Double Feedback Streaming Agent for Real-Time Delivery of Media Over 3G Wireless Networks

Gene Cheung, *Member, IEEE*, Wai-Tian Tan, *Member, IEEE*, and Takeshi Yoshimura, *Member, IEEE*

***Abstract*—A network agent located at the junction of wired and wireless networks can provide additional feedback information to streaming media servers to supplement feedbacks from clients. Specifically, it has been shown that feedbacks from the network agent have lower latency, and they can be used in conjunction with client feedbacks to effect proper congestion control. In this work, we propose the double-feedback streaming agent (DFSA) which further allows the detection of discrepancies in the transmission constraints of the wired and wireless networks. By working together with the streaming server and client, DFSA reduces overall packet losses by exploiting the excess capacity of the path with more capacity. We show how DFSA can be used to support three modes of operation tailored for different delay requirements of streaming applications. Simulation results under high wireless latency show significant improvement of media quality using DFSA over non-agent-based and earlier agent-based streaming systems.**

***Index Terms*—Multimedia communication, multimedia systems.**

I. INTRODUCTION

THIRD generation (3G) wireless cellular systems promise sufficient wireless bandwidth, currently up to 384 kbps, to support streaming video applications in addition to conventional voice and data applications [1]. Since 3G is a wireless wide-area network (W-WAN), its characteristics are distinctly different from wireless local-area networks (W-LAN) such as 802.11 networks and wired wide-area networks (WAN). Specifically, typical latencies in a wired link and a W-LAN link such as 802.11 are at most a few milliseconds. In contrast, many W-WAN systems, including several 2.5G and 3G systems, have roundtrip latencies between several hundred milliseconds to 1 s. Such large latencies are due to multiple interleaving and forward error correction (FEC) techniques that are needed to efficiently provide high capacity over wide areas [2]–[4]. The use of link-layer retransmissions over the high-delay wireless link further aggravates the latency problem.

It is known that the large latencies in 3G systems cause degradation in TCP throughput in a number of ways [1], [3], [5]. For adaptive streaming applications that rely on the timeliness of feedback for optimization, large roundtrip latencies can also significantly undermine the effectiveness of optimization procedures. A practical solution without significantly altering the end-to-end semantics is to employ an agent at the junction of the wired network and wireless link to provide additional feedbacks and loss protection. We outline such agent-based development in the following discussion.

A. End-to-End Approach

Conventional practice for streaming media ignores the effects of the last hop wireless link and employs endpoint media adaptations based solely on observable endpoint statistics. The long roundtrip latencies of W-WAN environments mean that end-to-end approaches will suffer from long reaction time. Furthermore, since endpoint statistics are aggregated across all wired and wireless links, it is impossible to distinguish the respective conditions of the links, leading to an inability to effect proper reaction. Specifically, if losses are due to wired network congestion, transmitting sources should reduce their transmission rates. In contrast, if losses are due to corruption in the wireless link, an appropriate reaction is to increase the error-resiliency of the stream instead of transmission rate reduction.

B. RTP Monitoring Agent: Statistical Feedbacks

For proper wired network congestion control, the use of *RTP monitoring agent* is proposed in [6]. It is a network agent placed at the intersection of wired network and wireless link that monitors existing streaming flows and periodically sends statistical feedbacks in the form of RTCP reports back to the senders of the flows. Let R_1^* be the permissible bandwidth of the wired network as determined by a standard wired network congestion control algorithm such as [7]. Let R_2^* be the maximum sending rate permissible for the wireless link, as determined by the base-station during wireless link resource allocation phase of the connection setup. The goal of the RTP monitoring agent is to provide feedbacks so that the source can perform proper wired network congestion control without overwhelming the wireless link. This is achieved by placing a *shaping point* in front of the agent, that "adjusts the outgoing rate of all packet traffic to the rate of the radio link" [6]. It works as follows: when $R_1^* < R_2^*$, the shaping point does nothing, and the streaming server sends at rate R_1^* by virtue of wired congestion control; when $R_1^* > R_2^*$, the shaping point drops enough packets to trigger the server to transmit at rate R_2^*. RTP monitoring agent is illustrated in Fig. 1.

C. Streaming Agent: Timely Feedbacks

Since RTCP reports are sent in mid-term (on the order of seconds to minutes) and do not contain information unique to in-

Manuscript received December 30, 2002; revised October 10, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Antonio Ortega.

G. Cheung is with Hewlett-Packard Laboratories, Tokyo 168-0072, Japan (e-mail: gene-cs.cheung@hp.com).

W.-T. Tan is with Hewlett-Packard Laboratories, Palo Alto, CA 94304 USA (e-mail: dtan@hpl.hp.com)

T. Yoshimura is with the NTT DoCoMo, Inc. Yokosuka 239-8536, Japan (e-mail: yoshi@spg.yrp.nttdocomo.co.jp).

Digital Object Identifier 10.1109/TMM.2003.822794

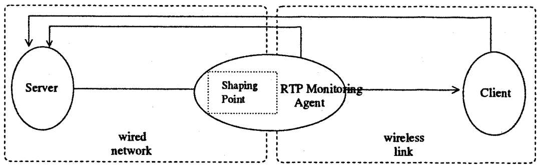

Fig. 1. RTP monitoring agent provides a second feedback to identify congestion.

dividual packets, it is argued in [8] and [9] that they are neither timely nor specific enough for many application-level optimizations, such as format adaptation or application-level retransmissions. One possible enhancement to RTP monitoring agent is to include *timely feedbacks* sent in short-term (within a second), with each feedback packet containing reception indications, rather than statistics, for packets in a stream. This enhanced agent is called streaming agent (SA). Possible gains of using SA have been shown in [8], [9].

D. Summary and Limitation of Previous Agent-Based Approaches

To summarize the previous agent proposals, the RTP monitoring agent sends statistical feedbacks to the server to perform proper wired-network congestion control, and SA sends timely feedbacks to the server to perform application-level optimizations such as format-adaptation or application-level retransmissions. Both the RTP monitoring agent and SA use a shaping point to pre-drop incoming packets to avoid overwhelming the wireless link.

Looking closely at the resulting behavior stemming from the use of the shaping point in [6], [8], [9], we see that in the case when $R_1^* < R_2^*$, as defined in Section I-B, the shaping point does not drop packets, and hence the feedbacks from the agent are reliable statistics of the wired network only. The network information that the server can induce using feedbacks from agent and client include

1) permissible sending rate in wired network R_1^*;
2) packet loss in wired network;
3) packet loss in wireless link.

In the case when $R_1^* > R_2^*$, the shaping point drops packets before packet arrival at the agent to reduce sending rate to R_2^*. The network information that the server can induce in this case include

1) permissible sending rate in wireless link R_2^*;
2) an upper bound of wired packet loss (wired loss plus drops at the shaping point);
3) packet loss in wireless link.

Of course, at any given time the server does not know whether it is in the first or second case. In the absence of that knowledge, the server can only induce the following information:

1) $\min\{R_1^*, R_2^*\}$, the resulting sending rate;
2) an upper bound of the wired packet loss;
3) packet loss in the wireless link only.

Focusing on the first item, we see that although the available network resources are R_1^* for the wired part and R_2^* for the wireless part, we are forced to send at the minimum of the two. This means available $R_1^* - R_2^*$ bandwidth is left unexploited in the wired part when $R_1^* > R_2^*$, and $R_1^* - R_2^*$ bandwidth is left unexploited in the wireless part when $R_1^* < R_2^*$. Moreover, the shaping point is enforcing this minimum sending rate behavior by artificially increasing packet losses in the second case. We identify this undesirable condition as the *network under-utilization of the shaping point.*

To address this undesirable condition, we propose in this paper a new agent called double feedback streaming agent (DFSA) that expands the capability of the streaming agent in two ways. First, DFSA provides additional feedbacks so that the server can determine and, together with DFSA, exploit the maximum allowed transmission rate at both the wired and wireless parts of the network. Second, unlike previous agent-based architectures [6], [8], [9], DFSA actively participates in the packet streaming system in one of three hybrid ARQ/FEC schemes. The actual scheme employed can be one of three modes of operation of DFSA, depending on the end-to-end delay requirement of the application. Note that although DFSA is more complex that previous agents, it nevertheless does not decode the payload of the streaming packets.

The outline of the paper is as follows. We first discuss non-agent-based related work in Section II. We then discuss the relevant characteristics of current 3G wireless networks in Section III. We then give an overview of the design of the proposed agent, DFSA, in Section IV. We then discuss the three modes of operation of DFSA in Section V. Analysis of DFSA is presented in Section VI. Results and conclusion are presented in Section VIII.

II. Non-Agent-Based Related Work

There is much previous work related to media streaming and wireless proxies. We divide related work into two categories: those focusing on transport layer and below in Section II-A, and those focusing on optimizing media streams at the application layer in Section II-B.

A. Related Work in Networking

The focus of previous work on data transport over wireless links has been on optimizing TCP over last-hop wireless links [3], [10], [11]. As an example, [10] proposed a Snoop protocol that improves the performance of TCP connections with a last-hop wireless link. The scheme employs TCP-aware link layer protocol that caches and retransmits TCP packets and suppresses negative acknowledgments from receivers.

For UDP-based transport, the IETF has been active in the RTP specification [12] to extend RTCP to include timely feedbacks [13], and to include the use of FEC in the standard RTP stream [14]. Researchers at HP Labs, Bristol, U.K., have introduced UDP Lite [15] that allows a packet with corrupted payload to be accessed. Our agent-based work is orthogonal to these developments as our proposed DFSA can potentially be modified easily to work with any of these new protocol specifications.

The idea of inserting proxies or intelligent routers at carefully chosen locations in the network is not new, and it has been reported in [16] to increase web traffic performance, in [17] to monitor network services, and in [18] for multicast video distribution. In contrast, our agents are designed for streaming delay-sensitive media content via large-latency last-hop wireless links.

B. Related Work in Optimized Media Streaming

Prior work on media streaming optimization at the application layer generally falls into three categories: 1) those that optimized media streaming at sender focusing on a few characteristics of the wireless link [19], [20], 2) those that optimized media streaming by performing intelligent application-layer scheduling or transformation, such as transcoding, at a media proxy close to the client [21]–[24], and 3) those that optimized media streaming by performing payload-blind packet-level optimization at a proxy close to the client [25].

As we have already mentioned in Section I-A, category 1 optimizations based solely on end-to-end observations cannot fully utilize the available bandwidth in both wired and wireless parts of the network and are therefore suboptimal. Our agent-based approach differs from work that depends on media proxies to perform application-layer scheduling or transformation in the following ways: 1) agents do not decode content of payloads in the media packets, and hence have lower complexity overhead, 2) by not decoding the content, agent-based approach avoids security issues since payload can be encrypted without affecting operational correctness, and 3) agent can retain the *soft state* property, similar to the Snoop agent, where the outage of the network agent—assuming the outage permits streaming traffic through—will cause a tolerable degradation of performance instead of a catastrophic breakdown of the streaming session.

The work in [25] performs packet-level protection without decoding payloads of packets. Assuming a packet loss model for the wired network and a bit error model for the wireless link, [25] suggests placement of an FEC-transcoding proxy at the junction of the wired/wireless intersection for transcoding FEC so that it offers packet loss protection in wired part and byte error protection in the wireless part. In contrast, we assume packet loss model in both wired and wireless part of the network. Moreover, our proposed DFSA fully exploits the available bandwidth in the wired and wireless part by using one of three hybrid ARQ/FEC schemes to protect packet stream.

III. Characteristics of 3G-WCDMA

We discuss here the main features of a wideband code division multiple access (WCDMA) communication system [1], the main 3G air interface technology to be deployed in Europe and Asia, including Japan and Korea. Two possible locations for DFSA within a WCDMA system are Radio Network Controller (RNC), which is responsible for the control of the radio resources of the radio access network, and transmitting base station (node B). Node B handles layer 1 processing such as channel coding and interleaving, rate adaptation, spreading etc. RNC, on the other hand, performs layer 3 packet processing such as header compression. Hence, it is logical to place the functionalities of SA at RNC.

If radio resource is properly managed by RNC, overload situations that cause wireless link congestion are exceptional ([1, p. 213]). We assume there is no wireless link congestion in this paper.

As shown in Fig. 2, when an IP packet travels from layer 3 to layer 2, it enters the radio link control (RLC) unit responsible for segmentation and retransmission services. The RLC allows three modes of transmission: acknowledged, unacknowledged, and transparent. For acknowledged mode, automatic repeat request (ARQ) is used for error control. The tradeoff between link quality and link delay can be set by adjusting the number of retransmissions, set by radio resource control (RRC) during configuration.

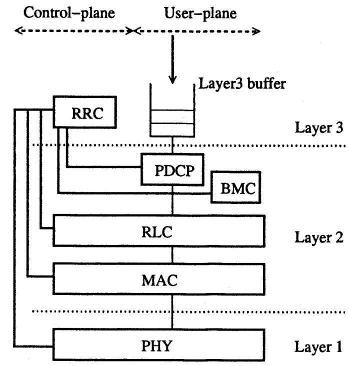

Fig. 2. IMT-2000 protocol stack.

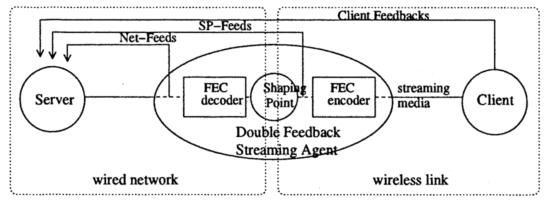

Fig. 3. Overview of DFSA.

IV. Overview of Double Feedback Streaming Agent

Our proposed network agent is called double feedback streaming agent (DFSA) and has three parts: FEC decoder and encoder, shaping point, and double feedback generator, as shown in Fig. 3. Table I shows the definitions for the symbols used in this paper. Specifically, R_1 represents the actual transmission rate at the server before entering the wired network, and R_2 represents the transmission rate at DFSA before the wireless link. Both R_1 and R_2 include packets of the actual media stream as well as loss protection packets such as retransmitted packets and FEC packets. Due to possible packet drops in the wired network and wireless link, the rates received after the wired network and wireless link, R_1' and R_2', respectively, can only be smaller: $R_1' \leq R_1$ and $R_2' \leq R_2$.

A. Finding R_2^* at DFSA

We assume the maximum permissible transmission rate in the wireless link R_2^*, though possibly time-varying, can be determined at DFSA using one of two ways depending on implementation. In the first method, DFSA knows the *nominal* transfer rate R_2^o that the wireless infrastructure has reserved during resource allocation phase of the wireless session setup. R_2^o is not likely to vary unless the session has been cut off and needs to be re-established, or the user moves to a different base station. IP packets are fragmented in RLC layer into smaller payload units (PUs) in the link layer. Suppose also that DFSA knows the raw PU loss rate γ of the link at a given time. Given a maximum of

TABLE I
DEFINITIONS FOR DFSA MEDIA STREAMING SYSTEM

Symbol	Meaning
α	packet loss rate in the wired network
R_1^*	maximum permissible transmission rate in the wired network
R_1	actual transmission rate at server
R_1'	reception rate at *DFSA* after wired loss (equals $(1-\alpha)R_1$)
$R_1^{(2)}$	rate inside *DFSA* after FEC decoding
$R_1^{(3)}$	rate inside *DFSA* after Shaping Point
γ	raw RLC payload unit loss rate in wireless link
M	number of PUs per IP packet
β	packet loss rate in the wireless link
K	number of link-layer retransmissions
R_2^o	nominal transfer rate in the wireless link
R_2^*	maximum permissible transmission rate in the wireless link
R_2	actual transmission rate at *DFSA*
R_2'	reception rate at client after wireless loss (equals $(1-\beta)R_2$)
r	streaming media coding rate

K link-layer retransmissions are used, the average number of transmissions for each PU, $\bar{K}$, is:

$$\begin{aligned}\bar{K} &= (1-\gamma)+\gamma(1-\gamma)2+\cdots+\gamma^{K-1}(1-\gamma)K+\gamma^K K\\ &= \frac{(1-\gamma^K)}{(1-\gamma).}\end{aligned}$$

The maximum wireless bandwidth, R_2^*, is obtained as

$$R_2^* = (\bar{K})^{-1}R_2^o. \quad (1)$$

With γ and K, we can also evaluate the packet loss rate in the wireless link β as

$$\beta = 1-(1-\gamma^{K+1})^M \quad (2)$$

where M is the number of PUs per IP packet.

In the second method, we assume R_2^o and γ are not available to DFSA. In this case, R_2^* is estimated simply by periodically observing $h(t)$, the instantaneous packet volume of the outgoing network queue at the base station waiting to be fragmented into PUs. See Fig. 2 for an illustration. In particular, suppose we sample the fullness of the queue in bits every Δt seconds, given the current outgoing rate of DFSA is R_2, we estimate R_2^* as

$$R_2^* = R_2 + \frac{h(t-\Delta t)-h(t)}{\Delta t}. \quad (3)$$

Given DFSA knows R_2^* at all time, we now discuss the three parts of DFSA in order.

B. FEC Decoder and Encoder

When packets first enter DFSA, an FEC decoder first channel-decodes wired network FEC if present. This possibly lowers the reception rate at DFSA from R_1' to $R_1^{(2)}$. We assume the use of systematic codes, i.e., the transmitted packets contain the original data packets plus parity packets, and the parity packets are sent on a separate stream [14]. FEC decoder tries to reconstruct the missing original data packets using the parity packets sent on a separate stream.

Before packets leave DFSA, an FEC encoder checks $R_1^{(3)}$, the rate inside DFSA after shaping point (to be discussed), against the wireless maximum permissible transmission rate R_2^*. If $R_1^{(3)} < R_2^*$, FEC encoder uses wireless link bandwidth surplus $R_2^* - R_1^{(3)}$ for FEC for protection on the wireless link. The end result is that the actual transmission rate in the wireless link R_2 is always approximately equal to R_2^*.

Employing FEC means there is a small amount of buffer required to decode FEC packets for the FEC decoder and to compute parity packets for the FEC encoder. Specifically, if Reed–Solomon (n,k) code is used, then we need buffer k packets to decode and encode FECs. Assuming the $\mathrm{RS}(n,k)$ code is relatively small, this will not introduce a large cost in buffering delay.

C. DFSA Shaping Point

Similar to [6], a shaping point is placed in the middle of DFSA. The shaping point first detects and eliminates duplicate packets with identical RTP sequence numbers. If the rate still exceeds the wireless maximum permissible transmission rate R_2^* and the IP packet buffer in the base station (see Fig. 2) is close to full, it randomly drops packets until outgoing rate is $\leq R_2^*$. In other words, the shaping point only pre-drops packets that would later be dropped anyway due to base station's buffer overflow. The end result is that any loss between FEC encoder and the client is the result of wireless link failure only, not buffer overflow.

D. Double Feedback Generator

Two feedback generators are located just before the FEC decoder and FEC encoder to provide feedbacks to the server. The preshaping point feedback generator sends *net-feeds* to the server, informing the server of the current wired network condition. net-feeds are statistical feedbacks containing summary of information collected over a window of packets, such as packet loss rate and mean and variance of packet arrival time, and are sent in the mid term (on the order of seconds). Upon receiving net-feeds, the sender can estimate loss rate and mean and variance of round trip time (RTT), used in the estimation of TCP-friendly bandwidth [7]. An example of such statistical feedbacks is RTCP reports [12]. The post-shaping point feedback mechanism sends *SP-feeds* to the server, in the form of packet acknowledgment packets (ACKs), to the server so that it can determine what packets are dropped prior to wireless transmission. These are sent in the short term (within a second). Note that SP-feeds need not send an ACK per packet if the volume of control packets is a concern. Instead, one summary ACK can be sent every P packets to conform to the recommended 5% control packets per session as recommended in [12]. See Fig. 3 for an illustration.

E. Deriving Network Parameters

The various feedbacks provide the server with sufficient information to derive the network parameters R_1^*, α, R_2^* and β. First, net-feeds allow the server to determine the maximum permissible transmission rate in the wired network, R_1^*. The wired network packet loss rate α is explicit in net-feeds. Second, because the shaping point allows no more outgoing packets than R_2^*, the server can obtain R_2^* by deducing the volume of packets received and the channel coding rate for the wireless link using client feedback.

Third, SP-feeds together with the client feedbacks provide enough information for the server to deduce the actual packet loss rate β. Specifically, by comparing SP-feeds and client feedback, the server can deduce the loss rate of the wireless link *after* FEC has been applied. Then, by using the client feedback, the server can deduce the amount of FEC applied. Using these two pieces of information, the server can deduce approximately the wireless channel loss rate β. With the deduced network parameters, the server can then use DFSA in one of three modes of operation to optimize streaming quality as detailed next.

V. DFSA Modes of Operation

Streaming media applications have widely varying latency requirements. For instance, interactive applications such as video conferencing are unusable for latencies above one second. On the other hand, it is acceptable to have tens of seconds of initial wait for applications involving viewing of a long stream, such as a movie. Given the already high latencies of 3G systems, DFSA is designed to be malleable to meet the different application needs. Specifically, for delay-tolerant applications, DFSA performs the best possible error recovery mechanisms without considering a delay constraint. In contrast, for delay sensitive applications, DFSA employs only mechanisms that incur minimal additional latency.

A. Mode I: Applications With High Delay Tolerance (ARQ/Ack)

For applications that can accept large initial start-up time, we setup the DFSA system as follows. First, the wireless link is configured with acknowledgment mode with a large maximum number of link-layer retransmissions K. This results in a nearly lossless link in a resource efficient fashion but with a large delay variation. Link layer retransmission is preferred to FEC due to the higher efficiency in bandwidth usage.

Second, net-feeds from DFSA are used for wired network congestion control at the server. Third, SP-feeds from DFSA are used for application-level retransmissions [9]. Since the wireless link is near lossless, the SP-feeds mimic the client state almost perfectly. Moreover, SP-feeds would arrive at server much faster than client feedbacks.

Note that in this mode, fine-grained timely feedbacks from the client are not necessary for the streaming system. This means 3GPP-PSS compliant clients [26], [27] that only send statistical RTCP reports but not timely feedbacks can use this mode for media streaming.

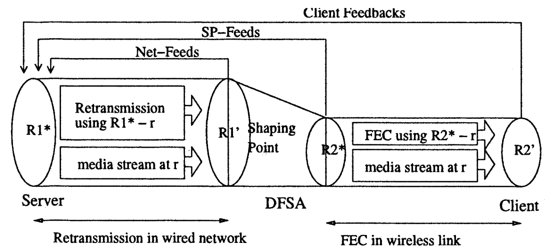

Fig. 4. Mode II: $R_1^* > R_2^*$.

B. Mode II: Applications With Medium Delay Tolerance (ARQ/FEC)

For applications that can tolerate several packet retransmissions in the wired network, but not retransmissions in the wireless link, we setup DFSA as follows.

Since the wireless link is typically more rate-constrained than the wired network, we will assume that $R_1^* > R_2^*$. As discussed earlier, the server can deduce R_2^*, R_1^* and β using the various feedbacks. This allows a server to choose a media coding rate r so that FEC at channel coding rate $R_2^* - r$ is appropriate for wireless loss rate β. The surplus bandwidth in the wired network of $R_1^* - r$ is exploited using application-level retransmissions. Since duplicate packets are dropped at the shaping point in DFSA, they will not overwhelm the wireless link. See Fig. 4 for an illustration.

C. Mode III: Application With Low Delay Tolerance (FEC/FEC)

For delay sensitive applications where it is inappropriate to use retransmissions in the wired or wireless part of the network, we setup DFSA as follows. We first use FEC in the wired network that is tailored for the loss characteristics of the wired network. This layer of FEC will be decoded at DFSA by the FEC decoder. Then FEC tailored for the wireless link will be added by the FEC encoder. Link-layer retransmission is turned off. We term this FEC conversion in the middle of the delivery path *FEC transcoding*. It is outside the scope of this paper to discuss more complex operations that can be potentially performed in a network agent to improve streaming quality [21]–[24]. The purpose of FEC transcoding operation in this paper is to provide a single concrete example that exploits the ability of DFSA to identify asymmetric transmission rates in the wired and wireless parts of the delivery path without decoding the packet payload.

Since FEC is often used with some amount of interleaving, there is an inherent delay associated with this mode. For applications in which even such delays are inappropriate, DFSA may not be useful for the purpose of mid-network processing. In such scenario, additional techniques that improve source characteristics such as error resilient source coding can be used. The incorporation of such techniques is beyond the scope of this paper.

VI. Performance Analysis of Streaming Using DFSA

In this section, we analyze the performance of a media streaming system using DFSA under one of three modes of operation discussed in Section V. Such analysis not only predicts the potential gain of using DFSA, but it also juxtaposes the performance of the three DFSA modes for different delay requirements. We begin in Section VI-A with a brief discussion

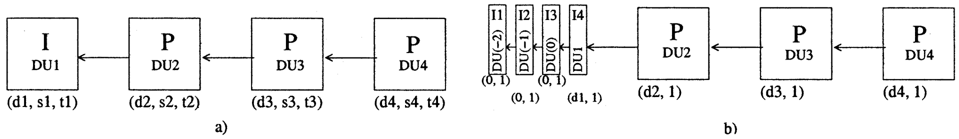

Fig. 5. Directed acyclic graph based source model. I-frame is shown broken into four data units, DU_{-2} to DU_1.

of a source model we are using for the analysis. We then present the performance analysis of the three modes in turn in Sections VI-B–D, respectively. We then conclude the section with an analytical comparison among the three modes in Section VI-E.

A. Source Model

The source model we employ is a simplified version of the directed-acyclic graph (DAG) abstraction proposed in [28]. The dependency in streaming content is modeled by a DAG of *data units*. In [28], each data unit, DU_i, is defined by a triple, (d_i, s_i, t_i), representing the reduction in distortion if DU_i is delivered and decoded correctly, and the size and delivery deadline of DU_i, respectively. When a data unit DU_i points to a set of data units DU_j's, DU_i can be decoded correctly iff DU_i is delivered and all DU_j's are decoded correctly as well. Fig. 5(a) illustrates the DAG model for data units that are I- and P- frames of typical encoded motion-compensated video.

To ease the analysis, we simplified the model in [28] further to that of Fig. 5(b). Each data unit has the same size — one RTP packet. The leading I-frame is divided into four data units, since I-frames are typically larger than P-frames. The distortion reductions d_i's of the first three data units are zero, meaning they contribute nothing unless all four data units of the I-frame are received. In a full analysis, the delivery deadlines should be considered. However, as we will discuss later, when the expected number of retransmissions is small, the additional delay is negligible, and the delivery deadline constraints can be omitted.

The DAG model of [28] is chosen as a compromise between complexity and accuracy. This model does not attempt to incorporate error concealment present in video communication, and more recent work [29], [30] improves upon this model at the cost of higher complexity.

B. Mode I Analysis

Suppose the transmitted video sequence consists of one I-frame followed by $L-1$ P-frames. Let G_1 be the DAG representation of the L-frame sequence. Let p_i be the probability that DU_i is *delivered* correctly in the wired network. We assume the wireless link packet loss probability to be zero given the use of link-layer retransmission scheme discussed in Section V-A. Assuming independent wired losses, we can write the expected end-to-end distortion for G_1 as

$$D(G_1) = D_o - \sum_{i=1}^{L} d_i \prod_{j \le i} p_j \tag{4}$$

where D_o is the distortion at the end client if no data unit is received correctly.

Let w be the transmission bandwidth for the L-frame sequence in packet transmission attempts. Let $r_{tr}(w,j)$ be the probability that transmission of DU_j is permitted and successful. Finally, let $r_i(w,j)$ be the probability that the ith retransmission of DU_j is required (previous transmission(s) of DU_j unsuccessful), permitted and successful. Transmission will not be permitted if the number of transmissions has already exceeded the budget w. We can now write DU_j's wired network delivery probability, p_j, as follows:

$$p_j = r_{tr}(w,j) + r_1(w,j) + r_2(w,j) + \cdots. \tag{5}$$

Assuming the wired network loss rate α is relatively small, we can neglect terms involving second retransmission and above.

To evaluate $r_{tr}(.,.)$ and $r_1(.,.)$, we make the following assumptions. First, because of the chain dependency of I and P frames, we give transmission priority to the earlier frames. Second, since we assume that at most one retransmission is needed for small α, it follows that the playback deadline for the DU is not likely to be violated given the relatively low delay of wired network.

We can now evaluate $r_{tr}(w,j)$ as follows. Given earlier assumption that an I-frame spans four DUs, there are $j+2$ data units, DU_{-2} to DU_{j-1}, that precede DU_j. See Fig. 5 for an illustration of DU labeling. These $j+2$ DUs can be divided into two groups: s DUs each requiring only one transmission for successful delivery, and $j+2-s$ DUs each requiring two transmissions for successful delivery. In order to have transmission budget for DU_j, the total number of transmissions of these two groups cannot exceed $w-1$:

$$s + 2(j+2-s) \le w - 1. \tag{6}$$

The inequality (6) in effect sets a lower limit for s. This lower limit, s^*, is given by

$$s^* = \max(0, 2j - w + 5). \tag{7}$$

To find $r_{tr}(w,j)$, we sum up all possible value of s and multiple by $1-\alpha$ for the transmission success probability of DU_j:

$$r_{tr}(w,j) = (1-\alpha) \sum_{s=s^*}^{j+2} \binom{j+2}{s} (1-\alpha)^s \alpha^{j+2-s}. \tag{8}$$

Given the definitions of $r_{tr}(w,j)$ and $r_1(w,j)$, it follows that

$$r_1(w,j) = r_{tr}(w-1,j)\alpha. \tag{9}$$

Hence we can approximate the wired network success delivery probability for data unit j, p_j, as

$$p_j = r_{tr}(w,j) + r_{tr}(w-1,j)\alpha. \tag{10}$$

C. Mode II Analysis

We can analyze mode II in a similar fashion to mode I, where we assume the transmitted video sequence contains one I-frame followed by $L-1$ P-frames. Let G_1 be the DAG representation of the L-frame sequence. Let p_i and q be the probabilities that DU_i is *delivered* correctly in the wired network and the wireless link, respectively. Assuming independent losses, the end-to-end distortion for G_1 is

$$D(G_1) = D_o - \sum_{i=1}^{L} d_i q^i \prod_{j \leq i} p_j \tag{11}$$

where D_o is again the initial distortion of the end client if no data unit is received correctly.

For maximum error correction capability, we employ an (n,k) systematic Reed-Solomon code, where $n-k$ parity packets are added to the original k source packets. As a result $1-q$ can be expressed as the probability that the original packet is lost multiplied by the probability that $n-k$ or more packets of the other $n-1$ packets are lost:

$$1-q = \beta \sum_{j=n-k}^{n-1} \binom{n-1}{j} \beta^j (1-\beta)^{n-1-j}. \tag{12}$$

The method in Section VI-B is used to calculate the wired packet loss, p_i, for data unit i.

D. Mode III Analysis

The success probability of a data unit transmitted over a channel with loss probability β using an (n,k) RS code is given in (12). The expected distortion of a set of data units G_1 using Mode III is

$$D(G_1) = D_o - \sum_{i=1}^{L} d_i q^i p^i \tag{13}$$

where q and p are expressed as follows:

$$1-q = \beta \sum_{j=n-k}^{n-1} \binom{n-1}{j} \beta^j (1-\beta)^{n-1-j} \tag{14}$$

$$1-p = \alpha \sum_{j=m-l}^{m-1} \binom{m-1}{j} \alpha^j (1-\alpha)^{m-1-j} \tag{15}$$

where (m,l) and (n,k) RS codes are used for the wired network and wireless link, respectively.

For a specific implementation, we set $m=n$ and set l and k according to permissible bandwidths R_1^* and R_2^* and encoding rate r:

$$\left(\frac{m}{l}\right) r \leq R_1^* \tag{16}$$

$$1 = \left\lceil \frac{mr}{R_1^*} \right\rceil. \tag{17}$$

Assuming $r \leq R_2^*$, shaping point does not drop any data units, and hence $R_1^{(2)} = R_1^{(3)}$. Note that $R_1^{(2)}$ is essentially r plus the effect of wired network loss after $\mathrm{RS}(m,l)$ has been applied. The experienced wired network loss rate α' is

$$\alpha' = \alpha \sum_{j=m-l}^{m-1} \binom{m-1}{j} \alpha^j (1-\alpha)^{m-1-j}. \tag{18}$$

Using α', we can derive k similar to (17):

$$k = \left\lceil \frac{n(1-\alpha')r}{R_2^*} \right\rceil. \tag{19}$$

E. Comparison of the Three Modes

Given the analysis of the three modes of operation, we can compare their streaming performance. We obtained the distortion values d_i's using the first 50 frames of the standard test sequence, `carphone` QCIF (176×144), at bit rate 230 kbps, 30 fps, and one I-frame every five frames. We fixed the wired network bandwidth R_1^* at ten packet transmission attempts per five -frame time, and the wireless link bandwidth R_2^* at nine packet transmission attempts per five-frame time. That translates to $w = 10$ for retransmission for Mode I and Mode II for the wired network, RS(9,8) code for FEC for Mode II and Mode III for the wireless link, and RS(10,8) code for FEC for Mode III for the wired network. The performance of the three modes using empirically obtained d_i's and the equations in Sections VI-A–D are shown in Fig. 6, with line i indicating the performance of mode i.

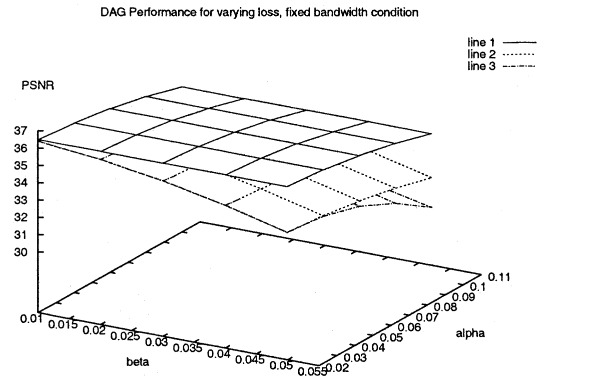

Fig. 6. Performance of mode I-III (line 1–3) for varying wired (α) and wireless (β) loss rates.

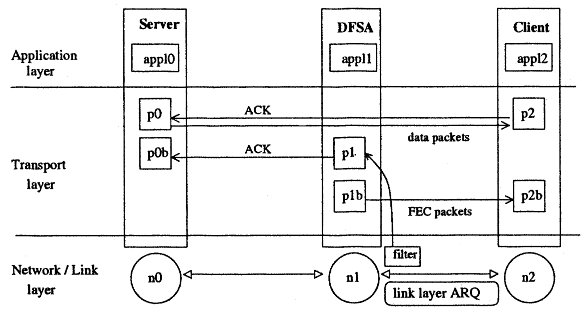

Fig. 7. Simulation setup.

We see that, as expected, Mode I outperformed Mode II, which in turn outperformed Mode III. This is expected, since we have to tolerate the highest end-to-end delay using Mode I, then Mode II and III. Notice also the performance differences among the three modes are most pronounced when both the wired and wireless conditions are poor.

VII. Results

A. Simulation Setup

We performed simulations using Network Simulator 2 [31]. The setup is shown in Fig. 7. Three nodes are constructed, $n0$, $n1$ and $n2$, representing the locations of the server, DFSA and the wireless client, respectively. To connect these nodes, two links are constructed. Link $\mathtt{n0}-\mathtt{n1}$, simulating the wired network between the server and DFSA, has constant propagation delay and uniform packet loss rate α. For link $\mathtt{n1}-\mathtt{n2}$ that simulates the wireless link of rate 144 kbps, we implemented link-layer retransmission as follows: a network layer packet is fragmented and grouped into transport blocks of 180 bytes each that spans 10 ms. Groups of x transport blocks are interleaved (spread) to give a one-way delay of $10x$ ms. In reality, larger spread reduces the probability of error due to fading at the cost of a larger end-to-end delay. Each transport unit is transmitted through the wireless link with success probability γ. When link-layer retransmission is used, as in the case for mode 1 of DFSA, the unit of retransmission is a transport block. The maximum number of retransmission is set at 20.

The transport layer has a duplex connection $(\mathtt{p0}-\mathtt{p2})$ from the server n0 to the client n2, a simplex connection $(\mathtt{p1}-\mathtt{p0b})$ from *DFSA* n1 to the server, and a simplex connection $(\mathtt{p1b}-\mathtt{p2b})$ from DFSA to the client. The $\mathtt{p0}-\mathtt{p2}$ connection is for endpoint data transmission from server to client and for feedbacks from client to server. The $\mathtt{p1}-\mathtt{p0b}$ connection is for feedbacks from DFSA to server. A packet sniffer is used to identify packets targeted to the client and then forward those packets to p1, which then sends ACKs to the server. The $\mathtt{p1b}-\mathtt{p2b}$ connection is for FEC packets when FEC is used from DFSA to client.

A server application, app0, sits at sender node and sends packets to the client using the connection $\mathtt{p0}-\mathtt{p2}$. Each packet has a sequence number in the packet header indicating the frame it contains. If it is an FEC packet of a $\mathrm{RS}(n,k)$ code, it also contains the values n and k as well as the index i identifying which packet it is in the n-packet group. A random number is also included as group ID to distinguish the current n-packet group from other n-packet groups.

For real video data, we use H.263 version 2 video codec (TMN10) to encode two 300-frame sequences container and foreman. They are coded in QCIF (176×144) format at 120 kbps, 30 frames/s and at one I-frame every 25 frames. The resulting average peak signal-to-noise ratio (PSNR)[1] for the compressed streams are 38.49 dB and 32.46 dB, respectively. During the experiment, when the receiver is unable to decode a certain frame i, the most recently correctly decoded frame j is used for display for frame i, and we calculate the PSNR using original frame i and encoded frame j. If no such frame j is available, then PSNR is 0.

B. Mode I: Applications With High Delay Tolerance

For applications with high delay tolerance, it is appropriate to employ large initial setup time at the client and to employ acknowledged mode in link layer. We compared three schemes under such circumstances: DFSA under mode I, end-to-end feedback without an agent (*NoAgt*), and end-to-end feedbacks with RTP monitoring agent for wired network statistics (*Agt*).[2]

For each scheme, a version of the rate-distortion optimized ARQ algorithm described in [28, pp.12–13] was employed at the server. In brief, at each optimization instance, exactly one data unit within an optimization window of data units was selected for transmission. The packet(s) corresponding to the selected data unit was(were) then transmitted, spaced apart in time T_s by an equation based congestion control algorithm [7]:

$$T_s = \mu_R\sqrt{\frac{2\epsilon_R}{3}} + 3(\mu_R + 4\sigma)\epsilon_R(1+32\epsilon_R^2)\sqrt{\frac{3\epsilon_R}{8}} \quad (20)$$

where ϵ_R, μ_R and σ_R^2 are the estimated packet loss rate, mean and variance of RTT, respectively. A new data unit was selected again T_s after the last packet is sent. See [28] for more details.

Fig. 8 shows the results when the wired packet loss rate and the wireless transport unit loss rate were 5% and 4%, respectively. The simulation time for each point was 600 seconds, in which each 10-s sequence was transmitted 60 times. Since SP-feeds arrived at the server much faster than client feedbacks, the server reacted more quickly by retransmitting lost packets when DFSA feedbacks were available. As a result, we see that DFSA maintained significant PSNR improvement over *Agt* and *NoAgt*. Specifically, over a wide range of wired and wireless delays, DFSA maintained about 1 dB and 2 dB improvement over *Agt* for sequences container and foreman, respectively. For the *NoAgt* scheme, the performance difference was even larger, up to 7 dB and 12 dB for container and foreman, respectively. We see that the *NoAgt* scheme performed reasonable well under low delay conditions. However, as delay increased, its effectiveness began to drop drastically due to the sole dependence on end-to-end feedback.

C. Mode II: Applications With Medium Delay Tolerance

For Mode II, we assumed a 1 s initial client buffer. The wired network, wireless link delays were 20 ms and 80 ms, respectively. We assumed the maximum permissible wireless transmission rate R_2^* to be 133 kbps. We turned off Acknowledgment mode at the wireless link layer; instead, a RS(10,9) code was used. Given the video stream was 120 kbps, that means the nominal transfer rate in the wireless link R_2^o would not exceed 133 kbps. The same two video sequences, container and foreman, were used for this part of the experiment, for varying wired and wireless packet loss rates from 0.01 to 0.1. As before, each data point on the plot is the result of a 600 s simulation in which the 10 s video sequences were repeated 60 times for an averaging effect.

Our DFSA scheme for Mode II used the discussed ARQ algorithm for the wired part and RS(10,9) for the wireless part. Because the DFSA scheme could fully utilize the maximum permissible wired network bandwidth R_1^*, computed using (20), $R_1^* - 120$ kbps was available for retransmission.

We compared our DFSA scheme to two other schemes. The *NoAgt* scheme employed the same optimized ARQ algorithm but used no network agents and relied on end-to-end statistics for congestion control and packet acknowledgments. The *RTP-Agt* scheme also employed the same ARQ algorithm, but used RTP monitoring agent for wired network statistics. No agent packet acknowledgments (DFSAs SP feeds) were available, however. Both schemes were rate-constrainted by the maximum wireless transmission rate $R_2^* = 133\,\mathrm{kbps}$ during any part of the transmission.

The performance in average PSNR of the three schemes are shown in Fig. 9 for varying wired network and wireless link

[1] $\mathrm{PSNR} = 20\log_{10}(255/\sqrt{\mathrm{MSE}})$

[2] Packet scheduling schemes via a network proxy [21], [22] are not used for comparison here because they require application-level knowledge such as distortion values of data units. For security and complexity reasons at the agent, we are only comparing schemes that only operate on packet headers.

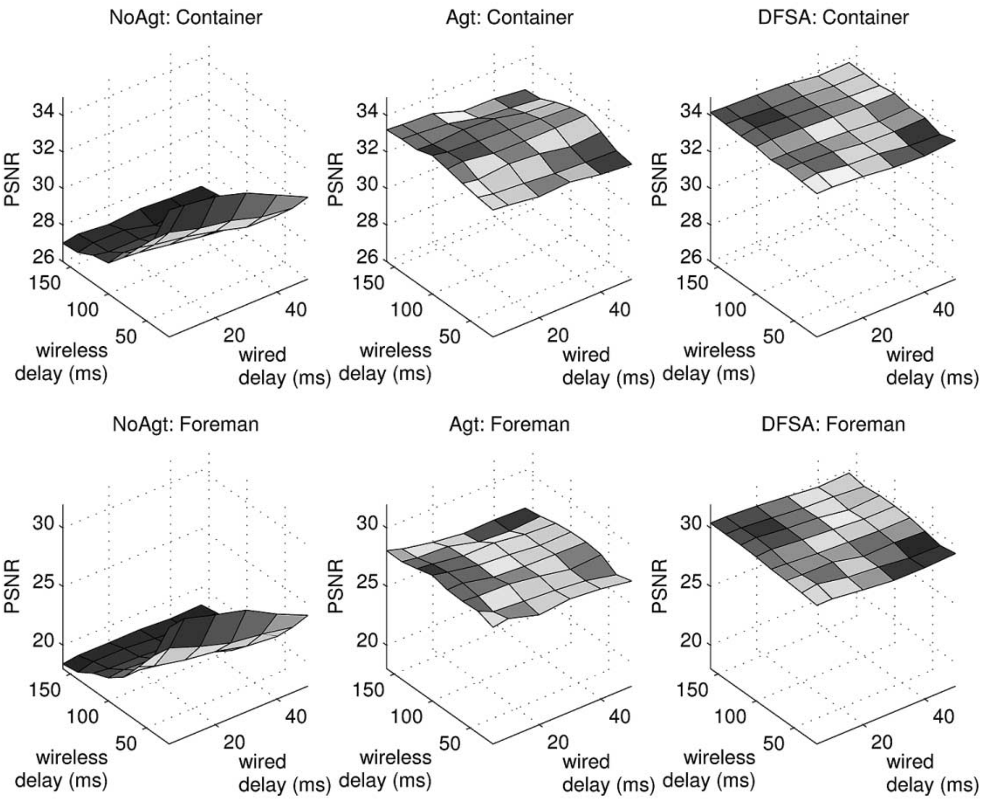

Fig. 8. Comparison of various streaming for mode I.

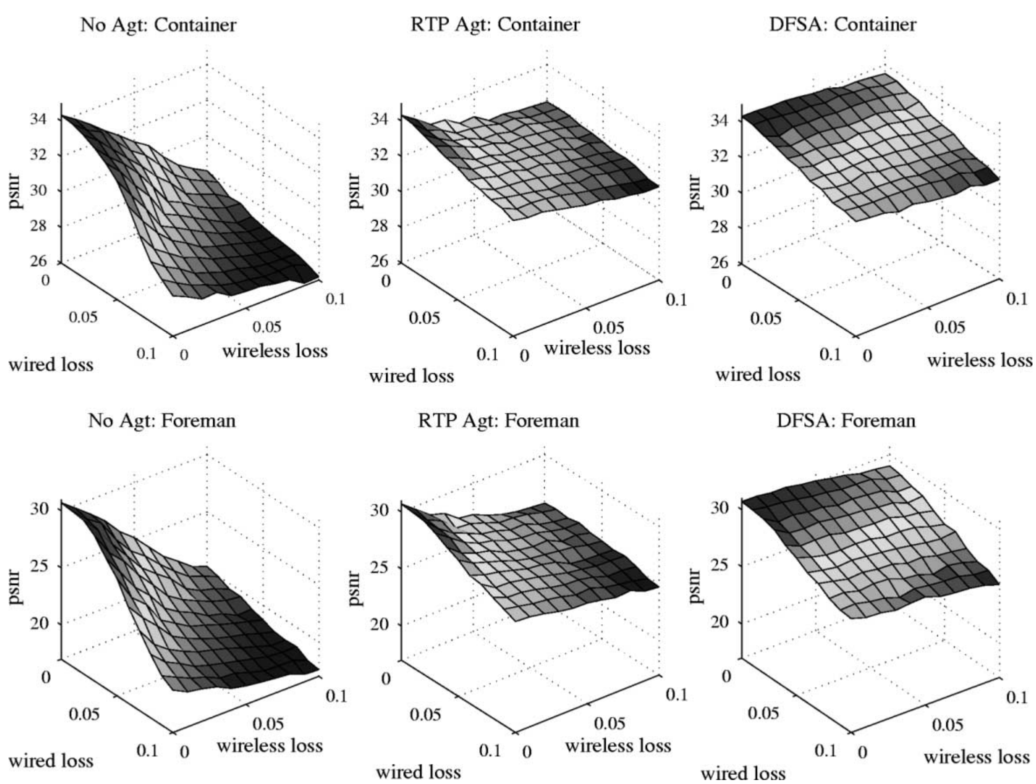

Fig. 9. Comparison of various streaming schemes for mode II.

packet loss rates. Several observations can be made immediately. First, while the performance of all three schemes degraded as the wired or wireless condition worsened, our DFSA scheme was the most error resilient of the three. Second, our DFSA scheme appeared more sensitive to wired loss than wireless loss. The reason is that poor wired network condition, in addition to

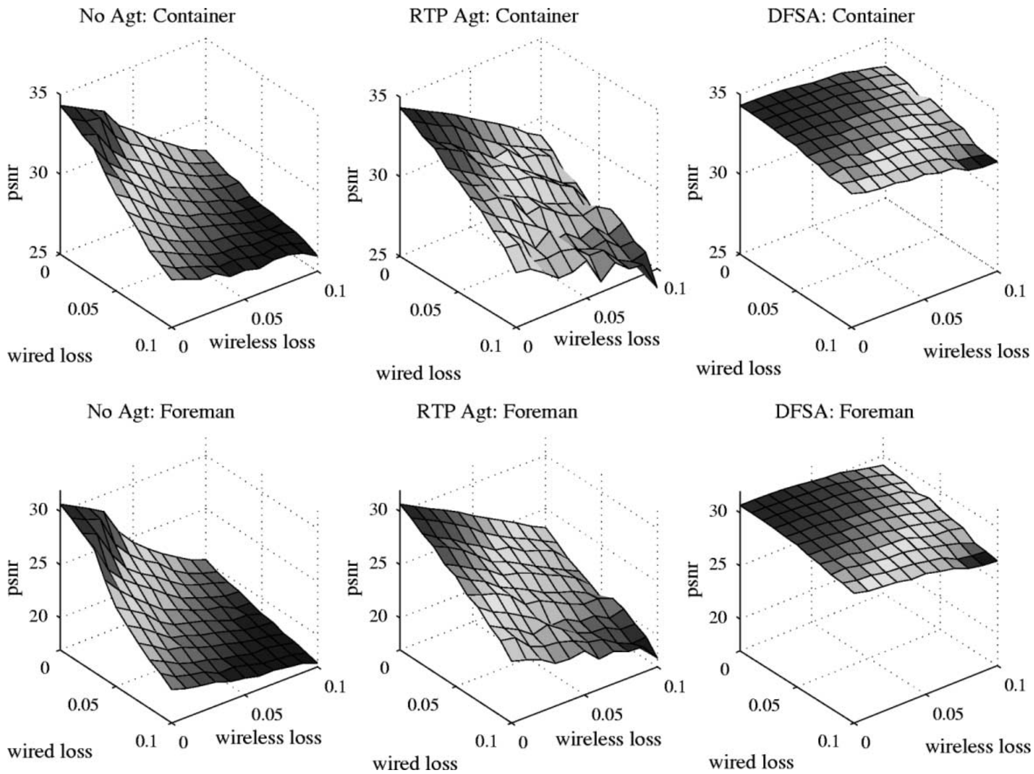

Fig. 10. Comparison of various streaming schemes for mode III.

corruption of packets, reduced the wired network bandwidth R_1^* via (20). Third, the performance difference between our DFSA scheme and *RTP-Agt* scheme was the largest when the wired network condition was good and the wireless condition was poor, up to 1.82 dB for `container` sequence and 3.37 dB for `foreman` sequence. The reason is twofold. First, good wired network condition implies a high wired network bandwidth R_1^* via (20), which our DFSA scheme was able to exploit for retransmission purposes. Specifically, DFSA has a retransmission budget of $R_1^* - 120$ kbps. The *RTP-Agt* scheme, on the other hand, was constrained by $\min\{R_1^*, R_2^*\}$, which was the maximum wireless transmission rate $R_2^* = 133$ kbps in this case. Second, the FEC protection RS(10,9) used by DFSA scheme can effectively recover losses in low packet loss environment, but not at high loss rates.

D. Mode III: Applications With Low Delay Tolerance

For Mode III, we used the same source and network parameters as we did for Mode II, with the exception that the client buffer was changed to 0.5 s. Our DFSA scheme used RS(10,9) for the wireless link and the strongest $\mathrm{Rs}(n, n-1)$ possible (smallest integer n possible) given wired network constraint R_1^*, again determined using (20). The *NoAgt* scheme used the strongest $\mathrm{RS}(n, n-1)$ possible, but with no agent feedback available, relied on end-to-end feedbacks for congestion control. *RTP-Agt* scheme differed from *NoAgt* scheme in that RTP monitoring feedbacks were available for proper wired network congestion control. Note that while our DFSA scheme could fully utilize R_1^* in wired network and R_2^* in wireless link, both *NoAgt* and *RTP-Agt* schemes were rate-constrained by $\min\{R_1^*, R_2^*\}$ under all testing scenarios.

The performance of the three schemes under various wired network and wireless link packet loss rates are shown in Fig. 10. We see that the plots exhibit the same trends we saw in Mode II. In this mode, the performance difference between our DFSA scheme and the *RTP-Agt* scheme was more dramatic: up to 8.08 dB for `container` sequence and 9.04 dB for `foreman` sequence.

VIII. Conclusion

Previous proposals for network agents to assist streaming have neglected the potential discrepancy between the capacity of the wired and wireless parts of the delivery path. In this paper, we propose to expand the capability of previous network agent in two ways. First, DFSA provides feedbacks so that the server can determine and exploit the maximum allowed transmission rate at both the wired and wireless parts of the network. Second, DFSA actively participates in media delivery to allow a different error-control mechanism to be employed on the wireless link compared to that of the wired network. Specifically, depending on the latency constraint of streaming applications, DFSA can flexibly switch among three operation modes of varying latency overhead. Simulation results show significant improvements in terms of PSNR compared to existing agent approaches.

References

[1] H. Holma and A. Toskala, Eds., *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications*, 2nd ed. New York: Wiley, 2002.

[2] E. Dahlman, P. Beming, J. Knutsson, F. Ovesjo, M. Persson, and C. Roobol, “WCDMA—The radio interface for future mobile multimedia communications,” *IEEE Trans. Veh. Technol.*, vol. 47, pp. 1105–18, Nov. 1998.

[3] H. Inamura, G. Montenegro, R. Ludwig, A. Gurtov, and F. Khafizov, "TCP Over Second (2.5G) and Third (3G) Generation Wireless Networks,", RFC 3481, 2003.
[4] R. Wu, "QVoice UMTS/WCDMA Measurement Results," Tech. Rep., Ascom AG, Solothurn, Switzerland, May 2002.
[5] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya, "Long Thin Networks,", IETF RFC 2757, 2000.
[6] T. Yoshimura, T. Ohya, T. Kawahara, and M. Etoh, "Rate and robustness control with RTP monitoring agent for mobile multimedia streaming," in *IEEE Int. Conf. Communications*, vol. 4, New York, Apr. 28–May 2, 2002, pp. 2513–2517.
[7] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *ACM SIGCOMM*, Stockholm, Sweden, Aug. 2000, pp. 43–56.
[8] G. Cheung and T. Yoshimura, "Streaming agent: A network proxy for media streaming in 3G wireless networks," in *Packet Video Workshop*, vol. 1, Pittsburgh, PA, May 2002, pp. 529–532.
[9] G. Cheung, W. t. Tan, and T. Yoshimura, "Rate-distortion optimized application-level retransmission using streaming agent for video streaming over 3G wireless network," in *IEEE Int. Conf. Image Processing*, vol. 1, Rochester, NY, Sept. 2002, pp. 529–532.
[10] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Networking*, vol. 5, pp. 756–769, Dec. 1997.
[11] B. Liu, D. Goeckel, and D. Towsley, "TCP-cognizant adaptive forward error correction in wireless networks," in *IEEE GLOBECOM'02*, vol. 3, Nov. 17–21, 2002, pp. 2128–2132.
[12] H. Schulzrine, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Application,", IETF RFC 1889, 1996.
[13] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Rey, "Extended RTP Profile for RTCP-Based Feedback: Draft-ietf-avt-rtcp-feedback-07.txt,", IETF Internet-draft, 2003.
[14] J. Rosenberg and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction,", IETF RFC 2733, 1999.
[15] L.-A. Larzon, M. Dagermark, and S. Pink, "Udp Lite for Real Time Multimedia Applications," HP Laboratories Bristol, Tech. Rep. HPL-IRI-1999-001, 1999.
[16] M. Margaritidis and G. Polyzos, "Mobiweb: Enabling adaptive continuous media applications over 3G wireless links," *IEEE Personal Commun. Mag.*, vol. 7, no. 6, pp. 36–41, Dec. 2000.
[17] M. Gunter and T. Braun, "Internet service monitoring with mobile agents," *IEEE Network Mag.*, pp. 22–29, May/June 2002.
[18] R. Keller, S. Choi, M. Dasen, D. Decasper, G. Fankhauser, and B. Planner, "An active router architecture for multicast video distribution," in *IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 1137–1146.
[19] Q. Zhang, W. Zhu, and Y.-Q. Zhang, "Network-adaptive scalable video streaming over 3G wireless network," in *Proc. IEEE Int. Conf. Image Processing*, Thessaloniki, Greece, Oct. 2001, pp. 579–582.
[20] H. Matsuoka, T. Yoshimura, and T. Ohya, "Design, implementation and performance measurement of multimedia streaming protocol (msp)," presented at the Asian Int. Mobile Computing Conf. (AMOC'2002), May 2002.
[21] J. Chakareski, P. Chou, and B. Girod, "Computing rate-distortion optimized policies for hybrid receiver/sender driven streaming of multimedia," in *Asilomar Conference on Signals, Systems, and Computers*, vol. 2, Pacific Grove, CA, Nov. 2002, pp. 1310–1314.
[22] ——, "Rate-distortion optimized streaming from the edge of the network," in *IEEE Workshop on Multimedia Signal Processing*, St. Thomas, U.S. Virgin Islands, Dec. 2002, pp. 49–52.
[23] S. Dogan, A. Cellatoglu, M. Uyguroglu, A. Sadka, and A. Kondoz, "Error-resilient video transcoding for robust internetwork communications using gprs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 453–464, June 2002.
[24] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: An overview," *IEEE Signal Processing Mag.*, pp. 18–29, Mar. 2003.
[25] A. Lee, G. Chan, Q. Zhang, W.-W. Zhu, and Y.-Q. Zhang, "Optimal allocation of packet-level and byte-level fec in video multicasting over wired and wireless networks," in *GLOBECOM*, vol. 3, San Antonio, TX, Nov. 2001, pp. 25–29.
[26] (2001) 3GPP TS 26.233 Transparent End-to-End Packet Switched Streaming Services (PSS); General Description (Release 4). [Online]. Available: ftp://ftp. 3gpp.org/Specs/2001-03/Rel-4/26_series/26 233-400.zip
[27] (2001) 3GPP TS 26.234 Transparent End-to-End Packet Switched Streaming Services (PSS); Protocols and Codecs (Release 4). [Online]. Available: ftp://ftp. 3gpp.org/Specs/2001-03/Rel-4/26_series/26 234-400.zip
[28] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," Microsoft Research Tech. Rep. MSR-TR-2001–35, vol. 3, Feb. 2001.
[29] G. Cheung and W. t. Tan, "Directed acyclic graph based source modeling for data unit selection of streaming media over QoS networks," in *IEEE Int. Conf. Multimedia and Expo.*, Lausanne, Switzerland, Aug. 2002, pp. 81–84.
[30] J. Chakareski and B. Girod, "Rate-distortion optimized packet scheduling and routing for media streaming with path diversity," in *IEEE Data Compression Conference*, Snowbird, UT, Mar. 2003, pp. 203–212.
[31] (2003) The Network Simulator ns-2 Release 2.26. [Online]. Available: http://www.isi.edu/nsnam/ns

Gene Cheung (M'00) received the B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1998 and 2000 respectively.

Since August 2000, he has been a Member of Technical Staff, Hewlett-Packard Laboratories Japan, Tokyo. His research interests include signal processing, computer networks and optimization.

Wai-Tian Tan (M'01) received the B.S. degree from Brown University, Providence, RI, in 1992, the M.S.E.E. degree from Stanford University, Stanford, CA, in 1993, and the Ph.D. degree from the University of California, Berkeley, in 2000.

He joined HP Labs, Palo Alto, CA, in December 2000 and is a Member of the Media Communications and Networking Department. He worked for Oracle Corporation, Redwood Shores, CA, from 1993 to 1995. His research focuses on adaptive media streaming, both at the end-point and inside the delivery infrastructure.

Takeshi Yoshimura (M'01) received the B.E. and M.E. degrees from the Department of Information and Communication Engineering, University of Tokyo, Tokyo, Japan.

He is a Research Engineer at NTT DoCoMo's Multimedia Laboratories, Yokosuka, Japan. His research interests include mobile streaming media technology and content delivery network architecture.