, the channel starts at a rate $C^* = 400$ kbps, and after 10 s it decreases to 200 kbps ans remains constant at this value, as shown in Figure 4. The media rate starts at $R^* = 500$ kbps and the pre-buffering period is $t_B = 22.5$ s, therefore the channel throughput change occurs during pre-roll. In this case, we decide to wait until the end of the buffering period before recomputing the rate. When the

playout starts, the request for a new rate is sent to the source and, as in the previous scenario, after a certain time t_{diff}, it sets to $R^* = 233.5$ kbps until the end.

The buffer evolution in this scenario is shown in Figure 5. It can be seen how the buffer growth rate changes during the pre-roll period. After the beginning of the playout, the buffer fullness decreases fastly until information is not available at the new rate. Figure 5 also shows that if the rate is not adapted, the buffer underflow condition is reached about 20 s after the playback starts.

V. Conclusions

In this paper, we analyzed the receiver buffer underflow problem. First, we derived the analytical expression of the underflow probability at each time instant, given an initial buffering time and the channel statistics, and supposing the media rate constant. Based on this computation, we showed that whenever the channel average throughput or variance vary significantly, this probability may excessively grow and therefore produce poor-quality decoded streams several seconds before the media playout end time. We also showed that small modifications in the media rate value can be effective in decreasing the buffer underflow probability to infinitesimal values. We specialized the formulation in the case of a constant channel throughput, and demonstrated that in this case the underflow probability becomes a delta function.

We proposed a rate adaptation algorithm, based on the previous formulation, which ensures the center of the delta function to be located at the end of the playout. If packets have been buffered to compensate for a given throughput, and the channel changes with respect to this value, then the buffer fullness may decrease fastly and the media may freeze; we use the proposed algorithm to recompute the new media rate at which the source should encode the remaining part of the stream, to avoid decoder starvation. This approach can be reiterated every time a channel throughput change is detected.

The receiver computes the new desired rate according to the new channel condition, taking into account also the quantity of data already present in the buffer. the value is then fed back to the encoder.

This algorithm showed to achieve the goal of avoiding buffer underflows at the expenses of a variability in the bitrate. The proposed algorithm is effective also in case of multiple channel variations, even when they occur during the pre-buffering time, and introduces a negligible complexity into the system.

References

[1] Tanir Ozcelebi, A. Murat Tekalp, and M. Reha Civanlar, "Optimal rate and input format control for content and context adaptive video streaming," in *Proceedings of International Conference on Image Processing*, Singapore, October 2004, vol. 3, pp. 2043–2046.

[2] Yu G. Chen Maria C. Yuang, Shih T. Liang and Chi L. Shen, "Dynamic video playout smoothing method for multimedia application," in *Proceedings of IEEE International Conference on Communications*, Dallas, USA, June 1996, vol. 3, pp. 1365–1369.

[3] Mark Kalman, Eckehard Steinbach, and Bernd Girod, "Adaptive playout for real-time media streaming," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, Scottsdale, AZ, USA, May 2002, vol. 1, pp. 45–48.

[4] Eckehard Steinbach Mark Kalman and Bernd Girod, "Adaptive media playout for low-delay video streaming over error-prone channels," *IEEE Transaction on Circuits and systems for video technology*, vol. 14, no. 6, pp. 841–851, June 2004.

[5] Aziz Shallwani and Peter Kabal, "An adaptive playout algorithm with delay spike detection for real-time voip," in *Proceedings of Canadian Conference on Electrical and Computer Engineering (CCECE)*, Montreal, Canada, May 2003, vol. 2, pp. 997–1000.

[6] Philip DeLeon abd Cormac J. Sreenan, "An adaptive predictor for media playout buffering," in *Proceeding of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Phoenix, USA, March 1999, pp. 3097–3100.

[7] Prathima Agrawal, Jyh-Cheng Chen, and Cormac J. Sreenan, "Use of statistical methods to reduce delays for media playback buffering," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, Austin, TX, USA, June 1998, pp. 259–263.

[8] Hrvoje Jenkac, "Buffering aspects for variable bitrate video streaming," in *Proceedings of Joint Workshop on Communications and Coding*, Barolo, Italy, November 2002, p. 23.

[9] Venkat Anantharam, "The optimal buffer allocation problem," *IEEE Transaction on Information Theory*, vol. 35, no. 4, pp. 721–725, July 1989.

[10] Mahesh Balakrishnan, "Buffer constraints in variable-rate packetized video system," in *Proceedings of International Conference on Image Processing (ICIP)*, Washington, USA, October 1995, vol. 1, pp. 29–32.

[11] Aylin Kantarci, Nukhet Ozbek, and Turhan Tunali, "Rate adaptive video streaming under lossy network conditions," *Signal Processing: Image Communication*, vol. 6, no. 19, pp. 479–497, July 2004.

[12] S. C. Liew L.S. Lam, Jack Y. B. Lee and W. Wang, "A transparent rate adaption algorithm for streaming video over the internet," in *Proceedings of Conference on Advanced Information Networking and Applications*, Fukuoka, Japan, March 2004.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

Dissemination of Dynamic Multimedia Content in Networked Virtual Environments

Tolga Bektaş, Farzad Safaei, Iradj Ouveysi and Osman Oğuz

***Abstract*— This paper aims to design an infrastructure for distribution of dynamic multimedia objects in a Networked Virtual Environment (NVE) based on a distributed proxy architecture. Dynamic objects (such as audio, video and images) are generated by the users on-the-fly, are often short lived and must be disseminated to others in their area of interest while the application is running. It is expected that a significant portion of network traffic for future NVE applications will be due to these objects. However, because of the upstream bandwidth constraints of clients, a peer-to-peer model for dissemination of these objects is unlikely to be scalable. We develop a distributed server infrastructure for this purpose and produce a mathematical model for optimal establishment of proxy servers, assignment of clients to proxies, and replication pattern of objects among the proxies. The formulation is non-linear (quadratic)in both the objective function and one set of constraints. We provide a technique to linearize the model, which provides the opportunity to solve reasonable size problems for benchmarking. In addition, we provide a fast and efficient heuristic that can be used to obtain near-optimal solutions to the problem in real-time. Finally the paper concludes with computational results showing the performance of the linearization procedure and the heuristic algorithm on randomly generated Internet topologies. It is shown that our heuristic algorithm produces solutions that are within 20% of the optimal.**

***Index Terms*— networked virtual environment, integer programming, heuristic algorithm.**

I. Introduction

THERE has been a significant increase in popularity of Networked Virtual Environments (NVE) in recent years. It has been estimated that by 2009 more than 230 million people will be playing multiplayer network games [1]. Other collaborative applications for work, education and entertainment are also likely to become widespread.

NVEs represent a possible approach for utilizing the power of Internet for enhancing group communication and interaction, where several participants (perhaps hundreds or thousands) can enter a common virtual environment for the purpose of sharing an experience, collaboration, recreation and so forth. Within the virtual environment, the participants are typically represented by their *avatars*. Each participant can control his/her avatar using a suitable interface (keyboard and mouse, game controller, gestures, etc.). Through this control and depending on the design and purpose of the application, participant's avatar may move within the virtual environment, interact with objects or other avatars and perform a variety of tasks to achieve its goals. The interactivity, therefore, is not confined to a web-style point-and-click and involves a more substantive and immersive kind - i.e., virtual presence and participation.

In the current generation of NVE's, the visual and audio scenes associated with the virtual environment are commonly comprised of static content only. The images of avatars, the environment (walls, rooms, mountains, trees), and other objects are created offline either by application designers or, in some cases, the users themselves and disseminated beforehand. Likewise, the audio objects such as the sound of closing a door, an explosion, waves or wind are pre-recorded and distributed with the application software. (In some cases, the sound may also be created synthetically using a suitable model for the physics of objects and collision.) Hence, the visual and audio scenes are created by the end clients based on information that is available locally and the real-time exchange of information across the network is primarily limited to the *state information* exchange.

The state information is used to communicate changes in the state of the virtual environment due to actions of participants (moving, rotating, shooting, etc). The aims is to create a consistent perception for all. Significant research effort in recent years has been devoted to design of suitable models for traffic characterization and processing of state information so that the NVE can be deployed cost effectively, accommodate a large number of players and scale with respect to geographical spread of participants (see [2] and references therein).

We are interested in situations where a significant portion of multimedia content within the virtual environment is *dynamic*. By 'dynamic' we mean that these objects are created on-the-fly, are often short lived and must be disseminated to others while the application is running. To a limited extend this is already happening. Most NVE's support some form of communication mechanism such as text chat or a single channel voice 'party line'. This information is often broadcast to everyone or a subset of participants. More advanced systems for immersive voice communication have been described in [3] and commonly require a separate server infrastructure for creation and delivery of personalized audio scenes for each avatar. In the future, we expect to see more dynamic objects including video and images in these environments. These could include images of participants, artifacts created by the users and shared, presentation slides or vide clips for education

T. Bektaş and O. Oğuz are with the Department of Industrial Engineering of Bilkent University, 06800, Ankara, Turkey (e-mail: tolgab@bilkent.edu.tr, ooguz@bilkent.edu.tr)

Farzad Safaei is with the School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, NSW 2522, Australia (email: farzad@uow.edu.au).

I. Ouveysi is with ARC Special Research Centre for Ultra-Broadband Information Networks (CUBIN), Department of Electrical and Electronic Engineering, University of Melbourne, Australia (email: iradjouveysi@yahoo.co.uk).

or collaboration, and gesture and haptics information. As the multimedia communication aspect of NVE's become richer and more immersive, having a common delivery platform for distribution of dynamic multimedia objects, as opposed to designing separate infrastructure for each, becomes more critical.

A conceptually simple model for dissemination of dynamic multimedia objects is to use a peer-to-peer model. In this case, the source of the dynamic object will have to multicast the object to all other clients that might be interested in this object. For example, audio and visual information should be streamed to every other avatar who happen to be within the hearing or visual range. This model has scalability issues especially with respect to access and network bandwidth utilization and latency of dissemination.

This paper aims to develop a suitable model based on a distributed proxy architecture to support dissemination of dynamic multimedia objects associated with the networked virtual environments. In the next Section, we describe the distributed proxy architecture of interest to this paper and identify the characteristics and requirements for effective dissemination of content. In Section 3, a mathematical model is developed to optimize the choice of proxy servers, assignment of clients to proxies, and replication of multimedia objects among the proxies. We will then use a suitable technique to linearize the formulation and develop a heuristic model for fast and efficient solution. The computation results and conclusions are presented next.

II. System Description

There is clearly some similarity between dissemination of dynamic multimedia objects amongst the participants of a NVE and content distribution networks (CDN) for delivery of web or streaming content in today's Internet. There are, however, some stark differences especially with respect to temporal and spatial aspects of demand distribution. The following considerations are relevant.

- The lifetime of dynamic multimedia objects may range from immediate (real-time streams like voice and video) to short-lived (images or presentation slides) to semi-permanent (new artifact added to the virtual environment). Consequently, timeliness and bandwidth cost of object replication among the distributed servers is a critical factor in designing the infrastructure.
- Within the virtual environment, a dynamic multimedia object is of relevance only to a subset of participants. For example visual objects are visible within a range and need not be distributed to all users. This 'area of interest' of an object depends on many contextual factors. The characteristics of the 'map' of the environment will impact the visual and audible range of objects (for example, presence of walls, doors, and buildings). The 'rules' of the virtual environment is also of relevance (for example, capabilities of avatars to zoom in on remote objects or obtain high-speed transportation). In addition, the 'area of interest' may differ for different types of media (audio and music, for example, could propagate through walls but not video). The combination of above factors leads to a rather complex demand model where the user's demand for dynamic objects depends more on their location and behaviour within the virtual environment than their physical distribution around the Internet. So in addition to dynamic content, the demand pattern is also highly dynamic due to often rapid movement of avatars in the virtual world.
- In CDN, the origin servers owned by the content provider contain the original content, which is then distributed for replication among the surrogates. For NVE's, the origins of dynamic objects are users/clients. It is reasonable to assume that the upstream bandwidth of the clients is not very large. Hence the 'cost' of download directly from the origin is very high.

In this paper, we propose to use a distributed proxy architecture for dissemination of dynamic multimedia objects within a NVE. This architecture has already been used for immersive voice communication [4] and our intention here is to extend the concept to the more general case of multimedia communications. We assume that the service provider has access to a set of distributed servers - referred to as proxies - across the Internet. In general, the service provider might be hosting more than one NVE, hence, a given NVE might use only a subset of proxies for the distribution and delivery of its dynamic objects. This is because the number and distribution of clients associated with each NVE has a significant influence on the choice of proxies needed. During different time periods, however, these characteristics might change. For example, the number of participants in an entertainment NVE may increase during night time. In this case, it might be suitable to add new proxies to the service. This addition has certain overhead and configuration cost that we refer to as the proxy "establishment" cost.

Every client must be assigned to an established proxy server. Each proxy, in turn, is responsible for a group of clients and performs the necessary functions of distribution, replication/catching of objects. The combination of these proxies will form an overlay network of distributed servers.

It is reasonable to assume that the network path connecting each client to its assigned proxy is known and cannot be controlled by us. (This usually is an Internet routed path from the user through its ISP Point of Presence and its quality is influenced by the topology of ISP infrastructure and subscription options available.) The service provider, however, can have some control over the communication paths between the proxies. Obviously, each proxy can communicate with every other proxy using an Internet routed path. However, in some cases the service provider could hire better quality communication paths between a subset of proxies. As an example, a Virtual Private Network provided by a carrier could be used to connect some proxies together if available/economical. For our purpose, it is sufficient to assume that there is connectivity between all proxies, but some of the overlay paths may offer better quality and bandwidth resources than others (usually at additional cost).

As stated above, we assume that clients are the *source* of dynamic objects but due to upstream bandwidth limitations,

only one copy is sent to their assigned proxy. The proxy then becomes the 'origin server' for this object for the purpose of replication among other proxies for the duration of object's lifetime. The clients are also the source of demand for other objects. This demand pattern, however, is highly dynamic and heavily dependent on the context of the virtual environment. To simplify the task of estimating demand, we partition the virtual environment into a number of *communication zones* for each type of object (audio, video, etc.). The avatars within the same communication zone would require each others objects. Within the lifetime of an object, there is also a possibility of other avatars moving into its communication zone. As such, we require a demand estimator that provides an estimate of demand from each avatar for a given dynamic object for the time period of interest(which could be in the order of tens of seconds to hours depending on the dynamics of application). The method of partitioning the virtual environment into communications zone and estimation of demand based on roaming behavior of avatars is outside the scope of this paper and will be subject of future publications.

III. Integer Programming Formulation

The problem of interest to this paper can be stated as follows. Given a particular set of dynamic objects created by the participants of a NVE within a time period, and given the estimated demands for these objects from other clients in this time period, our aim is to determine: (i) the subset of the distributed proxies that should be established for dissemination of these objects; (ii) assignment of each client to a proxy; and (iii) the replication pattern of objects among the proxies. Note that we have not included any request routing decision here. We have assumed that on the short time scales of interest to us, the practical approach for a client would be to obtain the required object from its own proxy (if the object is replicated there) or to fetch it directory from the originating client. The latter option, however, is assumed to incur a higher cost and will be subject to upstream bandwidth constraint of the originating client. The formal definitions and formulation of this problem are presented next.

We consider a fully meshed network $G = (V, E)$, where V is the set of nodes and $E = (\{i, j\} : i, j \in V)$ is the edge set representing the overlay paths connecting the nodes together. The node set V is further partitioned into two nonempty, mutually exclusive and exhaustive subsets as $V = I \cup P$, where I is the set of clients, and P is the set of potential nodes on which proxy servers can be installed. The known parameters of the problem are given in Table I.

To construct the integer programming formulation, we further define the following binary decision variables:

$$y_p = \begin{cases} 1, & \text{if a proxy server at node } p \in P \\ & \text{is established} \\ 0, & \text{otherwise} \end{cases}$$

$$x_{ip} = \begin{cases} 1, & \text{if client } i \in I \text{ is assigned to} \\ & \text{proxy server } p \in P \\ 0, & \text{otherwise} \end{cases}$$

$$z_{pk} = \begin{cases} 1, & \text{if proxy server } p \in P \text{ holds object } k \in K \\ 0, & \text{otherwise} \end{cases}$$

TABLE I

Parameters of the problem

Notation	Indication
c_{ij}	The unit cost of transferring an object over the path $\{i, j\} \in E$.
$c_{i,j(k)}$	The unit cost of transferring an object from the client who has generated the object k (denoted by $j(k)$) to a client i that requests the object directly. This is used in the case that a content requested by client i is not found in its assigned proxy server, and the client is forced to retrieve it from the originating client.
s_p	The capacity of the potential server at site p.
f_p	The establishment cost of server at site p.
K	The set of objects available in the system for a short time interval of interest.
K_i	The set of objects generated by client i in the system within the short time interval of interest ($K_i \subset K$).
b_k	The storage requirement of object $k \in K$.
β_k	The bandwidth requirement of object $k \in K$.
d_{ik}	The demand for object $k \in K$ by client $i \in I$.
B_i	The upstream bandwidth capacity of client i available for direct peer-to-peer transfer of objects generated in i to other clients.

We now present an integer programming formulation for the problem as follows (hereafter denoted by F):

$$\text{minimize} \sum_{p\in P} f_p y_p + \sum_{i\in I}\sum_{p\in P}\sum_{k\in K} b_k d_{ik} c_{ip} z_{pk} x_{ip} + \sum_{i\in I}\sum_{p\in P}\sum_{k\in K} b_k d_{ik}(1 - z_{pk}) c_{i,j(k)} x_{ip} \quad (1)$$

s.t.

$$\sum_{p\in P} x_{ip} = 1, \qquad \forall i \in I \quad (2)$$

$$x_{ip} \le y_p, \qquad \forall i \in I, p \in P \quad (3)$$

$$\sum_{k\in K} b_k z_{pk} \le s_p y_p, \qquad \forall p \in P \quad (4)$$

$$\sum_{i\in I, i\neq l}\sum_{p\in P}\sum_{k\in K_l} \beta_k x_{ip}(1 - z_{pk}) \le B_l \qquad \forall l \in I \quad (5)$$

$$y_p \in \{0, 1\}, \qquad \forall p \in P \quad (6)$$

$$x_{ip} \in \{0, 1\}, \qquad \forall i \in I, p \in P \quad (7)$$

$$z_{pk} \in \{0, 1\}, \qquad \forall p \in P, k \in K \quad (8)$$

The objective function is given in (2). Here, the first summation represents the total cost of establishing proxy servers for this NVE during this time period. The second summation denotes the total cost of delivery of objects to the clients. The first part of this summation is for the case when client i receives content k that is located in its own proxy p (reflected by the cost $b_k d_{ik} c_{ip} z_{pk} x_{ip}$ summed over all the proxies, clients and objects). In the case when the requested object is not located in its proxy server, an additional cost incurs to further request the object from the generating client. This is reflected in the second part of the summation by $b_k d_{ik}(1 - z_{pk}) c_{i,j(k)} x_{ip}$, over all proxies, clients and objects. Similar cost functions have also been suggested in [5], [6], [7], and [8]. Constraint (3) implies that a client can be assigned to a node only if a proxy server is established on that node. Constraint (4) implies that the total size of objects held in each proxy

server is constrained by the available capacity. Constraint (5) is related to the bandwidth capacity between the clients. More specifically, this constraints represents the upstream capacity limit of clients to transfer objects directly to other clients who could not fetch it from their own proxy. Finally, constraints (6)–(8) denote the integrality of the decision variables.

Formulation F belongs to the class of $\mathcal{NP}$-Hard problems (see Appendix I). There are several ways to solve the F presented above. One strategy would be to use techniques specifically devised for quadratic optimization problems. Another way is to linearize the formulation that will enable one to solve the resulting linearized formulation via a commercial optimization package. This is presented in the following section.

IV. Model Linearization

It is clearly seen that the objective function (2) of F contains a quadratic term due to the multiplication of the x_{ip} and z_{pk} variables. We propose a transformation procedure to linearize the model using a continuous variable and two sets of constraints. The following is the basis of our linearization.

Proposition 1: The following constraints are sufficient to linearize the preceding model,

$$\varphi_{ipk} \leq x_{ip}, \qquad \forall i \in I, p \in P, k \in K \tag{9}$$

$$\varphi_{ipk} \leq z_{pk}, \qquad \forall i \in I, p \in P, k \in K \tag{10}$$

where $\varphi_{ipk} = z_{pk}x_{ip}$ and is a continuous variable in $[0, 1]$.

The proof of Proposition 1 is provided in Appendix II. Note that the linearizing variable φ_{ipk} is actually an indicator of whether client i is connected to the proxy server p *and* the proxy server holds the requested object k or not. Based on the preceding proposition, we can now substitute the quadratic term $x_{ip}z_{pk}$ that is present in both the objective function (2) and constraints (5) by the linearization variable φ_{ipk}. Under the proposed linearization, the integer linear programming formulation of the problem can be given as follows (hereafter denoted by F_L):

$$\begin{aligned}\text{minimize} \sum_{p\in P} f_p y_p + \sum_{i\in I}\sum_{p\in P}\sum_{k\in K} b_k d_{ik} c_{i,j(k)} x_{ip} \\ + \sum_{i\in I}\sum_{p\in P}\sum_{k\in K} b_k d_{ik} \varphi_{ipk}(c_{ip} - c_{i,j(k)})\end{aligned} \tag{11}$$

s.t.

$$\begin{aligned}
\sum_{p\in P} x_{ip} &= 1, && \forall i \in I \\
x_{ip} &\leq y_p, && \forall i \in I, p \in P \\
\sum_{k\in K} b_k z_{pk} &\leq s_p y_p, && \forall p \in P \\
\sum_{i\in I, i\neq l}\sum_{p\in P}\sum_{k\in K_l} \beta_k(x_{ip} - \varphi_{ipk}) &\leq B_l && \forall l \in I \qquad (12) \\
y_p &\in \{0, 1\}, && \forall p \in P \\
x_{ip} &\in \{0, 1\}, && \forall i \in I, p \in P \\
z_{pk} &\in \{0, 1\}, && \forall p \in P, k \in K \\
\varphi_{ipk} &\in [0, 1], && \forall i \in I, p \in P, k \in K
\end{aligned}$$

The (linearized) formulation F_L can now be solved using any integer programming solver. However, bearing in mind that the number of users and the objects existent in a NVE can be huge, solving the preceding formulation for such a data set during short time periods of interest would be impractical. Therefore, we develop a heuristic procedure in the next section that will be used to solve the problem fast, is scalable, and provides reasonably good solution quality.

V. A Heuristic Algorithm

Given any ordering $\{1, 2, \ldots, |P|\}$ of the set of potential proxy locations, the heuristic begins with establishing the first proxy in the first iteration and continues on establishing the p^{th} proxy of the ordering in the p^{th} iteration, until all the potential proxies are engaged to the system. The ordering chosen here can be such that the proxies are sorted in the nondecreasing order of f_p's or the nonincreasing order of s_p's. At every iteration, users are connected to an established proxy that incurs the minimum cost. Then, the objects are replicated based on what we refer to here as the *saving* of an object $k \in K$, calculated as follows:

$$\pi_{pk} = b_k c_{p,j(k)} \sum_{i\in N(p)} d_{ik} \tag{13}$$

where $N(p)$ denotes the set of clients connected to proxy p. A similar measure has also been used by Xuanping et al. [9]. Verbally, the saving of an object is the amount of cost reduced by placing object k on a proxy p. Then, the objects are sorted in the non-increasing order of their savings. Let $\{O_{p1}, O_{p2}, \ldots\}$ denote this order. The objects are placed in the available proxy server(s) using this order without violating the capacity constraints. We also use $b(L)$ to denote $\sum_{k\in L} b_k$. The outline of the heuristic is given in the following:

Heuristic procedure

0. Let the best solution be $\overline{c} = +\infty$ and $l = 1$.
1. Repeat the following while $l \leq |P|$:
   1.1. Select the first l proxies to be opened, (denoted by $P \supseteq \overline{P} = \{1, \ldots, l\}$).
   1.2. for each client $i \in I$
   let $x_{ip^*} := 1$ such that $\sum_{k\in K} b_k d_{ik} c_{ip^*} = min_{j\in\overline{P}}(\sum_{k\in K} b_k d_{ik} c_{ip})$
   1.3. for each server $p \in \overline{P}$
   Sort the objects and let the ordering be $\{O_{p1}, O_{p2}, \ldots, O_{p|K|}\}$.
   $L := \emptyset$.
   $t := 1$
   while $b(L) \leq s_p$
   begin
   $k := O_{pt}$
   if $b(L \cup \{k\}) \leq s_p$
   $z_{pk} := 1$
   $L := L \cup \{k\}$
   $t \leftarrow t + 1$
   end
   1.4. Check current solution. If feasible, then

Calculate the cost of the current solution c^l.
If $c^l < \overline{c}$, set $\overline{c} = c^l$.

1.5. $l \longleftarrow l + 1$.

Here, set L given in the algorithm is used to record the set of objects located on a proxy. At step 1.4., the solution is checked with respect to constraint (5). If the solution does not satisfy this constraint, it is discarded. On the other hand, if the solution does satisfy this constraint and is better than the current solution with respect to cost, it is set as the incumbent solution.

VI. Computational Results

In order to assess the computational performance of the proposed model and the heuristic algorithm, we have performed a set of computational experiments. These experiments are based on randomly generated Internet topologies, using an Internet topology generator, available at the web address http://topology.eecs.umich.edu/inet/, which mimics the characteristics of the real Internet topology. The instances used for comparison purposes have different number of potential proxy locations, clients and objects (denoted by $|P|$, $|I|$, and $|K|$ respectively). A number of nodes are randomly selected to be potential proxy nodes and client nodes for each topology. The size of each object is chosen from a uniform random variable between 0 and 1. The fixed cost of installing a proxy server is also a uniform random variable between 600 and 1000. The capacity of each proxy server is calculated as 50% of the total size of all objects. The cost of download from the originating client is set to be equal to 5 times the cost of download from its proxy. The demand distribution for the objects have been modelled using a Zipf-like distribution. The Zipf-like distribution assumes that the probability of a request for an object is inversely proportional to its popularity. More specifically, let a number of objects be ranked in order of their popularity where object i in this order is the i^{th} most popular object. Then, given an arrival for a request, the conditional probability that the request is for object i is given by $P_K(i) = \frac{\Omega}{i^\alpha}$, where $\Omega = (\sum_{i=1}^{K} \frac{1}{i^\alpha})^{-1}$ is a normalization constant and α is an exponent. When $\alpha = 1$, we have the true Zipf-distribution. In [10], it is shown that α varies from 0 to 1 for different access patterns and is usually between 0.64 and 0.83 for web objects. This value was recorded to be $\alpha = 0.733$ for multimedia files in [6]. We have used this specific value in our implementation. The metric used to assess each solution is a *normalized cost* metric, as used in other studies (e.g. [11]), which is defined as *normalized cost = cost of the network output by the procedure/cost of the network without any replicated proxies*. Here, the cost of the network without any replicated proxies is the scheme where all the clients are assumed to retrieve the required objects from the originating clients directly. Note that the smaller the normalized cost, the better the solution found by our procedure.

For the experiments, random problems have been generated with 3 to 5 potential proxy locations, 10 clients and 20 to 80 objects. We solve each problem once using the linearized formulation F_L by employing a state-of-the-art commercial optimization package CPLEX 9.0 running on a Sun UltraSPARC 12x400 MHz with 3 GB RAM, and once with the heuristic algorithm. We report our findings in Table II. In this table, the first three columns correspond to the number of potential proxy locations, number of clients and number of objects, respectively. The next column, v_{model} reports the value of the optimal solution of the F_L whereas the next column (Time (seconds)) presents the corresponding computational solution time. Column $v_{heuristic}$ reports the value of the best solution found by the heuristic algorithm. Finally, the last column indicates how much the solution found by the heuristic algorithm deviates from that of the formulation. This value is calculated as $\frac{v_{heuristic} - v_{model}}{v_{model}} * 100$.

TABLE II
Computational analysis of the proposed model and the heuristic algorithm

$\|P\|$	$\|I\|$	$\|K\|$	v_{model}	Time (seconds)	$v_{heuristic}$	dev
3	10	20	0.069922	21	0.081356	16.35
3	10	30	0.067774	52	0.074079	9.30
3	10	40	0.064923	74	0.076072	17.17
3	10	50	0.061548	227	0.072084	17.12
3	10	60	0.06113	128	0.073112	19.60
3	10	70	0.06002	626	0.066189	10.28
3	10	80	0.057473	718	0.068427	19.06
5	10	20	0.036788	121	0.041944	14.02
5	10	30	0.033101	214	0.035707	7.87
5	10	40	0.034180	409	0.037644	10.13
5	10	50	0.031248	503	0.034908	11.71
5	10	60	0.031922	1496	0.036109	13.12
5	10	70	0.030649	2217	0.036493	19.07
5	10	80	0.029493	2770	0.033503	13.60

As the figures provided in Table II indicate, it is not practical to use F_L to obtain a solution to the problem, even for small sized instances such as those considered in Table II. This can be seen by observing that the computational time required to solve F_L to optimality increases heavily as the problems grow in size. On the other hand, the greedy heuristic seems to perform fairly well. We do not report on the corresponding time required by the heuristic algorithm, since this is below 1 second for all the instances considered in Table II). Furthermore, based on the data given in the last column of this table, we can state that the heuristic is able to produce fairly good solutions in very short computational times.

VII. Conclusions

Timely dissemination of dynamic multimedia objects in Networked Virtual Environments to all clients who might be interested in these object is a challenge when the upstream bandwidth of the clients is limited and the cost and delay associated with distribution is important. In this paper, we have developed an integer programming model that can be used on short time intervals for the purpose of optimally locating proxy servers, identifying the replication pattern of objects among the servers and assigning clients to the proxies so as to minimize the total transfer cost of the content. Only when the required object is not found in the proxy, a client will be forced to fetch it from the originating client. Our formulation includes a constraint on the upstream capacity of the clients to reflect the realistic scarcity of access bandwidth.

Since the proposed model includes a quadratic objective function and constraints, we have made use of a linearization technique to convert the formulation to a linear model. This is solved as a benchmark to test the performance of our heuristic algorithm. It is shown that our heuristic algorithm produces results which are close to optimal based on a set of random problems. The efficiency and effectiveness of the algorithm makes it suitable to run online and produce near-optimal solution for short time intervals of interest.

APPENDIX I
PROOF OF THE COMPLEXITY OF THE PROBLEM

Proposition 2: The problem F is $\mathcal{NP}$-Hard.

Proof: We prove the proposition by restriction (see [12]). where the following instance of the problem is considered. Let $K = \{1\}$ (i.e. there is only a single object), $b_1 = b$, $d_{i1} = d_i$ and $s_p \geq b$, $\forall p \in P$ (i.e. all the proxies have a sufficiently large capacity). Since there are no capacity constraints in this case, constraints (4) and (5) become redundant and $z_{p1} = 1$, $\forall p \in P$. In this case, the resulting problem becomes the Uncapacitated Facility Location Problem (FLP), which is known to be $\mathcal{NP}$-Hard (see e.g. [13]). ■

APPENDIX II
PROOF OF PROPOSITION 1

Proof: After simplifying the objective function of F as $\sum_{i\in I}\sum_{p\in P}\sum_{k\in K}(b_k d_{ik} c_{i,j(k)} x_{ij} - b_k d_{ik}(c_{ip} - c_{i,j(k)})\varphi_{ipk})$, where $\varphi_{ipk} = z_{pk} x_{ip}$, the proof relies on the observation that the coefficient of φ_{ipk} in the objective function is $-b_k d_{ik}(c_{ip} - c_{i,j(k)})$, which is always negative. By definition, φ_{ipk} should be 1 if and only if $z_{pk} = 1$ and $x_{ip} = 1$, and 0 for all other cases. Now, assume that $z_{pk} = 1$ and $x_{ip} = 1$ for a specific (i, p, k) triplet. Then, according to constraints (9) and (10), φ_{ipk} is only constrained by the upper bound 1 and the minimizing objective function implies $\varphi_{ipk} = 1$. In all other cases (i.e. $x_{ip} = 1$, $z_{pk} = 0$; or $x_{ip} = 0$, $z_{pk} = 1$; or $x_{ip} = 0$, $z_{pk} = 0$) constraints (9) and (10) together imply $\varphi_{ipk} = 0$. ■

REFERENCES

[1] D. Intelligence, "The Online Game Market 2004," *California, USA*, August 2004.

[2] S. F. B. P. Brun, J., "Distributing Network Game Servers for Improved Geographical Scalability," *Telecommunications Journal of Australia*, vol. 55, pp. 23–32, March 2005.

[3] S. F. D. M. Boustead, P., "DICE: Internet Delivery of Immersive Voice Communication to Crowded Virtual Spaces," *Proceedings of IEEE International Conference on Virtual Reality, VR 2005, Bonn, Germany*, March 2005.

[4] S. F. Dowlatshahi, M., "A Recursive Overlay Multicast Algorithm for Distribution of Audio Streams in Networked Games," *Proceedings of the IEEE International Conference on Networks (ICON 04), Singapore*, November 2004.

[5] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 568–582, 2000.

[6] M. Yang and Z. Fei, "A model for replica placement in content distribution networks for multimedia applications," in *Proceedings of IEEE International Conference on Communications (ICC '03)*, vol. 1, 2003, pp. 557 –561.

[7] A. Datta, K. Dutta, H. Thomas, and D. VanderMeer, "World Wide Wait: a study of Internet scalability and cache-based approaches to alleviate it," *Management Science*, vol. 49, no. 10, pp. 1425–1444, October 2003.

[8] T. Bektas, O. Oguz, and I. Ouveysi, "Designing cost-effective content distribution networks," *Computers and Operations Research*, 2005, in press.

[9] Z. Xuanping, W. Weidong, T. Xiaopeng, and Z. Yonghu, *Data Replication at Web Proxies in Content Distribution Network*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2003, vol. 2642, pp. 560–569.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proceedings of IEEE INFOCOM'99*, vol. 1, New York, March 1999, pp. 126–134.

[11] J. Xu, B. Li, and D. Lee, "Placement problems for transparent data replication proxy services," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1383–1398, 2002.

[12] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Franciso, California: W. H. Freeman and Company, 1979.

[13] G. Cornuejols, G. Nemhauser, and L. Wolsey, "The uncapacitated facility location problem," in *Discrete Location Theory*, P. Mirchandani and R. Francis, Eds. John Wiley & Sons, 1990, ch. 3, pp. 119–171.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

First Video Streaming Experiments on a Time Driven Priority Network

Mario Baldi and Guido Marchetto
Department of Control and Computer Engineering
Politecnico di Torino (Technical University of Torino)

Abstract — As broadband access becomes more widely available and affordable, future Internet traffic will be dominated by streaming media flows, such as video-telephony, video-conferencing, high definition TV, 3D video, virtual reality, and many more. Consequently, networks will have to offer quality of service with scalable solutions — i.e., currently relied-upon overprovisioning is not likely to be a viable solution to accommodate streaming media traffic. This paper describes a testbed and experiments demonstrating the deployment time-driven priority scheduling — an implementation of pipeline forwarding — to support video streaming. The purpose of the presented experiments is to intuitively show the benefits the proposed solution provides to UDP-based streaming applications, while preserving efficient support for elastic TCP-based traffic.

Index Terms— Quality of service, UDP-based streaming, Testbed and experimentation, Efficient utilization of network resources, Traffic engineering

I. INTRODUCTION

High speed Internet access has by now become widely available potentially opening a large market to new services, that are potential new source of revenue for an ailing telecom market. Many service providers offer VoIP based telephony services, video broadcasting, and video on demand. Such applications are often referred to as *multimedia* or *real-time* as, contrary to traditional data applications, the timing of packet delivery is important for them to work properly. Packet networks, traditionally designed and deployed for data applications are not engineered to tightly control the delay packet experience in routers where they might contend for resources (e.g., transmission capacity) and consequently queued.

Right now the requirements of multimedia applications are commonly satisfied through *overprovisioning*, i.e., by keeping the network lightly loaded so that contention for network resources is low and queuing time consequently small. This approach is feasible as only a small fraction of broadband access subscribers are currently using such multimedia services and they deploy their Internet connection with traditional applications, such as web browsing and e-mail. Consequently, users do not fully deploy the large bandwidth of their access connections and both access and backbone networks are currently lightly loaded. Although some users more heavily exploit their broadband access with peer-to-peer file sharing applications, these do not require real-time service. As a consequence, the above overprovisioning approach can still be applied if coupled with traffic differentiation, e.g., according to the Differentiated Services solution [1], to separate and prioritize multimedia traffic. However, this approach is not any longer feasible if multimedia traffic grows to become dominant and technology does not evolve fast enough to enable a proportional enhancement of the network infrastructure.

Pipeline forwarding [2] is particularly suitable to carry streaming media applications over the Internet since it offers:

1. Quality of service guarantees (deterministic delay and jitter, no loss) for (UDP-based) constant bit rate (CBR) and variable bit rate (VBR) streaming applications — as needed;
2. Support of elastic, e.g., TCP-based, traffic — i.e., existing applications based on "best-effort" services are not affected in any way;
3. High scalability of network switches (multi-terabit/s in a single chassis) [3],

This paper reports on the first experiments of video streaming through a testbed network of routers supporting time-driven priority (TDP) scheduling [2] that is an implementation of pipeline forwarding. The aim and contribution of this paper is to demonstrate in an intuitive and visual way, i.e., through the user perceived quality of the video stream played at the receiver, that

- The prototypal router implementation with TDP support works properly, thus providing the expected quality of service, and
- Multimedia streaming applications can benefit from it

Notice that although the experiments presented in this work have been done with one way streaming video, the results and considerations in this paper apply to interactive media as well, where the short and constant end-to-end delay observed in the experiments is even more critical.

Section II focuses on pipeline forwarding, the technology underlying the presented testbed, by presenting its operating principles and properties and how it can be deployed in current packet networks. The testbed on which the presented experiments are run is detailed in Section III that describes its architecture and the implementation of its main component, an IP router implementing TDP scheduling. Section IV describes the experiments, including their setup and outcome. Lesson learned and future research directions are discussed in Section V.

The presentation of this work was supported by the European Commission under the E-Next Project FP6-506869

II. Underlying Principles and Technologies

A. Pipeline Forwarding: Time-Driven Priority

Pipeline forwarding is a known optimal method that is widely used in computing and manufacturing. The necessary requirement for pipeline forwarding is having common time reference (CTR). In the presented prototypal router UTC (coordinated universal time) is used for CTR, consequently, the method used in the testbed is called *UTC-based pipeline forwarding*. An extensive and detailed description of UTC-based forwarding is outside the scope of this paper and is available in [2].

In UTC-based pipeline forwarding all packet switches are synchronized and utilize a basic time period called time frame (TF). The TF duration (T_f) may be derived, for example, as a fraction of the UTC second received from a time-distribution system such as the global positioning system (GPS) and, in the near future, Galileo. As shown in Fig. 1, TFs are grouped into time cycles (TCs) and TCs are further grouped into super cycles, each super cycle lasts for one UTC second. TFs are partially or totally reserved for each flow during a resource reservation phase. The TC provides the periodicity of the reserved flow. This result in a periodic schedule for IP packets to be switched and forwarded, which is repeated every TC.

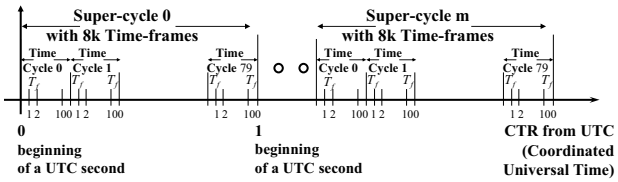


Fig. 1. Common time reference structure

The basic pipeline forwarding operation is regulated by two simple rules: (*i*) all packets that must be sent in TF *t* by a node must be in its output ports' buffers at the end of TF *t*-1, and (*ii*) a packet *p* transmitted in TF *t* by a node *n* must be transmitted in TF $t+d_p$ by node $n+1$, where d_p is an integer constant called *forwarding delay*, and TF *t* and TF $t+d_p$ are also referred to as the *forwarding TF* of packet *p* at node *n* and node $n+1$, respectively. The value of the forwarding delay is determined at resource-reservation time and must be large enough to satisfy (*i*). In pipeline forwarding, a *synchronous virtual pipe* (SVP) is a predefined schedule for forwarding a pre-allocated amount of bytes during one or more TFs along a path of subsequent UTC-based switches

UTC-based forwarding guarantees that reserved real-time traffic experiences: (*i*) bounded end-to-end delay, (*ii*) delay jitter lower than two TFs, and (*iii*) no congestion and resulting losses.

Time-driven priority (TDP) [2] is a synchronous packet scheduling technique that enables combining UTC-based pipeline forwarding with conventional routing mechanisms to achieve the high flexibility together with guaranteed service. While scheduling of packet transmission is driven by time, the output port can be selected according to either conentional IP destination-address-based routing, or multi-protocol label switching (MPLS), or any other technology of choice.

B. Non-pipelined Traffic

Non-pipelined (i.e., non-scheduled) IP packets — namely packets that are not part of a SVP (e.g., IP best-effort packets) — can be transmitted during any unused portion of a TF, whether it is not reserved or it is reserved but currently unused. Consequently, links can be fully utilized even if flows with reserved resources generate fewer packets than expected. Moreover, any service discipline can be applied to packets being transmitted in unused TF portions. For example, various traffic classes could be implemented for non-pipelined packets in accordance to the Differentiated Services (DiffServ) model [1]. In summary, pipeline forwarding is a best-of-breed technology combining the advantages of circuit switching (i.e., predictable service and guaranteed quality of service) and packet switching (statistical multiplexing with full link utilization) that enables a true integrated services network providing optimal support to both multimedia and elastic applications.

C. Multimedia System Architecture

Fully benefiting from UTC-based pipeline forwarding requires providing network nodes and end-systems with a CTR and implementing network applications in a way that they use it to maximize the quality of the received service. However, the Internet is currently based on asynchronous IP packet switches and hosts. Thus, especially in an initial deployment phase, UTC-based pipeline forwarding must coexist and interoperate with current equipment and applications (e.g., IP video-phones, video-streaming servers and clients, etc.), as depicted in Fig. 2. The experiments presented in this work reproduce such scenario: senders generate asynchronous multimedia traffic then entering a TDP domain. Edge TDP routers are connected to traditional asynchronous IP nodes through a SVP (synchronous virtual pipe) interface that polices and shapes incoming traffic flows. Specifically, asynchronous packets are stored in a buffer waiting for their previously evaluated forwarding TF.

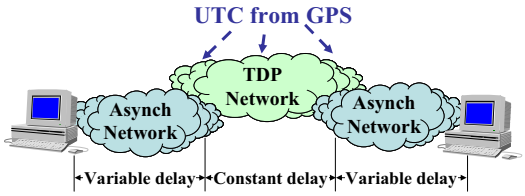


Fig. 2. Interoperation between TDP and asynchronous networks

III. Testbed

The network architecture for the interoperation of TDP and asynchronous networks presented in the previous sections has been demonstrated by building the testbed for video distribution shown in Fig. 3. In particular, we aim at showing the effectiveness of UTC-based pipeline forwarding by means of the quality of the service perceived by a viewer of a streaming video routed though the TDP network along the path highlighted in Fig. 3. Although current experiments have been done with one way streaming media, the results and considerations apply to interactive media as well, where short and constant end-to-end delays are even more critical

requirements.

A. Architecture and Components

The testbed, which reproduces the network scenario in Fig. 2, consists of various asynchronous end-systems — implemented by two personal computers (PCs) and a router tester — interconnected by a UTC-based pipeline forwarding network consisting of four TDP routers. The two end systems, 2.4 GHz Pentium IV personal computers running Linux Fedora Core 3, implement a distribution system for one-way video based on the Fenice ver. 1.9 video server software and the Nemesi ver. 0.5.2 video streaming client [4]. A video stream enters the TDP network through TDP Router 1 and reaches the client through the other three TDP routers along the path shown in Fig. 3.

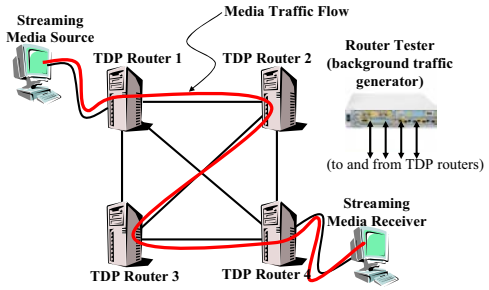


Fig. 3. Experimental testbed

All network interface cards deployed in the testbed are Intel PRO/1000 MT Gigabit Ethernet server adapters operating at 100 Mb/s. An Agilent N2X Router Tester is used to generate the two types of asynchronous flows described below that act as background traffic.

Delay insensitive flows — possibly modeling traditional Internet data traffic such as file transfers and e-mail exchanges — enter the network as non-pipelined traffic and are handled as best-effort traffic in the TDP network. This type of background traffic is used to experimentally verify that the capability to perfectly isolate pipelined traffic from the non-pipelined one has been properly implemented in TDP routers. In fact, a large amount of non-pipelined traffic, possibly overloading the network, is expected not to affect the service provided to pipelined traffic.

Delay sensitive flows — possibly modeling real-time traffic such as voice over IP (VoIP), video on demand, and videoconferencing — are handled as pipelined traffic in the TDP network. This type of background traffic is used to demonstrate the ability of pipeline forwarding to guarantee deterministic quality of service (QoS) also in case most — potentially 100% — network resources (e.g., transmission bandwidth) are dedicated to traffic with specific QoS requirements, e.g., real-time service. This is a significant improvement over other QoS approaches:

- The DiffServ model [1] assumes that differentiated traffic is only a small fraction on the network capacity;
- Conventional (asynchronous) techniques for guaranteeing service performance in packet networks [5], possibly adopted in the context of the Integrated Services model [6], do not allow to fully load the network with packet flows that require short delay and jitter, especially if they are low rate flows (see [7] for a detailed discussion).

B. TDP Router Implementation

The core of the testbed is the TDP network composed of four TDP routers. These are based on the routing software of the FreeBSD 4.8 operating system running on a 2.4 GHz Pentium IV PCs; the TDP scheduling algorithm is implemented in the FreeBSD kernel [8].

Generically, a router data plane moves packets from input ports to output ports through three modules that perform *input processing*, *forwarding*, and *output processing*, respectively. The operations performed by each module of a TDP router are briefly discussed in the following (see [8] for details).

The input module of an interface connected to another TDP router determines the forwarding TF of each pipelined packet by adding the forwarding delay to the estimated forwarding TF at the previous node. The current router implementation leverages on the DS field [1] of the IP header to

- Distinguish pipelined packets from non-pipelined packets
- Ensure that the estimate of the forwarding TF is correct even in case multiple packets are lost (see Section III.B of [8] for protocol details).

The forwarding module processes packets according to the specific network technology; the presented prototype performs conventional IP routing and forwarding.

Consequently, TDP support concerns mainly the output module where a per-TF, per-output queuing system is needed to store packets while waiting for their forwarding TF to begin. The queue in which each packet ends up is determined by both the input module — deciding the forwarding TF — and the forwarding module — determining the output interface. The output module also responsible for the timely transmission of all the packets stored in the queues corresponding to a TF when it begins and before it ends.

The input module of SVP interfaces (i.e., interfaces of a boundary TDP router not connected to a pipeline forwarding node) includes mechanisms to

- Classify each incoming packet to identify the data flow it belongs to
- Determine, based on the flow's resource reservation, the TF in which the packet should be forwarded by the output module (i.e., its forwarding TF).

UTC is provided to our prototypal router by a Symmetricom GPS receiver PCI card that can generate interrupts at a programmable rate ranging between less than 1 Hz (1PPS — pulse per second) and 250 kHz (every 4 μs). Such interrupts are used to pace the beginning of TFs; whenever an interrupt occurs the values of the current TF and TC are updated. The current version of the prototype does not implement signaling functions, i.e., TFs and TCs are statically allocated to flows through manual configuration.

IV. Experiments

A. Basic System Parameters

The current router implementation does not support preemptive priority, i.e., the capability of interrupting the transmission of a non-pipelined packet in a non-disruptive

way when a new TF begins (see Section 2.2 of [2] for details). Consequently, if a portion of a TF is not used by pipelined traffic and a non-pipelined packet does not fully fit in it, the current implementation leaves the link idle, thus lowering link utilization. As this would not happen in a full-fledged implementation with support for preemptive priority, system parameters are chosen to avoid the above situation: packet size is chosen such that the transmission of an integer number of packets lasts (basically) a whole TF. Having set the TF duration to 250 µs, 25,000 bits (i.e., about 3 KB) can be transmitted during one TF and 1 KB packets are deployed..

B. *Resource Reservation*

The needed amount of bandwidth should be reserved in proper portions of TFs for the reference real-time flow depicted in Fig. 3, a 10 Mb/s MPEG video stream. As the deployed video server generates 1 KB packets, capacity for at most three video packets could be reserved during each TF. Since the video stream results from encoding 25 video frames per second, the average frame size is about 49 KB. However, an MPEG codec produces a significantly different amount of bits for a frame depending on which of the following encoding it is using[1].

Intra-frame Coding eliminates spatial redundancy inside pictures and the resulting encoded picture is called *I-frame*.

Predictive Coding eliminates temporal redundancy between a picture and the previous one through motion estimation. The obtained encoded picture is called *P-frame* and it is typically from 2 to 4 times smaller than an I-frame.

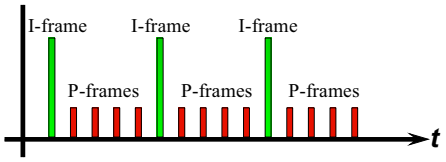


Fig. 4. MPEG video stream

Normally codecs apply intra-frame coding and predictive coding on different frames according to a fixed, re-occurring pattern, as depicted in Fig. 4. However, I-frame and P-frame size is generally unpredictable. Moreover, some codecs can use intra-frame coding and predictive coding on different portions (macroblocks) of the same frame. This results in a packet flow with a very variable bit rate.

Having configured the TC to be composed of 160 TFs, one video frame is transmitted every TC. A network analyzer was used to observe the traffic corresponding to the video stream deployed in our experiments in order to determine how much transmission capacity to reserve during each TC. The maximum size of a video frame, i.e., the maximum burst size, resulted to be about 100 KB. In order to minimize end-to-end delay while avoid packet loss (i.e., in order to ensure that a whole frame can be transferred during a single TC) the capability of transmitting 100 KB (i.e., 100 packets) each TC must be ensured. For example, capacity for transmitting 3 packets could be booked during 34 TFs. This results in bandwidth overallocation (about 20 Mb/s for a 10 Mb/s flow) and low efficiency in the utilization of network resources. However, such issue is beyond the scope of the current experiments that primarily aim at verifying the correct operation of the system. Section V discusses ways to improve utilization of reserved resources, which is key in engineering a scalable solution.

In order to limit the variation of the delay introduced by the SVP interface on video packets, the TFs in which resources are allocated should be as evenly distributed as possible across the TC. Issues related to minimizing the jitter due to the SVP interface are outside the scope of this work.

C. *Background Traffic Pattern*

Delay sensitive background traffic is generated by the router tester as a set of CBR flows with destination and bit rate chosen to maximize TF utilization and contention on the links traversed by the streaming video. Such links can host up to 78.75 Mb/s of additional pipelined traffic to be transmitted during the 126 TFs not reserved to the video stream.

A possible solution to have delay sensitive background traffic fully load the links traversed by the streaming video flow is to define ten 7.875 Mb/s CBR flows that follow the same path. However, such a traffic pattern does not maximize resource contention, which potentially causes long queuing delays. In fact, after packets from the video stream and other delay sensitive flows have contended for transmission on the link between TDP router 1 and TDP router 2 in Fig. 3, they can stream through the subsequent links in the same order without further contention, hence without being possibly queued. This obviously results in limited delay and jitter even if no particular QoS oriented scheduling mechanisms, whether TDP or conventional ones, are deployed in output modules. Consequently, the experiment would have little significance[2].

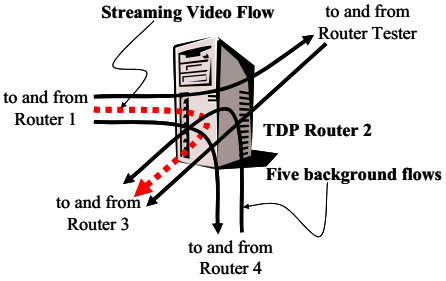


Fig. 5. Link contentions on TDP Router 2

A more complex traffic scenario is therefore used in the presented experiments. Thirty 7.875 Mb/s flows enter the TDP network from the various TDP routers and follow paths defined in such a way the reference video flow competes with other ten delay sensitive flows at each hop, as shown in Fig. 5 for TDP router 2. The dotted line represents the video stream, while each of the continuous lines represents a group of five delay sensitive background flows. The ten background flows sharing an output link with the video stream arrive from different input links and consequently actually contend for the

[1] Actually, a third type of encoding, called bi-directional predictive coding exists; without loss of generality, it was not considered in this work.

[2] Notice that as far as TDP is concerned, its principles of operation are such that contention is avoided in any case. Realizing a scenario in which contention naturally occurs is essential in order to (*i*) verify proper operation of the TDP implementation and (*ii*) demonstrate TDP theoretical properties.

transmission capacity as their packet arrival processes are, in general, independent. Once they reach TDP router 3, the 2 groups of five background flows follow different paths, thus contending again with the streaming video but for different links.

According to the described traffic scenario, the total amount of bandwidth reserved on the links traversed by the video stream is 99.15 Mb/s (out of 100 Mb/s). Unused bandwidth, i.e., parts of TFs, are deployed for transmitting delay insensitive background traffic that is provided with a best-effort service, i.e., queued in a queue served by a FIFO (first in first out) scheduler. Delay insensitive background flows follow similar routes as delay sensitive ones in order to maximize contention for them as well. As several 10 Mb/s flows are generated, links are overloaded and a significant amount of packets is discarded.

D. Jitter Control and Compensation

Replay buffers are commonly implemented in media streaming clients to absorb the jitter experienced by packets across the network. Since the network topology in the presented testbed is very simple (hence routers have few interfaces), even with QoS unaware packet scheduling algorithms, such as FIFO, jitter does not grow very large in spite of the complex traffic patterns defined for maximizing resource contention. Although replay buffers with typical sizes would certainly suffice to absorb the jitter accumulated in the presented testbed, this is not the case in general.

On the other hand, the jitter on a TDP network is very low and independent of the path (i.e., number of nodes) and traffic. In order to offer a user perceivable demonstration of the effectiveness of TDP in limiting jitter, the replay buffer size is minimized. Specifically, in our experiments the replay buffer is set to 1,096 bytes — which is the minimum size allowed by the deployed client software. As 1 KB packets are generated by the video server, the content of each packet is decoded immediately as the corresponding packet is received, without waiting for the replay buffer to fill up in order to compensate possibly late packets.

E. Results

In all the experiments run in the described scenario the video stream is replayed at the receiver with optimal user perceived quality and unnoticeable delay. As no visible losses occur, neither router output buffers nor video client replay buffer overflow, i.e., TDP scheduling actually limits packet jitter as expected.

This result is especially significant considering that about 89% of the network capacity is on average used by delay sensitive traffic — each 100 Mb/s link traversed by the 10 Mb/s video stream carries an additional 79 Mb/s of delay sensitive background traffic. This result is quite far from what could be achieved with a DiffServ approach [1] that heavily relies on the assumption that differentiated (e.g., delay sensitive) traffic is only a small fraction of the network capacity. Also conventional (asynchronous) QoS techniques [5], possibly adopted in the context of the Integrated Services model [6], hardly enable delay sensitive traffic to account for 89% of the link capacity, while guaranteeing short delay and jitter. This is especially hard if delay sensitive traffic is composed of low rate flows [7]; although the experiments presented here are related to a high rate video stream, the achieved results are independent of the flow rate, as it can be easily inferred from the simple TDP operating principles [2]. Finally, TDP is a very simple and scalable scheduling discipline, while conventional QoS techniques with best properties (e.g., WFQ) feature high implementation complexity and suffer from limited scalability (i.e., applicability in large scale, significant scenarios).

Measurements taken in experiments run on a similar network topology demonstrated TDP properties in terms of limited jitter and expected delay, independently of the number of hops, also when links where fully loaded (not “just” 90%) by (synthetic) delay sensitive traffic [8].

V. Discussion and Improvements

As previously mentioned, in the presented experiments the network could not be fully loaded because, because a large amount of bandwidth is allocated to the video stream as a simple way of coping with the unpredictability of its rate. Specifically, enough capacity is allocated during each TC to transmit a maximum size video frame. This results in allocating twice the average video stream rate; an even larger ratio of allocation over average rate could result for video streams that have few very detailed and/or very fast scenes that result in few frames much larger than all the others. This approach was used only as a “quick fix” to enable us to obtain first results that visually demonstrate both proper operation of the testbed and TDP properties. However, the approach goes against the very principles that motivated this work as it results in low utilization of reserved resources, i.e., low *reservation efficiency* as it is called in the context of this paper. The following subsections discuss various ways to maximize reservation efficiency. Although they are not implemented in the current testbed, some of them are the object of ongoing work.

A. Limited allocation without losses and large delay

Capacity is booked in each TC so that the allocated rate is larger than the video stream rate (although smaller than the reservation deployed in the presented experiments). When the amount of packets generated by the video server for a video frame is larger than the capacity booked during a TC, the exceeding packets can be transmitted by the SVP interface in the following TC (or TCs). Consequently, packets are buffered in the SVP interface for a time, possibly spanning multiple TCs (i.e, video frame periods), that depends on the burstiness of the video stream and the reservation. Moreover, as the delay experienced at the SVP interface by the packets of the video stream is highly variable, a correspondingly large replay buffer is required at the client.

A larger capacity allocation reduces the delay (and jitter) experienced by packets and buffering requirements at the SVP interface, but lowers reservation efficiency. In general, this solution is not suitable for interactive applications, such as telephony and videocoferencing, but could be applied for one way streaming media possibly featuring a low bit rate (in

order to limit buffer requirements at the SVP interface).

B. Limited allocation with possible loss and limited delay

Delay could be reduced by relaxing the requirements on loss, i.e., by allowing the possibility that a certain percentage of packets be lost. In particular, the size of the queue for the video stream at the SVP interface is limited and possibly overflowing packets are either discarded or forwarded in the network as non-pipelined traffic. Obviously, the quality of the video played at the client will be degraded depending on the allocated capacity in each TC, the burstiness of the video stream, the size of the queue at the SVP interface, and the network load.

The user perceived quality can be improved, while keeping the same delay bound (i.e., queue size and allocation), by

- Handling non-pipelined traffic according to the DiffServ model;
- Taking into account the perceptive importance of the video packets when deciding which ones to forward in the network using TDP and which ones to handle as non-pipelined traffic.

The latter approach, which we consider very promising, needs to be defined in detail, thoroughly studied, and validated, which is the object of ongoing work.

C. Optimal allocation without losses and optimal delay

As argued in [9], an optimal solution for the transmission of (interactive) media is obtained by synchronizing to UTC the video server (or video codec in case of real-time video). In fact, in this case the allocation can be minimized while optimizing the delay introduced by the network and applying UTC-based forwarding to all video packets. In particular, a different amount of capacity can be allocated in different TCs following the pattern of I-frames and P-frames. For example, with reference to the sample video stream depicted in Fig. 4 and given a super-cycle of 25 TCs (as it is in our experiments):

- The capacity needed to send the amount of bytes encoding an I-frame is allocated during TCs 0, 5, 10, 15, and 20. In order to minimize end-to-end delay, the allocation should be done during subsequent TFs.
- The capacity required to send the amount of bytes encoding a P-frame is allocated during the remaining TCs. In order to minimize end-to-end delay, the allocation should be done during the first TFs allocated in TCs 0, 5, 10, 15, and 20.

In order to minimize end-to-end delay, the video server should be programmed to (encode a frame and) generate the packets corresponding to a frame during the TFs reserved to the video stream. Additionally in case of real-time video, the codec should be programmed to finish encoding each video frame right before the set of allocated TFs.

However, as pointed out before, the size of I-frames, as well as P-frames, is not constant. This can be handled in one of the following ways:

- A reservation larger than the average stream rate is performed by allocating the size of the largest frame for both I-frame and P-frame reservation. Reservation efficiency might still be quite high compared to the solution described in Section IV.B because the size difference between I-frames, as well as P-frames, is typically smaller than the one between I-frames and P-frames. However, this solution might be impractical because, especially in real-time (e.g., interactive) video applications, the maximum size of encoded frames cannot be known at resource reservation time.
- Allocation is performed based on the, possibly estimated, average frame size and packets exceeding the reservation are either discarded or forwarded in the network as non-pipelined traffic. The resulting quality degradation can be controlled by
  - deploying the DiffServ model for non-pipelined traffic,
  - taking into account the perceptive importance of the video packets when deciding which ones to forward in the network as non-pipelined traffic
  - dynamically adjusting the resource reservation based on the actual size of encoded frames, i.e., the characteristics of the video scenes.
- In case of real-time video, the codec is modified to take into account the resource reservation and generate an amount of bytes for each encoded video frame as close as possible to the corresponding reservation [9], while maximizing the user perceived quality. In addition, in order to optimize the perceived quality the resource reservation could be adjusted based on the feedback from the codec on the characteristics of the video scenes.

All the approaches presented in this section are considered very promising as long term solutions (as they require UTC-aware end systems and applications) and will be subject of our future research.

ACKNOWLEDGMENT

The authors wish to thank Flavio Bonatesta for setting up the testbed to run the presented experiment.

REFERENCES

[1] S. Blake *et al*., *An Architecture for Differentiated Services*, IETF Std. RFC 2475, Dec. 1998.

[2] C.-S. Li, Y. Ofek, and M. Yung, Time-driven priority flow control for real-time heterogeneous internetworking, in *Proc. IEEE (INFOCOM' 96), vol. 1*, (Mar. 24–28, 1996), 189–197.

[3] M. Baldi, Y. Ofek, "Multi-Terabit/s IP Switching with Guaranteed Service for Streaming Traffic," *IEEE INFOCOM 2006 High-Speed Networking Workshop*, Barcelona (Spain), Apr. 2006.

[4] (LS)³ - Libre Streaming, Libre Software, Libre Standards. An open multimedia streaming project, "(LS)³ Tools," [Online] Available at: http://streaming.polito.it/tools

[5] H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks," *Proceedings of the IEEE*, Vol. 83, No. 10, Oct. 1995.

[6] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: an Overview," *IETF Std. RFC 1663*, July 1994.

[7] M. Baldi, F. Risso, "Efficiency of Packet Voice with Deterministic Delay," *IEEE Communications Magazine*, Vol. 38, No. 5, May 2000, pp. 170-177.

[8] M. Baldi, G. Marchetto, G. Galante, F. Risso, R. Scopigno, F. Stirano, "Time Driven Priority Router Implementation and First Experiments," *IEEE International Conference on Communications (ICC 2006), Symposium on Communications QoS, Reliability and Performance Modeling*, Istanbul (Turkey), June 2006.

[9] M. Baldi and Y. Ofek, "End-to-end Delay of Videoconferencing over Packet Switched Networks," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 4, Aug. 2000, pp. 479-492.

This full text paper was peer reviewed by subject matter experts for publication in the MULTICOMM 2006 proceedings.

Open Issues in P2P Multimedia Streaming

Djamal-Eddine Meddour
France Telecom R&D
2, Avenue Pierre Marzin, 22307, Lannion CEDEX
Lannion, France
djamal.meddour@francetelecom.com

Mubasher Mushtaq, Toufik Ahmed
LaBRI, University of Bordeaux 1
351, cours de la Libération, 33405, Talence Cedex
Bordeaux, France
{mushtaq,tad}@labri.fr

***Abstract*— Peer to Peer networks (P2P) consist of a set of logically connected end-clients called peers, which form an application-level overlay network on top of the physical network. P2P solution facilitates contents/files sharing among Internet users in a fully distributed fashion. This paradigm is anticipated to resolve observed limitations in current centralized solution distribution and to significantly improve their performance. P2P networks are undergoing rapid progress and inspiring numerous developments. Although initially P2P networks were designed for file sharing, but their dynamic nature makes them challenging for media applications streaming. Despite recent advances in streaming P2P multimedia system, many research challenges remain to be tackled. This paper presents a state of the art study on several solutions, which exploit the power of P2P technique to improve the current multimedia streaming protocol. Different aspects related to the topic are explored in order to point out the open research issues in the domain of Peer to Peer Multimedia Streaming. Our foremost objective in this paper is to motivate and guide the ongoing research to tackle these challenging problems and help to realize efficient streaming multimedia P2P mechanisms.**

Keywords: P2P, Video Streaming, Overlay Network architecure, Video Coding

I. INTRODUCTION

Within the next generation Internet, it is expected that the interest on multimedia services and in particular Video/Audio Streaming will grow up significantly. P2P traffic will take accordingly a non negligible amount of the global Internet exchange in the near future. Multimedia streaming over the Internet is mainly managed by Content Distribution Networks platforms (CDNs) such as Akamai [18], Limelight Networks [19] … Recall that a CDN platform is composed of a set of dedicated servers that are in charge of (1) content storing and (2) serving client demands by streaming and unicasting the requested content towards clients. Consequently, in order to achieve correct performance, CDNs must conduct an important infrastructure cost in order to avoid server bottleneck issues. Moreover, since multimedia streaming requires high bandwidth, server network bandwidth runs out rapidly using these architectures.

Another alternative may consist of using IP multicast systems for these applications. Indeed, IP Multicast is probably the most efficient solution; however its deployment remains limited due to many practical and political issues, such as the lack of incentives to install multicast-capable routers and to covey multicast traffic. Furthermore, the use of IP multicast is not adapted for some interesting cases (streaming from multiple senders for instance).

Concurrently, we observed the extreme popularity of P2P networks during last few years. They are autonomous and distributed systems that aggregate a large amount of heterogeneous nodes known as Peers. These peers incorporate with each other to accomplish some tasks/objectives. Such a system encompasses interesting characteristics like self configuration, self adaptation and self organization. P2P phenomenon offers several facilities. It allows information flow exchange from and back to end user, rapid and dynamic set up of communities sharing the same interests. The main targets of such systems are file sharing applications like Kazaa [15], eDonkey [16], BitTorrent [17]…

These intrinsic characteristics make the peer-to-peer (P2P) model a potential candidate to solve the pointed out problem in multimedia streaming over the Internet. P2P networks overcome the setback of bottleneck around centralized server due to its distributed design and architecture. Moreover, it facilitates to manage dynamically the available resources in the networks since they scale with the number of peers in the systems.

Although, P2P technology gives novel opportunities to define an efficient multimedia streaming application but at the same time, it brings a set of technical challenges and issues due to its dynamic and heterogeneous nature. Even though the problem has been already studied in the literature [11,12,13,14], works on P2P media streaming systems is still in the early stages, and for a P2P streaming to be enhanced, important research efforts and investigations are still required. Existing P2P protocols must be revised or re-invented and other specific problem need to be addressed to meet the multimedia streaming requirement.

Our objective in this article is two fold, firstly to provide a better understanding of the basic concepts of multimedia streaming over P2P networks, and secondly to identify research challenges related to this area. The rest of this article is organized as follows: P2P streaming network architecture is described in section II, a comparison for different video coding techniques in the context of P2P streaming is illustrated in section III, some existing solutions for the P2P media streaming are presented in section IV, we highlight certain issues for the domain in section V and paper is summed up by making some concluding remarks in section VI.

II. P2P NETWORK STREAMING ARCHITECTURE

The network streaming architecture refers to the manner used for multimedia content transfer and the entities which are involved during the streaming mechanism. In the context of P2P streaming, a given peer can play three different roles:

- Source: Peer containing the media contents and intended to share with other peers. Peer can store whole or a part of a given content
- Destination: It is the client who requests for the content. Client peer can obtained media contents from one or more sender peers depending on the architecture.
- Intermediate: An intermediate peer, receive a given content and then transmit it to the next intermediate peer. Intermediate peer serves as a transport node to facilitate the streaming mechanism.

The content media is distributed to the clients using generally an overlay network organized as an appropriate tree structure. The latter is rooted at the source or destination peer depending on the approach employed. Thus, we define two kinds of Network architectures

A. *Multiple sources*

Multiple source architecture is used when the multimedia contents are either replicated/existed within many source peers in the network, or it can be smartly split and dynamically placed within several peers.

Although the first case is more trivial since any content can be found in several emplacement into the network, especially if it is popular. In the second case a pre-processing phase and continuous analyses of client request is essential to place the different pieces of the content in the network to meet the client demands.

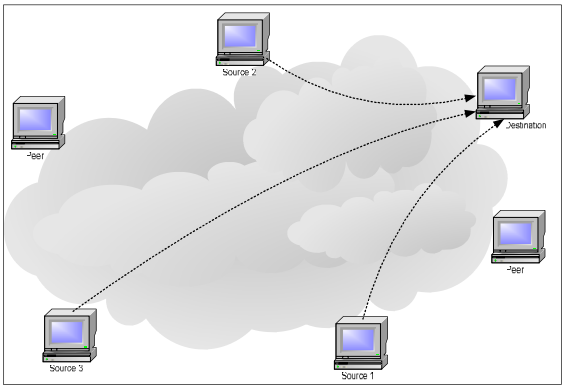


Figure 1. Multi-Source P2P streaming model

Therefore, as shown in Figure 1 contents can be retrieved from several peers into the network simultaneously. Here, each client peer receives packets of a multimedia content from multiple sender peers (peer having contents) while each sender peer can send packets to one or multiple client peers. The role of intermediate peer is limited to the transfer of the received packet towards the destination peer. Intermediate peers are not shown in the figure.

B. *Single source*

In this case, the multimedia content is stored into only one source peer in the network. The content peer starts transmitting the content to all client peers requesting for it. In this case, the intermediate nodes can play a more important role. To be significantly efficient, the intermediate nodes store some part of the content in their internal buffer. When a new client peer joins the network, it can directly retrieve the requested content from a given intermediate node. Hence, the overload around the source can be distributed over the entire network.

Figure 2 gives an example of single source streaming towards two peer clients.

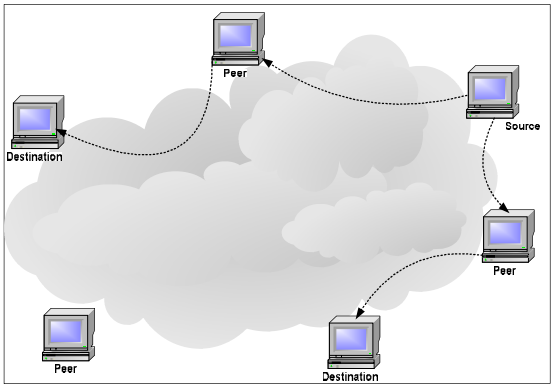


Figure 2. Single-Source P2P streaming model

In literature there exists other architecture for media streaming over P2P networks as well, like Central Server Based. CoopNet [11] solution is based on this central server model but here, we are concentrating on pure P2P architecture.

III. VIDEO CODING

In the following section, we explore video coding techniques used for video transportation over the IP network, and accordingly in P2P networks. Understanding of these techniques and related challenges to the choice of adequate techniques will help us to optimize the resource usage and to improve substantially the overall video quality.

Packet loss and error propagation that occur frequently over current IP networks can dramatically reduce the video quality at the receiver end. Hence, error resilience and handling packet loss are critical issues in the streaming applications. Several coding solutions have been developed to tackle these issues, to enhance the overall quality and to protect multimedia traffic against severe network conditions. There are mainly two major techniques for video encoding, Multiple description coding (MDC) and Layered Coding (LC) [4], which are analogous techniques

MDC and LC are useful in the case of varying bandwidth/throughput and losses/erasures due to congestion (as for Internet) and uncorrectable errors (as for wireless channels). Layered coding provides a scalable representation that enhances rate control but it is sensitive to transmission losses. On the other hand, multiple description coding provides increased resilience to packet losses by creating multiple streams that can be decoded independently.

In both these schemes, the multimedia stream is split into many descriptions/layers where each description/layer can contribute to the definition of one or more characteristics of multimedia data. The difference between MDC and LC lies in the dependency among description/layer. In the case of LC, layers are referred to as "base layer" and "enhancement layers". The base layer is one of the most important layers while the enhancement layers are referenced to the base layer. Enhancement layers can not be decodable independent to base layer. In contrast to Layered coding, in MDC each description can be decoded individually to get the base quality. However, with more descriptions being acquiring and decoded, the video distortion can be lowered and the larger is the output signal quality. MDC/LC eases the management of variable bandwidth/throughput by transmitting a suitable number of descriptions/layers.

MDC greatly improves loss/erasure resilience because each bitstream can be decoded independently, making it unlikely to have the same portion of data corrupted in every description. LC can improve error resilience when the protection level for a given layer can be adapted to its importance so that the base layer is more protected.

Other techniques have been proposed to enhance error/loss resilience of multimedia streams sent through unreliable channels. Among these, there are techniques like forward error detection/correction codes (FEC) or automatic repetition requests (ARQ). ARQ is very effective but it requires a feedback channel and it can be used only in point-to-point communications, not for broadcast. Of course, time must be allowed for retransmissions. On the opposite, FEC does not require a feedback channel and it is suitable for broadcast. Both these techniques, ARQ and/or FEC, can be used together with MDC/LC.

In the following section, we presented some related works that attempt to study the impact of video coding strategies on the streaming over P2P networks.

In [1], authors investigate streaming layered encoded video using peers. Each video is encoded into hierarchical layers which are stored on different peers. The system serves a client request by streaming multiple layers of the requested video from separate peers. The system provides an unequal error protection for different layers by varying the number of copies stored for each layer according to its importance. A comparison is made for the performance of layered coding with multiple description coding. The obtained results showed that layered coding is a better choice when the system can find and switch over to another server peer quickly, while MD-FEC performs better if the replacement time is non-negligible.

In [2], authors study the performance of multiple description coding and of layered coding for video streaming over the Internet. A comparison is conducted using different transmission schemes. Scenarios where transmission over multiple network paths is employed are also considered. The obtained results show that the relative performance of the two techniques varies substantially depending on the transmission scenario under consideration. It is seen that layered coding outperforms multiple description coding when rate-distortion optimized scheduling of the packet transmission is employed. The converse is observed for scenarios where the packet schedules are oblivious to the importance of the individual packets and their interdependencies.

In [3], authors examine both MDC and LC coding, distribution and substream placement in the network. For both schemes a traffic theory is developed. Authors formulate and solve the optimal solution of the problem of finding the optimal number of descriptions and their rates. The optimal number of sub-stream replicas for each video. A simple mechanism for placing the replicas in the server, for selecting servers, and for admission control is proposed as well.

For both MD and LC, the video quality improves as the peer "connect probability" increases. When the peer "connect probability" is small, the performance of the MD system is much better than the layered system. As peer connect probability increases, the performance of the layered system increases at a rate faster than the MD system. With zero replacement time, as the peer "connect probability" increases beyond a certain point, the layered system outperforms the MD system.

When the network is more reliable, LC is more efficient than MD. However, when the replacement time increases, the performance of the MD system is always better than the layered system. Therefore, the time to find a replacement peer has a bigger impact on the layered system than the MD system. The reason is that MD-FEC has inherent protection against sub-stream loss. When a single sub-stream is lost for MD-FEC, the video quality is only slightly affected. But for layered coding, all layers higher than this sub-stream cannot be decoded independent to the base layer at the receiver end.

IV. STREAMING OVER P2P NETWORK

In this section, we examine some works attempting to define efficient solutions for the multimedia streaming over P2P network. We presented the limitations for the proposed solutions as well.

Heffeda et al propse PROMISE [4]. The system realizes several optimizations so that the receiver will observe minimum fluctuation of media streaming quality. PROMISE encompasses the following functionalities (1) selecting the best sending peers, (2) monitoring the characteristics of the underlying network, (3) assigning streaming rates and data segments to the sending peers, and (4) dynamic switching of sending peers.

Three approaches are proposed for the selection of the best peers: (1) Random selection of peers that can fulfill the aggregate rate requirements. (2) E2E selection which estimates the “goodness” of the overlay path between each candidate peer and the receiver. (3) Topology aware selection which infers the underlying topology and its characteristics and considers the goodness of each segment of the path. Evaluation conducted formally and by simulation shows that the topology aware selection enable a judicious selection by avoiding peers whose paths are sharing a tight segment.

In [5] authors studied the construction of an efficient overlay P2P for multimedia streaming to handle network dynamicity and selfish user behavior.

The peer clients form overlays to forward the video streams over peer-to-peer. A given client, who joins the network, has to select a parent node that has sufficient bandwidth to itself. The selection mechanism of the parent node should allow admitting as many clients as possible in the long run. Assume that client sends a service request to the directory server and the directory server returns a list of candidates that can provide the service. The QoS parent selection algorithm uses the distance-bandwidth ratio as the metric in selecting the parent peer client.

ZIGZAG [6] deals with the problem of one source towards multiple destinations with consideration of network condition. The objectives are to minimize the E2E delay, to manage user dynamicity and to keep the overhead traffic as small as possible to achieve scalability

To realize this objective, ZIGZAG organizes receivers into a hierarchy of bounded-size clusters and builds the multicast tree based on that. The connectivity of this tree is enforced by a set of rules, which guarantees that the tree always has a height O($\log_k N$) and a node degree O(k^2), where N is the number of receivers and k a constant. The proposed approach helps in reducing the number of processing hops processing to avoid the network bottleneck.

In [8] DONet is presented, a Data-driven Overlay Network for live media streaming. The core operations in DONet are very simple and do not need any kind of complex tree structure for data transmission. Actually, every node periodically exchanges data availability information with a set of partners, and retrieves unavailable data from one or more partners, or supplies available data to partners. Authors show through analysis that DONet is scalable with bounded delay and also address a set of practical challenges for realizing DONet. An efficient member and Gossip based partnership management algorithm is proposed, together with an intelligent scheduling algorithm that achieves real-time and continuous distribution of streaming contents. Furthermore, mechanism for node failure and system recovery are also investigated

P2VoD [9] takes advantage of intermediate peers that forward the multimedia content by caching the most recent content of the video stream it receives. Existing clients in P2VoD can forward the video stream to a new client as long as they have enough out-bound bandwidth and still hold the first block of the video file in the buffer. A caching scheme is used to allow a group of clients, arriving to the system at different times, to store the same video content in the prefix of their buffers. An efficient control protocol to facilitate the manage join and failure recovery processes based on multicast tree is also proposed.

GnuStream [10] is built on the top of Gnutella. it is designed to takes into consideration the underlying P2P network dynamics and its heterogeneity. It handle bandwidth aggregation, adaptive buffer control, peer failure or degradation detection and streaming quality maintenance. The changes for peer status are detected using periodic probing. The Recovery from failure or degradation is handled by selecting the best peer.

CoopNet [11] is a solution for distributing streaming media content using cooperative networking. CoopNet solution is based on central server model. It provides resilience by introducing redundancy both in network paths via multiple, diverse distribution trees and in data using MDC. A centralized tree management protocol is used to construct short and diverse trees and support quick joins and leaves. Moreover, a scalable feedback mechanism is used to drive an adaptive MDC optimization algorithm. The tree efficiency is ensured by mapping between logical and physical topology.

Our proposed quality adaptive streaming mechanism [12] is based on the End-to-End “RTT” estimation among the receiver peer and sender peers. Active monitoring is performed to analyze the new network conditions and in peer switching is performed in the case of superfluous changes occurred on the network. To avoid the overhead we proposed End-to-End “RTT” estimation i.e. “RTT” among receiver and sender peers, for P2P streaming mechanism. We proposed to construct overlay networks for the sending peers based on the “RTT” and video quality offered by each peer. We used Object Classification Model for the MPEG-4 video by classifying the Audio and Video objects having certain priorities. Furthermore, layered coding is proposed for data encoding where original video is decomposed into different layers (Base Layers and Enhanced Layers) where Base Layer is most important.

In [13] A Hybrid Overlay Network protocol for on-demand media streaming is proposed. The overlays are maintained to ensure data transmission, called a "tree overlay" and a "gossip overlay". As named, the tree overlay is based on a tree structure rooted to the source. The gossip overlay is a random graph that uses random dissemination mechanism. Most data segments are delivered through the gossip overlay; only if a node fails to receive a data segment till certain deadline, will it resort to the tree overlay to fetch the segment from its parent. Compared to the tree structure, the random dissemination in gossip exploits the available bandwidth from all the potential network paths and also enhances robustness in the presence of bandwidth oscillations or malfunctions of internal tree nodes.

Anysee [14] is a peer-to-peer live streaming system tailored to fit cases where of multiple overlays is considered. Anysee adopts an inter-overlay optimization strategy by constructing and maintaining efficient paths using peers in different overlays.

Several optimizations are introduced: (1) the use of location mechanism for overlay construction to map underlay and overlay topology where constructing a given overlay network and its logical connection (2) the selection of an overlay manager by overlay network to manage peers join and leave. (3) On each node an Inter-overlay optimization manager is in charge to maintain one active path and backup path set. (4) Key node manager which enforce an adaptive admission control mechanism by introducing several queues. Actually received requests are transferred to the appropriate internal queue according to its priority. (5) Buffer manager which is responsible for receiving valid media data from multiple providers in the active streaming path set and continuously keeping the media playback.

Reza et al. have proposed the PALS framework [20] for P2P adaptive layered streaming. It is a receiver centric framework, where a receiver coordinates delivery of layer encoded stream from multiple senders. In this framework initial peers are selected on random basis because there is no information available in the start of streaming mechanism. After this initial stage, peer selection is performed by an iterative process. Each time a new peer is admitted and kept as sender peer only, if it enhances the overall throughput otherwise it is dropped. For the quality adaptation (QA), receiver manages its buffer regularly on the basis of packets consumption and sends the buffer state to each sender regularly. The QA mechanism for PALS determines inter-layer bandwidth allocation for a period of time rather than on a per-packet basis.

V. ISSUES IN MULTIMEDIA P2P STREAMING

The distinct features of P2P streaming systems bring many challenging issues. Despite the availability of many solutions, P2P streaming is still an active research area with many challenging problems to be addressed. We believe that an efficient solution must capture the following features:

A. Appropriate video coding scheme

The prone-error nature of multimedia content makes it highly sensible to the transmission over networks offering non-guaranteed transmission. Therefore, a reliable multimedia transmission system must involve a reliable video coding scheme. The use of an appropriate video coding scheme is more then essential, such a scheme must be sufficient flexible to meet the P2P network dynamics and its heterogeneity.

B. Managing Peer dynamicitiy

Since the peers (network nodes) are end-users terminal, their behaviour remains unpredictable. Due to dynamic nature of P2P networks, they are free to join and leave the service at any time without making any prior notification to other nodes. Thus, dynamicity management is crucial for the smooth play back rate during streaming session.

To prevent service interruption due to peer entrance\leaving, we need a robust and adaptive mechanism to manage such changes. The proposed mechanism must incorporate recovery phase gracefully to tackle the sudden changes occurred in the network. When a sender peer leaves the system, it must detect as early as possible and replaced with another sender peer to perform streaming in a smooth fashion.

C. Peer heterogeneity

Peers are heterogeneous in their capabilities. At network level, this heterogeneity may be caused either by different access networks connecting the peers, or by difference in the willingness of the peers to contribute. Each sender peers can have a different available bandwidth and that too might fluctuate after the connection is established. Peers Selection mechanism must be capable to tackle such heterogeneity problems as well.

D. Efficient overlay network Construction

The objective is to organize participating peers into a logical topology that must infer the underlying topology. In fact, a non suitable overlay topology can result in extra overhead and can reduce the system performance drastically. The overlay construction should be scalable.

E. Selection of the best peers

An efficient and flexible strategy must be introduced for the selection of sender peers and intermediate peer. In fact, another feature that must be captured in a streaming multimedia system is minimizing end to end delay performance metric where keeping the global overhead reasonable. In fact, the less this delay is, the more live the multimedia content is.

Since the multimedia content may have to go through a number of intermediate nodes, this will increase the E2E delay. The latter may also be long due to an occurrence of bottleneck at the source node. In both cases the routing protocol must select suitable strategy that enable the selection of the best peers minimizing the global E2E delay. Another important point is how to deal with underlay network optimization objectives while trying to satisfy P2P streaming overlay constraints.

Intelligent selections criteria need to be proposed to minimize the E2E delay by keeping in mind the number of intermediate node to be traversed and different routing policies.

On the other hand, for efficient use of network resources, the global control overhead introduced for network topology management should kept as small as possible. This is important to the scalability of a system with a large number of receivers.

F. Monitoring of network conditions

The network condition during streaming phase can be changed dramatically due to the dynamic nature of P2P architecture. So, along with the dynamicity management, it is important to monitor the current network conditions regularly. The available resources (bandwidth) can vary during streaming phase due to change in resource sharing by peers present in the network or due to arrival or removal of peers. The monitoring of current network conditions is necessary to maximize the utilization of available resources and to minimize the packet drop ratios at certain links.

G. Incentives for participating peers

In many studies, it is found that many peers join the P2P network to benefit from share other's resources (more often data content) but they never share their own resources (bandwidth). It was reported that in 2000, 70% of Gnutella users shared no file but they only download contents [21]. In the presence of this issue, when no one is ready to share its bandwidth but wants to get share from other's bandwidth, P2P network starts behaving like client-server architecture and it fails due to increasing number of client peers.

In future studies, we should tackle this issue as well. The issue can be resolve by offering some incentives to peers participating in streaming mechanism. The incentives can be incorporated using some economical modeling etc.

VI. CONCLUSION

Even we assist to a proliferation of P2P streaming solution over IP network; work in this area is still in the earlier stages.

In this paper, we have presented a comprehensive state of the art and related issues. Towards this end, we first exposed recent research work providing preliminary results related to the problems of P2P streaming over IP network. Moreover, we highlighted open challenges in the design of reliable solution that overcome the limitation of current approaches.

In our future works, we plan to investigate the above mentioned issues. Our major concern is to improve the overall received quality of multimedia streaming by leveraging the underlying network topology within the overlay P2P network and to model their interaction.

VII. REFERENCES

[1] Y. Shen, Z. Liu, S. P. Panwar, K. W. Ross and Y. Wang, "Streaming Layered Encoded Video using Peers", IEEE International Conference on Multimedia and Expo (ICME), July, 2005.

[2] J. Chakareski, S. Han, and B. Girod, "Layered Coding vs. Multiple Descriptions for Video Streaming over Multiple Paths," Multimedia Systems, Springer, online journal publication: Digital Object Identifier (DOI) 10.1007/s00530-004-0162-3, January 2005

[3] Y. Shen, Z. Liu, S. Panwar, K.W. Ross, and Y. Wang, "Peer-Driven Video Streaming: Multiple Descriptions versus Layering" submitted.

[4] Andrea Vitali, Marco Fumagalli, "Standard-compatible Multiple-Description Coding (MDC) and Layered Coding (LC) of Audio/Video Streams", Internet Draft - Network Working Group. July 2005. http://ftp6.us.freebsd.org/pub/rfc/internet-drafts/draft-vitali-ietf-avt-mdc-lc-00.txt

[5] M. Hefeeda, A. Habib, B. Botev, D. Xu, B. Bhargava, PROMISE: Peer-to-Peer Media Streaming Using CollectCast, In Proc. of ACM Multimedia 2003, pages 45--54, Berkeley, CA, November 2003.

[6] Duc A. Tran, Kien A. Hua, Tai T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming". In Proceedings of IEEE INFOCOM 2003, March 30-April 3, San Francisco, CA, USA

[7] S. Itaya, T. Enokido, M. Takizawa, A. Yamada, "A scalable multimedia streaming model based-on multi-source streaming concept", Proceedings of 11th International Conference on Parallel and Distributed Systems, 2005.

[8] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: A Data-driven Overlay Network for Live Media Streaming", IEEE INFOCOM'05, Miami, FL, USA, March 2005.

[9] T.Do, KA. Hua, M. Tantaoui, "P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment" to appear in the Proc. of the IEEE International Conference on Communications (ICC 2004), June 20-24 2004, Paris, France.

[10] X. Jiang, Y. Dong, D. Xu, B. Bhargava, "GnuStream: a P2P Media Streaming System Prototype" IEEE International Conference on Multimedia and Expo Baltimore, MD, July 2003.

[11] V. N. Padmanabhan, H. J. Wang, P. A. Chou, "Resilient Peer-to-Peer Streaming", IEEE ICNP 2003, Atlanta, GA, USA November 2003.

[12] M. Mushtaq, T. Ahmed, D.E Meddour, " Adaptive packet video Streaming Over P2P NETWORKS", To appear

[13] M. Zhou and J. Liu, "A Hybrid Overlay Network for Video-on-Demand, IEEE International Conference on Communications (ICC'05), Seoul, Korea, May 2005

[14] Xiaofei Liao, Hai Jin, Yunhao Liu, Lionel M Ni, and Dafu Deng, "AnySee: Peer-to-Peer Live Streaming", To appear at IEEE INFOCOM 2006, Barcelona, Spain, April 2006.

[15] Kazaa http://www.kazaa.com.

[16] eDonkey http://www.edonkey.com.

[17] Bittorrent http://www.bittorrent.com/

[18] Akamai http://www.akamai.com

[19] Limelight Networks http://www.limelightnetworks.com.

[20] Reza Rejaie, Antonio Ortega, " PALS: Peer-to-Peer Adaptive Layered Streaming" in Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video, Monterey, California, June 2003

[21] Eytan Adar and Bernardo Huberman, "Free Riding on Gnutella, First Monday". October 2000 available at http://www.firstmonday.dk/issue5_10/adar/

Author Index

NOTES

NOTES

NOTES

NOTES

NOTES