# On Revealing the ARQ Mechanism of MSTV

*Oliver Hohlfeld, *Balamuhunthan Balarajah, †Sebastian Benner, *Alexander Raake, *Florin Ciucu
*Technische Universität Berlin / Deutsche Telekom Labs †T-Systems
{oliver, bbala, florin}@net.t-labs.tu-berlin.de
{alexander.raake}@telekom.de {sebastian.benner}@t-systems.com

***Abstract*—Ensuring a high customer satisfaction by monitoring Quality of Experience (QoE) aspects has become common practice for service providers. Such monitoring solutions, together with underlying QoE models, are mostly limited to measures captured in the core or access network and may thus neglect the QoE impact of recovery mechanisms deployed at client-side, e.g., FEC or ARQ. This limitation makes QoE models prone to mispredict QoE and consequently may lead the operators to misleading interpretations of customer experience. In this paper, we empirically study the behavior of the Microsoft TV Set-top Box (STB) with respect to the deployed ARQ recovery mechanism. As the ARQ implementation details of the STB are proprietary, we implement and simulate three ARQ algorithms of different complexities and evaluate their performance by comparing with corresponding empirical measurements. This comparison reveals insights into the ARQ scheme implemented in the STB. Moreover, it leads us to speculate that MSTV uses simple ARQ schemes which are sufficient to drastically improve the QoE in the presence of a multitude of loss patterns.**

## I. Introduction

The provisioning of broadband access has enabled the deployment of new services such as the distribution of TV content over IP networks (IPTV) or Video on Demand platforms. For the successful deployment of such services, it has become increasingly important for service providers/operators to understand and control Quality of Experience (QoE) aspects. In a broad sense, QoE refers to the service quality perceived by customers and can be assessed in terms of objective and subjective psychological measures [1].

In the case of IPTV, controlling desirable QoE levels is a complex problem as it involves as diverse aspects as the characteristics of network traffic, video content, codec properties and the actual subjective perception of video quality. The common solution adopted by service providers consists in the real-time monitoring of QoE. Observations of drops in the QoE levels demands for network control to improve the service delivery and ultimately the customers' satisfaction.

For running in real-time, monitoring solutions face the critical requirement of relying on QoE models of low to moderate computational complexity. This requirement is met by parametric QoE models [2], [3], [4], [5], [6], [7] which rely on easily measurable QoS and content parameters (e.g., packet loss, jitter, bitrate) and simple heuristics for QoE computation. However, QoE parametric models are mostly limited to measures captured in the core or access network and may thus neglect the QoE impact of recovery mechanisms deployed at client-side. This includes recovery mechanisms that are usually deployed on various layers in the protocol stack. For instance, on the application layer, visual concealment techniques are commonly used for video loss recovery. Without considering for such recovery mechanisms, QoE models are prone to mispredict QoE and consequently may lead to suboptimal network control by the operators. In order to prevent such misguided reactions, it becomes critical for the operators to understand the influence of recovery schemes on QoE models.

In the literature, recovery mechanisms are often categorized into *active* and *passive*. Passive recovery techniques can be implemented at either the transport or application layer by using FEC algorithms to embed redundant information in the bitstream and therefore allow for bit-exact error recovery without any interaction between sender and receiver [8], [9]. Such characteristics make passive recovery appropriate for delay critical interactive real-time applications, e.g., VoIP or video conferencing. In turn, active recovery techniques are characterized by the interaction between the sender and receiver in the form of retransmission requests of lost information. Moreover, they induce much lower network load and processing overhead at low error rate and are thus preferred for error control design. In order to address the problem of underestimating QoE due to neglecting recovery mechanisms, we study in particular the active recovery mechanism that is implemented in the Microsoft TV (MSTV) solution used by several big ISPs (e.g., Deutsche Telekom, AT&T) for their IPTV systems. MSTV implements active recovery at client-side in the Set-Top Box connected to the TV set. In order to circumvent the unavailability of ARQ implementation details of the STB, which are proprietary, we implement three ARQ algorithms of different complexities and evaluate their efficiency in recovering errors and the amount of generated overhead. Such factors are particularly important for the ISP as they contribute to the achievable QoE. Based on empirical STB observations and further comparisons with simulation results of our ARQ algorithms under realistic network conditions, we are able to provide insights, into the ARQ scheme implemented in a widely used IPTV system and its impact on QoE. Moreover, the observed results lead us to speculate that MSTV uses simple ARQ schemes which are sufficient to drastically improve the QoE.

The remainder of this paper is structured as follows. Section II describes the measurement setup used for empirically evaluating the MSTV STB. Section III introduces the de-jitter buffer model and the used ARQ schemes implemented in our simulator. Properties of the ARQ scheme implemented in the MSTV STB as well as the comparison between empirical observations and simulation results are further discussed in

Section IV. Section V summarizes related work on the performance of active and passive recovery, and finally Section VI presents brief conclusions.

## II. Measurement Setup

In order to study the behavior of the ARQ scheme implemented in the MSTV STB, we use an edge-based measurement setup as depicted in Figure 1. The STB joins a multicast group to receive a TV channel at a constant bitrate of 3 Mbps. This scenario resembles a home setting with an STB connected to an ADSL line subscribed to the IPTV service. The considered scenario accounts for congestion in the access network where multicast and ARQ traffic is impaired by the same loss process. However, due to the considered topology, it does not account for different loss processes between ARQ and multicast traffic in backbone networks which may arise, e.g., in the presence of different routes. The study of this setting is interesting as errors are known to mostly occur in access networks which form a bottleneck and rarely in the (mostly) overprovisioned and/or QoS enabled backbone network.

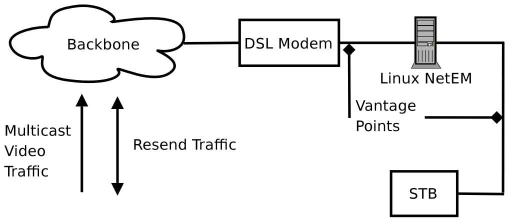


Fig. 1: Measurement Setup

To make our setup and subsequent analysis more realistic, we account for different network conditions by injecting uniform and bursty packet losses with and without jitter. To this end we use a Linux machine running the Network Emulator (Netem) functionality. Due to limitations of Netem in supporting the Weibull distribution, the injected jitter is generated instead according to the normal distribution.

The measurement setup uses a capture process to automatically record a large set of traffic traces with and without impairments over multiple days. This is accomplished by placing the vantage points before and after Netem. The capturing process works as follows. Once the network impairment is configured in the emulator we wait for 20 seconds to allow the STB to stabilize and then start the capturing process for 130 seconds. Upon the completion of the capture process we reset the loss and jitter settings in the emulator to provide the STB with an additional 20 seconds of unimpaired traffic before capturing the next network impairment configuration. The obtained traffic traces enable the empirical analysis and sets the basis for simulating the STB behavior.

## III. Simulation Design and Buffer Models

We designed a discrete event simulator to evaluate the behavior of the setup in Figure 1. The simulator only uses the multicast traffic contained in the captured traces by isolating the additional ARQ traffic. This isolation replicates the network conditions faced by the STB, including the packet loss and jitter processes. In this way we are able to appropriately speculate about the ARQ design used in the STB.

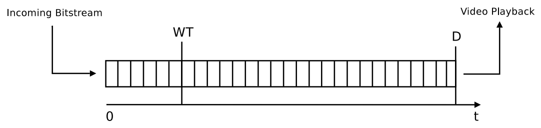


Fig. 2: De-Jitter Buffer Model

### A. De-Jitter/Playout Buffer Model

An important component of our simulator is the de-jitter/playout buffer. This component is generally featured by multimedia applications in the Internet where the packet delay is known to fluctuate rapidly (jitter) [10] due to processing and queueing delays. In order to compensate for such varying network delay, multimedia applications generally deploy de-jitter buffers that delay the playout of a packet by buffering at the client-side. In general, such buffers can be implemented in a static or dynamic manner [11], [12], depending on whether the buffer depth is fixed or dynamically adapted during streaming.

De-jitter/playout buffers are especially necessary in the presence of ARQ mechanisms in order to wait for the generated retransmissions to arrive before their scheduled playout time. The time needed for the waiting process depends not only on the conditions of the network (RTT and jitter), but also on the processing delay of the server hosting the retransmission cache and handling the retransmissions. Due to such factors, the dimensioning of buffer is generally a challenging problem.

For the sake of simplicity, we consider the static de-jitter buffer illustrated in Figure 2. The buffer holds $D$ ms of incoming traffic and serves the video decoder a constant rate bitstream of packets that are free of jitter. The playout of each packet is delayed for $D$ ms due to buffering. Packets which never arrive by their scheduled playout time are considered to be lost and discarded. Note that for our simulation settings we use a value of $D = 1000$ ms for the buffer depth, suggested by observations in the measurement results.

### B. Implemented ARQ Schemes

In addition to the de-jitter buffer, our simulator implements three ARQ schemes of different complexities. The first two mechanisms are motivated by how the Real-Time Control Protocol (RTCP) [13] defines feedback in RTP streaming systems. The third scheme is motivated by the STB behavior observed in measurements.

Each ARQ scheme triggers a packet loss detection after waiting for $WT$ time, in order to allow for out-of-order packets to arrive before the loss detection. Estimates of $WT$ and the mechanism used by MSTV are provided in Section IV-C. In future work, we plan to study the optimal range of $WT$.

We assume that the deployed transport protocol implements sequence numbers such that any missing packets can be easily detected by gaps in the received packet sequence. This is realized by the Real-time Transport Protocol (RTP) that is used for streaming in the studied IPTV system. Packets that are missing at time $WT$—relative to the beginning of the buffer—are considered to be lost and are subsequently requested by the ARQ protocol. In order to increase fault tolerance, all three mechanisms issue two identical copies of the same request packet spaced randomly 10 to 30 ms from each other, as also observed from the STB measurements.

In the following we give a more detailed description of the implemented ARQ schemes starting with the simplest one.

*1) Mechanism 1:* In this scheme a resend-request is generated for each lost packet. As mentioned earlier, this elementary approach is motivated by the immediate feedback mode discussed in [13], but is however restricted to allowing for a single request per packet only.

Setting $WT = 0$ results in the situation in which the lost packets are immediately requested when the next packet is enqueued in the buffer. An undesirable consequence of this extreme setting is that it triggers an increase of the network load and of the retransmission cache server in case of out-of-order packets. Setting higher waiting times $WT$, within the buffer depth $D$, allows out-of-order packets to arrive before requests are issued.

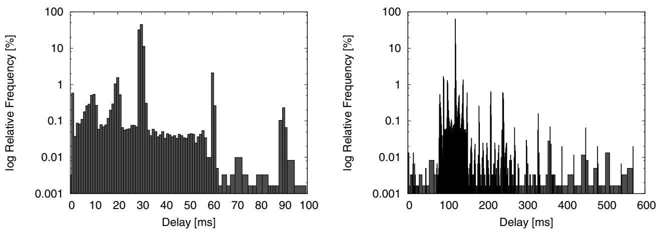
We point out that, in general, $WT < D - \text{RTT} - \sigma_{\text{Jitter}}$ must hold in order to allow retransmissions to arrive before playout (here, RTT denotes the mean delay to the resend-server and $\sigma_{\text{Jitter}}$ denotes the standard-deviation of the delay variability).

*2) Mechanism 2:* To avoid the inherent increase in network load characteristic to *Mechanism 1*, we now allow for requests of multiple video packets in a single compound request. This is implemented in the resulting *Mechanism 2* by a timer that is started at the first packet loss and expires after $WT$ time. All packet losses that occur during this time period are requested in a batch. After the resend request is sent, the timer is stopped and is started again upon the next packet loss event.

*3) Mechanism 3:* This mechanism allows for the periodic generation of resend requests as observed in the STB measurements. Concretely, such a request is issued every $WT$ time, iff $n \geq 1$ packet losses occur within $WT$ time. This is implemented by modifying the timer introduced in *Mechanism 2* to start when the first packet enters the buffer and to expire after $WT$ time. The timer restarts irrespectively of whether a request was issued or not.

## IV. Evaluation

In this section we discuss the empirical measurements and simulations of the ARQ schemes. Our main objective is to gain insights into the properties of the ARQ scheme implemented by the MSTV STB. We start by first analyzing the observed STB behavior. Then we proceed to compare the measured STB performance and corresponding simulations with respect to correction efficiency and induced network load. The observed STB correction efficiency is further analyzed in terms of achievable QoE improvements.



(a) Delay between identical copies (b) Delay between different requests

Fig. 3: Delay between requests for for the uniform loss dataset

### A. Set-top Box Behavior

Our study focuses on STB service degradations which can provide direct insight into the details of the implemented ARQ scheme. We particularly evaluate the statistical properties of the request and retransmission packets from the capture traces, and which are next discussed.

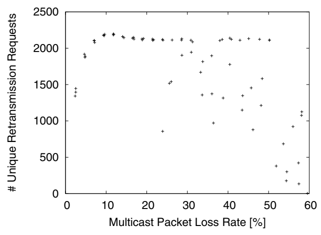
*1) Request Packets:* We start our analysis by investigating properties of retransmission requests issued by the STB. For those requests we observed that the STB sends two identical packets. As pointed out earlier, this approach of requests duplication increases the fault tolerance by increasing the probability of retransmission success.

We also evaluated the delay between two identical copies of the same request and between two non-identical requests. The corresponding distributions are shown in Figure 3.(a) and (b). In the case of identical copies, the distribution shows peaks at $\{1, 10, 20, 30, 60, 90\}$ ms, where 30 ms was found to be the dominant peak in all of our measurements (see Figure 3.(a)). In turn, in the case of non-identical requests, the delay distribution shows a dominant peak at 120 ms (see Figure 3.(b)).

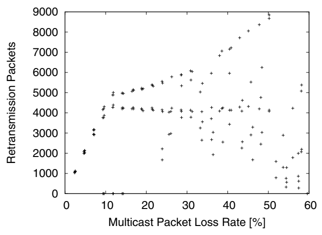
In addition to the delay distributions, we observed a rate of no more than 10 non-identical requests per second issued by the STB per second, as shown by the upper bounded traffic in Figure 4.(a). This indicates the presence of periodic checks in the STB which motivated our Mechanism 3.

Moreover, in terms of packet sizes, the analysis reveals that only 4 distinct sizes are used, i.e., 151, 154, 157, 160 bytes for the UDP datagram length. By increasing the loss rate we observed an interesting tendency for higher packet sizes. The observed packet sizes suggest the existence of multiple slots used for requesting packets; recall that the elementary *Mechanism 1* uses a single slot. To better understand whether slots are used in STB for single or multiple requests we analyzed traces with periods of subsequent losses. As no trend of increasing packet sizes was observed, as opposed to the case of isolated losses, we conclude that a slot is used by STB for requesting loss bursts consisting of subsequent packets.

*2) Retransmission Packets:* Unlike the original RTP packets which are sent to the STB by multicast, the corresponding retransmission packets are sent by unicast. This implementation detail permits the identification of the loss packets as the format used for requests is not known.



(a) Requests



(b) Retransmissions

Fig. 4: Number of request and retransmission packets for the uniform loss dataset with traces of length 130 sec

The total number of retransmissions per packet loss rate from our measurements is shown in Figure 4.(b). The figure indicates the presence of two trends. The first trend shows a saturation for isolated losses at 4*10*2 packets per second at loss rates $> 10\%$. In turn, the second trend shows a steady increase in the number of retransmissions for loss bursts. Unlike the case of retransmissions, the amount of request packets is always upper bounded. This observation supports our previous speculation on the usage of slots in request packets by the STB.

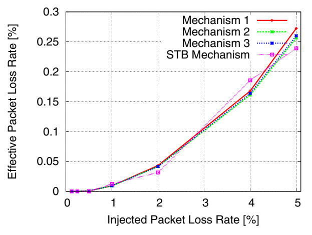
## B. Comparing Correction Efficiency

Correction efficiency of the deployed ARQ mechanisms is particularly important for the achievable QoE at the client side: the more losses can be corrected before playout the higher the resulting visual quality. To understand the correction efficiency, in the particular case of the STB, we evaluate it for several broad network conditions with multiple (WT, packet loss) combinations under uniform and bursty packet loss with and without jitter. For every (WT, packet loss) combination we run 1000 simulations and show the average correction efficiency. The results are compared to the measured STB behavior which is constant for every (WT, packet loss) combination.

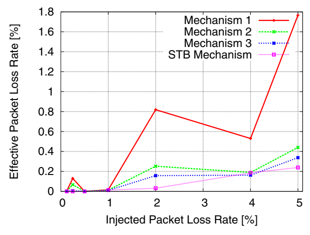
Figure 5 shows the results for uniform packet loss without jitter under unlimited slots for *Mechanisms* 2 and 3. For small values of $WT$, the buffer provides sufficient buffering to allow all retransmissions to be completed. Thus, both the simulated algorithms and the STB implementation achieve complete loss correction except for lost requests and retransmissions. We also observed (figure not shown here) that by limiting the number of slots to 4, *Mechanisms* 2 and 3 perform slightly weaker than for the default case of unlimited slots.

In contrast the case of small values of $WT$, the observed correction efficiency optimal behavior decays at larger values of $WT$ which are closer to the chosen buffer depth of one second. The reason is that retransmissions can arrive after the scheduled playout and are thus discarded.

In the case of bursty packet loss and jitter we observed similar correction efficiency behavior (figures not shown here). As a direct consequence for accounting for jitter, small delay variations (e.g., mean = 100 ms) do not influence the correction efficiency at small values of $WT$.

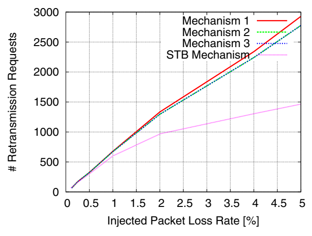


(a) $WT = 115ms$

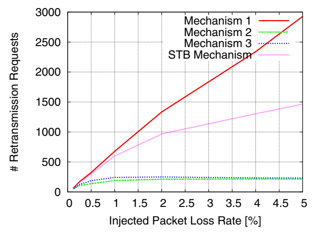


(b) $WT = 900ms$

Fig. 5: Correction efficiency for uniform packet loss



(a) $WT = 0ms$



(b) $WT = 900ms$

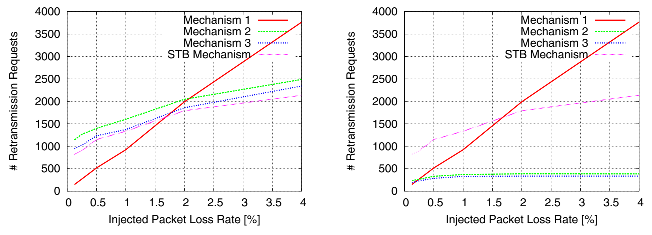
Fig. 6: Network and server load for uniform packet loss

## C. Comparing Induced Network and Server Load

In addition to providing desirable QoE to its customers, service providers are also considered about the influence of the deployed mechanisms on the network and server load. In this section we express and evaluate this load by the amount of request packets generated by the different ARQ mechanisms and the STB, under the presence of both loss and jitter.

Figure 6 shows the generated retransmission traffic when the video transmission is impaired by uniform packet loss. In the extreme case of no waiting time is specified ($WT = 0$, see (a)), all considered mechanisms generate significantly more load than the STB when the packet loss is greater than $.5\%$. Moreover, *Mechanisms* 2 and 3 barely show any improvement in decreasing load over *Mechanism* 1, because of infrequent grouping multiple requests in one packet in the particular case of $WT = 0$. The visible slight improvement can be explained by the existence of loss bursts. In another extreme case of high $WT$, both *Mechanisms* 2 and 3 generate significantly less load than the STB, whereas *Mechanism* 1 does not change its behavior from the case of $WT = 0$. In contrast to these differences, we observed however that *Mechanism* 3 with an unlimited number of slots performs similarly as the STB when $WT$ is in the order of 100 ms (figure not shown here). These observations lead us to speculate that the STB implements an ARQ scheme similar to *Mechanism* 3 with $WT \approx 100$ ms.

When jitter is also present, in addition to loss, we further observed that the load increases at small values of $WT$ (see Figure 7) for all *Mechanisms* and the STB. The explanation stems from the request of out-of-order packets. However, at higher values of $WT$, only *Mechanism* 1 and STB experience a load increase. This indicate that the STB behavior is much sensitive to the presence of jitter than loss.



(a) $WT = 115ms$ (b) $WT = 900ms$

Fig. 7: Network and server load impact for uniform packet loss and normally distributed jitter with a mean=10ms

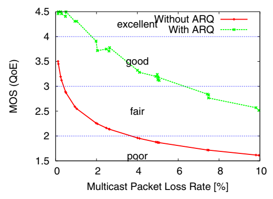


Fig. 8: QoE with and without MSTV ARQ

### D. QoE Impact

To analyze the impact of ARQ on QoE as predicted by parametric QoE models, we applied a recently proposed model [2] to the measured correction efficiency of the ARQ scheme implemented by the MSTV STB. Concretely, the QoE model was applied to the measured loss process before and after ARQ. The QoE improvement for uniform packet loss is schematically shown in Figure 8 for an 16 Mbps HD video. The obtained results show an expected drastic improvement of the QoE in the presence of ARQ. However, the achievable QoE improvement depends on the loss process obtained after applying ARQ and will thus vary with different ARQ schemes. In practical monitoring, parametric QoE models can be combined with analytical models describing the correction efficiency of the used ARQ scheme or augmented with live measurements of ARQ traffic.

## V. Related Work

The performance of ARQ in the fast RTP retransmission scheme was evaluated in [14] based on simulations. This study discusses the amount of caching required at ARQ servers and the influence of multiple retransmission requests to increase fault tolerance. Our study complements this work by evaluating multiple mechanisms and comparing the obtained results against the behavior of a real STB. To the best of our knowledge, we are the first to analyze the behavior of the MSTV STB. Additional, but less related works, include Jiang and Schulzrinne [15] which studies the efficiency of passive repair techniques subject to bursty packet loss impairing audio transmissions. The performance of FEC was studied analytically for multimedia streaming in more general settings in [16]. The packet loss introduced by a de-jitter buffer due to discarding out-of-order packets was analytically bounded in [17] for static de-jitter buffers. In addition, [18] derived the buffer depth to meet a certain packet loss rate by discarding late arrivals. The amount of buffering necessary for a smooth playback was analytically studied in [19].

## VI. Conclusion

We empirically evaluated the performance of the ARQ mechanism deployed in the MSTV Set-top Box (STB) used by several big ISPs. As concrete implementation details are proprietary, we considered three ARQ schemes of different complexity and simulated their behavior. Their performance was evaluated under different network conditions: bursty and uniform packet loss with and without jitter. Simulation results were compared to empirical measures of the STB. Based on this comparison we concluded that the STB implements simple algorithms which are sufficient for desirable QoE levels. Our analysis revealed insights into the ARQ scheme implemented by the MSTV STB. The application of a parametric QoE model showed a drastic improvement of QoE in the presence of ARQ and motivated the reflection of ARQ in QoE models.

## References

[1] P. Brooks and B. Hestnes, "User measures of quality of experience: Why being objective and quantitative is important," *IEEE Network*, vol. 24, no. 2, pp. 8–13, March 2010.
[2] M. Garcia and A. Raake, "Parametric packet-layer video quality model for IPTV," in *ISSPA*, 2010.
[3] O. Verscheure, P. Frossard, and M. Hamdi, "User-oriented QoS analysis in MPEG-2 video delivery," *Real-Time Imaging*, 1999.
[4] K. Yamagishi and T. Hayashi, "Parametric packet-layer model for monitoring video quality of IPTV services," in *ICC*, 2008, pp. 110–114.
[5] F. You, W. Zhang, and J. Xiao, "Packet loss pattern and parametric video quality model for IPTV," in *ICIS*, 2009, pp. 824–828.
[6] ITU-T, "Recommendation G.107: The E-Model, a computational model for use in transmission planning," Tech. Rep., 2003.
[7] K. Yamagishi, T. Kawano, and T. Hayashi, "Hybrid video-quality-estimation model for IPTV services," in *GLOBECOM*, 2009, pp. 1–5.
[8] J. Rosenberg and H. Schulzrinne, "Integrating packet FEC into adaptive voice playout buffer algorithms on the internet," in *INFOCOM*, Mar. 2000, pp. 1705–1714.
[9] W. Jiang and H. Schulzrinne, "Modeling of packet loss and delay and their effect on real-time multimedia service quality," in *NOSSDAV*, 2000.
[10] J. Bolot, "End-to-end packet delay and loss behavior in the internet," in *SIGCOMM*, 1993, pp. 289–298.
[11] R. Ramjee, J. F. Kurose, D. F. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide-area networks," in *INFOCOM*, 1994, pp. 680–688.
[12] L. Sun and E. Ifeachor, "New models for perceived voice quality prediction and their application in playout buffer optimization for VoIP networks," in *ICC*, 2004, pp. 1478–1483.
[13] "Extended RTP profile for real-time transport control protocol (RTCP)-based feedback (RTP/AVPF)," RFC 4585, 2006.
[14] M. J. Prins, M. Brunner, G. Karagiannis, H. Lundqvist, and G. Nunzi, "Fast RTP retransmission for IPTV – implementation and evaluation," in *GLOBECOM*, 2008, pp. 2308–2313.
[15] W. Jiang and H. Schulzrinne, "Comparison and optimization of packet loss repair methods on voip perceived quality under bursty loss," in *NOSSDAV*, 2002, pp. 73–81.
[16] P. Frossard, "FEC Performances in Multimedia Streaming," *IEEE Communications Letters*, vol. 5, no. 3, pp. 122–124, 2001.
[17] R. G. Cole and J. H. Rosenbluth, "Voice over IP performance monitoring," *SIGCOMM Comput. Commun. Rev.*, vol. 31, no. 2, pp. 9–24, 2001.
[18] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustments: performance bounds and algorithms," *Multimedia Systems*, vol. 6, pp. 17–28, 1998.
[19] A. ParandehGheibi, M. Medard, A. Ozdaglar, and S. Shakkottai, "Access-network association policies for media streaming in heterogeneous environments," *ArXiv e-prints*, Apr. 2010, 1004.3523.