Real-Time Video Transport Optimization Using Streaming Agent Over 3G Wireless Networks

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

***Abstract*—Feedback adaptation has been the basis for many media streaming schemes, whereby the media being sent is adapted in real time according to feedback information about the observed network state and application state. Central to the success of such adaptive schemes, the feedback must: 1) arrive in a timely manner and 2) carry enough information to effect useful adaptation. In this paper, we examine the use of feedback adaptation for media streaming in 3G wireless networks, where the media servers are located in wired networks while the clients are wireless. We argue that end-to-end feedback adaptation using only information provided by 3G standards is neither timely nor contain enough information for media adaptation at the server. We first show how the introduction of a streaming agent (SA) at the junction of the wired and wireless network can be used to provide useful information in a timely manner for media adaptation. We then show how optimization algorithms can be designed to take advantage of SA feedbacks to improve performance. The improvement of SA feedbacks in peak signal-to-noise ratio is significant over nonagent-based systems.**

***Index Terms*—Computer networks, multimedia communication, multimedia systems.**

I. Introduction

THE GOAL of this paper is to improve real-time video transport over 3G wireless networks [1]. By real-time video transport, we mean a piece of video content being delivered from a server in a wired network to a mobile client via a last-hop wireless link, to be decoded and viewed by the client before the entire content has been downloaded. This video streaming service must be compliant with the 3GPP packet streaming service (3GPP-PSS) [2], where the server uses IETF RFC-compliant RTP [3] for media transport and each client sends only RTCP reports as feedback to its server.

One common objective of media adaptation is congestion control whereby video sources reduce their transmission rates in reaction to deduced network congestion [4]. For paths involving wired and wireless links, it has been shown [5] that end-to-end feedback information alone is ineffective for congestion control purposes since it is not possible to identify where losses occur. Specifically, if losses occur in the wireless link due to poor wireless condition, it is not helpful for the sources to reduce their transmission rate. On the other hand, if losses occur in the wired network due to congestion, the sources should reduce their transmission rate. One effective mechanism to provide additional information that allow sources to take appropriate actions is the RTP monitoring agent [5]—a network agent located at the junction of the wired and wireless network that sends *statistical feedbacks* (RTCP reports in particular) back to the sender to help the sender determine the proper action. However, the limited information contained in such statistical feedbacks is often insufficient for fine-grained application-level streaming optimization schemes [6]–[9].

TABLE I
DEFINITIONS OF NETWORK AND APPLICATION STATES

wired network state	statistical wired network info, such as round trip time and loss rate
wired application state	which packets arrived at wired network edge correctly and on-time
end application state	which packets arrived at the client correctly and on-time

Another limitation of using only end-to-end feedbacks is the long time for the feedbacks to arrive. In today's 3G wireless network, typical one-way delay of radio links is quite large—on the order of 100 ms—without link layer retransmissions [10]. Thus, the actual end-to-end delay in practice can be quite large, especially with wireless link-level retransmissions implemented. Such long delay can severely impede the effectiveness of feedback information for the purpose of congestion control and beyond.

Both problems above can be solved simultaneously using a special agent called a streaming agent (SA) [11], located at the junction of the wired network and wireless link. Unlike the RTP monitoring agent [5], which provides only statistical feedbacks such as average roundtrip time (RTT) and packet loss rate, SA sends *timely feedbacks*, such as acknowledgment packets (ACKs), that tell the sender whether each packet has arrived at SA correctly and on time. We call such information provided by SA the *wired application state*, in contrast with information provided by RTP monitoring agent, which we term *wired network state*. Obviously, using fine-grained timely feedbacks one can derive wired network state as well as wired application state. (See Table I for a list of definitions.) Since most of the delay is in the wireless link, SA can provide much faster response about the condition of the wired network so that congestion control can react faster to alleviate network congestion. Furthermore, by providing the wired application state rather than just the wired network state, SA allows senders to have much more flexibility in media adaptation than it is possible with wired network state alone.

Armed with SA's extra feedbacks, we next design application-level streaming systems that can take advantage of these

Manuscript received October 8, 2002; revised February 17, 2004. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Chang Wen Chen.

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

W. Tan is with Hewlett-Packard Laboratories, Palo Alto, CA 94306 USA.

T. Yoshimura is with NTT DoCoMo Multimedia Laboratories, Yokosuka, Japan.

Digital Object Identifier 10.1109/TMM.2005.850974

feedbacks. While it is easy to conceive a variety of applications that can benefit from such delivery path information to improve performance, we focus in this paper on one optimization. It is a complexity-scalable automatic retransmission scheme that capitalizes on SA feedbacks in estimating the success delivery probability of transmitted packets.

The paper is organized as follows. We first discuss related work in Section II. We then discuss the implementation of SA in 3GPP-WCDMA in Section III. We then discuss the one application of SA in this paper: complexity-scalable retransmissions in Section IV. We present results of the SA application in Section V. Finally, we discuss related standardization efforts and conclude in Sections VI and VII, respectively.

II. Related Work

The idea of inserting agents at carefully chosen locations in the network is not new, and it has been reported in [12] to increase web traffic performance and in [13] to monitor network services. In contrast, our agent-based approach focuses on the delivery of delay-sensitive media content over 3G wireless networks. Prior research related to wireless media streaming are extensive. We divide related work into three sections. We first discuss work focusing on network protocols in Section II-A. We then discuss work optimizing media streams at the application level in Section II-B. We then position our work against contemporary work in Section II-C.

A. Related Work in Network Protocols

The focus of works on wireless data transport has been on optimizing TCP over last-hop wireless networks [14]–[16]. As an example, [14] proposed a Snoop protocol that improves the performance of TCP over a last-hop wireless link connection. In brief, it is a TCP-aware link layer protocol that caches and retransmits TCP packets and suppresses negative acknowledgments from senders. Since we are focusing on streaming media content that are highly delay sensitive, the unpredictable transmission delay and delay variance of TCP over wireless links mean TCP is not appropriate for our application scenario.

The IETF has been active in extending the current RTP specification [3] to enable better streaming quality, including proposals to extend RTCP to include timely feedbacks [17], and the use of forward error correction (FEC) to the standard RTP stream [18]. In related protocol development, HP labs Bristol has introduced UDP Lite [19] that offers more flexibility than the current UDP specification so that a packet with checksum errors will not be dropped automatically before being passed upward to the application. Our work is orthogonal to these developments, and our proposed streaming agent and associated optimizations can potentially be modified to work with these new protocol specifications.

B. Related Work in Media Optimization

An extensive body of previous research [6], [8], [7], [9], [20] designs optimized media transmission schemes assuming the entire delivery path is one packet independent channel. Our work differs in that we separate the channel into two parts: a wired network and a last-hop wireless link. Our work can also be viewed as an extension of [8] and [9]: we show how SA timely feedbacks—information along the delivery path—can be used to enhance end-to-end streaming performance in a rate-distortion sense.

Earlier works on media streaming optimization [21]–[25] for wireless links have focused on one or a few characteristics of the unique medium that differ from our work when optimizing media streams. The work in [21] assumed a packet loss model for the wired network and a bit error model for the wireless link and discussed an FEC scheme that offered packet-level and byte-level protection, respectively. In contrast, we assume a packet loss model for both the wired network and the wireless link as it is common in current 3G networks. The work in [22] focused exclusively on the wireless last-hop. They assumed low layer parameters such as frame/packet transition probabilities of a two-state Markov chain used to model the data link layer can be dynamically passed upwards to the streaming server at the application layer for joint error and power optimization. Our work differs from [22] in two aspects: 1) we assume the media stream needs to traverse through both a loss-prone wired network and a vulnerable wireless link and 2) we assume low layer parameters are not available to be passed to the optimizing streaming server. Reference [23] focused on the loss aspect of the 3G wireless link in designing a new transport layer protocol—Multimedia Streaming Protocol (MSP) — that implements forward error correction (FEC) in the transport layer to alleviate the burden of error control at the application layer. Instead, our streaming agent and associated optimizations derive benefits from the contrastingly different natures of both loss and delay of the wired network and the wireless link. The work in [24] and [25] performed rate-distortion optimized streaming using the same assumption as [21]—that packets can be dropped in the wired network and corrupted in the wireless link. As stated earlier, we assume a packet loss model for both wired network and wireless link.

C. Contemporary Work

The work in [26] and [27] employs a *proxy* at the wired/wireless boundary that caches media packets and intelligently requests packet retransmission from the streaming server after performing application-level rate-distortion optimized streaming procedures. In contrast, our *agent*-based approach is inherently a network level service provided by a wireless network operator, and hence has the following contrasting characteristics: 1) our agent-based approach does not require fine-grained feedbacks from the clients, hence it is 3GPP-PSS [2] compliant where clients send only RTCP statistical reports; 2) the agent does not interpret the payload of the media stream nor buffer any RTP packets, and hence it has much lower complexity overhead than a proxy; 3) it avoids security issues since payload can be encrypted without affecting operation correctness; 4) as a network service, the agent's feedbacks are useful for a hose of streaming applications at the server, one of which will be discussed in detail later; and 5) it retains the *soft state* property similar to the Snoop agent, where soft state is defined as the outage of a network element—temporary out-of-service due to equipment failure, etc.—will cause a tolerable degradation of performance instead of a catastrophic breakdown of

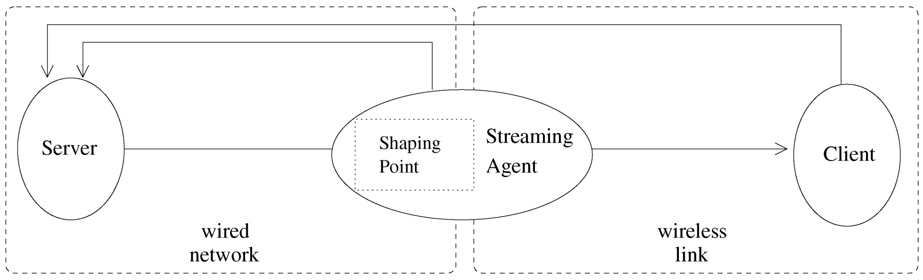


Fig. 1. Using streaming agent (SA) to provide timely feedback.

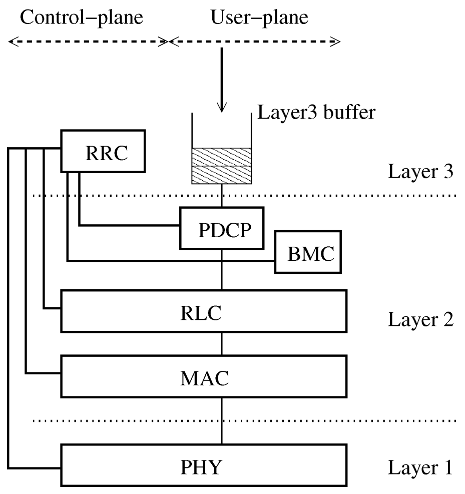


Fig. 2. IMT-2000 protocol stack.

the streaming session. Despite these differences, we will show in the results section that our light-weighted agent approach performs only marginally worse than the heavy-weighted proxy approach.

III. Streaming Agent

SA [11], an enhanced version of the RTP monitoring agent [5], is a network agent installed by the wireless network provider to provide network services to the server/client pair that are not possible with endpoints alone. It is located at the intersection of the wired core network and the transmitting wireless link (see Fig. 1 for an illustration). During a server-client RTP streaming session, SA identifies the stream by packet classification: look for matches at selected fields of RTP headers of incoming IP packets such as source and destination addresses and source and destination port numbers. SA periodically sends timely feedbacks (SA-FB) to the sending server in sub-second intervals, reporting the arrival status of the last K RTP packets of a stream. The frequency of SA-FB and the value K can be preset by the network operator, or by the sender prior to the start of the streaming session via a message exchange between the sender and SA.

In order not to overwhelm the wireless link, a *shaping point*, located just prior to SA, is used to limit the sending rate so that the packet rate is no larger than the wireless link bandwidth. Essentially, a layer-3 IP packet queue stores packets waiting to be fragmented and transmitted in lower layers (see Fig. 2 for an illustration). If the wireless link condition is poor, the number of retransmissions until successful transmission will be large, causing the IP packet queue to build up. The shaping point reacts to the fullness of the queue by pre-dropping packets that would have been dropped anyway due to eventual queue overflow. This way, packets that are dropped at the wireless link are dropped solely due to poor channel condition. Details of the shaping point are discussed in [5].

A. Using SA in 3GPP-WCDMA

We discuss here the implementation of SA in a wideband code division multiple access (WCDMA) communication system [1], one of the main 3G air interface technologies to be deployed in Europe and Asia, including Japan and Korea. Two possible locations for SA within a WCDMA system are the radio network controller (RNC), which is responsible for the control of the radio resources of the radio access network, and the transmitting base station (node B) (see Fig. 3 for an illustration). 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 a radio resource is properly managed by RNC, overload situations that cause wireless link congestion should be exceptional ([1, p. 213]). We will assume there is no wireless link congestion in this paper.

Several link layer transmission modes are available for WCDMA. As an IP packet is passed down from layer 3 to layer 2, the RLC provides segmentation and retransmission services (see Fig. 2 for an illustration). Whether and how many retransmissions are done depends on the RLC modes. There are three modes: transparent, unacknowledged and acknowledged. For acknowledged mode, an 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. Typical implementations use interleaving, channel codes and other techniques in lower layers to reduce raw packet loss rate to a reasonable level under normal conditions (less than 5%). If a small number of link-layer retransmissions is used in addition, then the resulting wireless link loss is very small.

SA-FB can be used to deduce wired network state—essential for the sender to perform proper congestion control. SA-FB can also be used to deduce wired application state (see Table I). Ideally, the server wants timely feedbacks directly from the client (C-FB) to reconstruct the *end application state*. In contrast, wired application state is at best an estimated end application state. So why are SA-FBs still desirable?

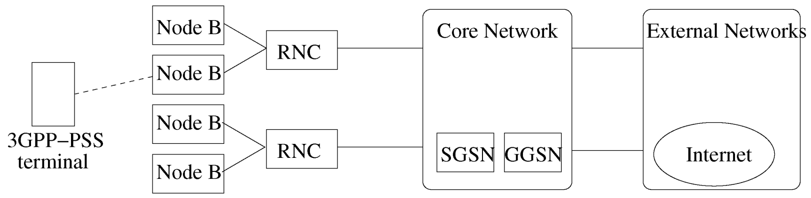


Fig. 3. 3GPP-WCDMA network elements.

First and foremost, the current 3GPP-PSS [2] specification has dictated the use of existing RTP and RTCP specifications [3] only. That means one can rely on compliant 3GPP-PSS handsets to provide only RTCP feedbacks—statistical feedbacks only. Second, even if mobile client can be modified to provide fine-grained feedbacks, it may have severe power constraint due to limited battery life. Having client send frequent feedbacks while receiving streaming video may not be desirable. Finally, if the intended streaming application can tolerate a fixed initial delay up to several seconds, we can elect to use acknowledgment mode as the wireless link layer transmission mode of choice to perform a small number of link-layer retransmissions. Doing so means the resulting wireless link loss is very small, and so wired application state closely mimics end application state, and therefore SA-FBs are essentially a much faster version of C-FBs, given the common hundreds of millisecond delay of the 3G wireless link.

IV. SA Application: Complexity-Scalable ARQ

A. Problem Formulation

We next focus on a scheme that employs SA for video streaming optimization. We call the scheme *complexity-scalable ARQ*. In short, we derive a rate-distortion optimized application-level retransmission scheme, leveraging on previous work [8], [9], that adapts to SA and possibly client feedbacks to optimize video quality. As an added feature, it has the property that the complexity of the optimization can be scalably tuned down at the cost of gracefully degrading streaming quality. We will discuss how this is done next.

The basic problem framework is the following. There is a predictively coded (IPPP…) video sequence with I-frame frequency L. At any given optimization instance, an optimization window equal to M-frame time is selected. The window is defined to be the set of frames whose delivery deadline (to be discussed) falls within start time $start(t)$ and end time $end(t)$. Frames are brought into the optimization window at time $start(t)$. Frames in the window expire at time $end(t)$ when they cannot reasonably be expected to be delivered to the client on time. The slope of both functions—the rate at which they advance in time—is the playback speed at the client.

We follow a standard congestion control protocol by adjusting the packet spacing in time, using the equation-based TCP-friendly congestion control in [4]

$$T_t = \mu_t\sqrt{\frac{2\epsilon_t}{3}} + 3(\mu_t + 4\phi_t)\epsilon_t(1+32\epsilon_t^2)\sqrt{\frac{3\epsilon_t}{8}} \qquad (1)$$

where ϵ_t, μ_t and ϕ_t are the updated estimates at time t of packet loss rate, mean round trip time and round trip time variance, respectively. Suppose the optimization can be run at the server no more frequently than every $P_{\min}$ seconds, we select the optimization period at time t, P_t, as

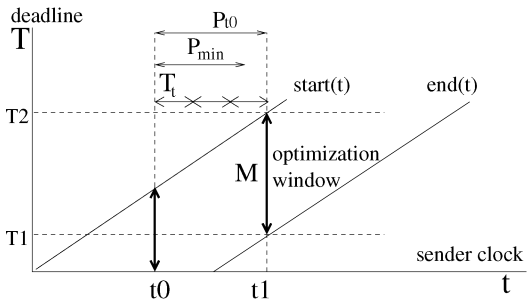


Fig. 4. Optimization window.

$$P_t = \left\lceil \frac{P_{\min}}{T_t} \right\rceil T_t. \qquad (2)$$

The number of packets that can be selected for transmission at optimization instance t—the bandwidth at time t, is $R_t = P_t/T_t$ (see Fig. 4 for a plot of frame deadline T against sender running time t).

Given the observed network conditions and ACKs from SA and possibly client, the scheme decides which frames within the optimization window to transmit and for how many times. To make the problem mathematically tractable, we first introduce simple source and network models as follows.

1) Source Model: We begin with a directed acyclic graph based source model to model a pre-encoded video sequence, similarly done in [9]. Each frame i is represented by one *data unit* (DU_i). Data unit is the smallest atomic unit considered during the optimization. Each DU_i is characterized by three numbers: delivery deadline T_i, size in number of RTP packets B_i, and reduction in distortion d_i. DU_i points to a set of DUs that DU_i is dependent on for correct decoding. Specifically, for a predictively coded (IPPP…) sequence, DU_i is correctly decoded iff each DU_j, $k \leq j \leq i$, is correctly delivered by the delivery deadline T_j to the client, where k is the last I-frame. If correctly decoded, DU_i reduces distortion at the client by d_i. Otherwise, DU_i reduces distortion by 0.

To calculate d_i for an I-frame and subsequent $L-1$ P-frames, we do the following. For d_1 of the first I-frame, we calculate the peak signal-to-noise ratio (PSNR) of encoded I-frame against the original frame 1, *plus* the PSNR of encoded I-frame against the original frame 2 through L. The reason is that in the event of a frame i decoding failure, $i > 1$, as a simple error concealment strategy we can display correctly decoded frame 1 during playback time of frame i. This resulting reduction in distortion needs to be accounted for in d_1. To calculate d_i of frame $i > 1$, we calculate the PSNR of encoded P-frame i against the original frame i, plus the PSNR of encoded P-frame i against the original frames $i+1$ through L as we did for the first frame, *minus* the PSNR of encoded frame $i-1$ against the original frame i through L. The reason for the last term is that if frame i is correctly decoded, that it should be more similar to future frame $j > i$ than frame $i-1$, then using frame i as display time of frame $j > i$ will cancel out the benefit of using frame $i-1$ for error concealment of frame j. If we let $d(i,j)$ be PSNR of encoded frame i against original frame j, then the above analysis can be summed up as follows:

$$\begin{aligned} d_1 &= d(1,1) + \sum_{j=2}^{L} d(1,j) \\ d_i &= d(i,i) + \sum_{j=i+1}^{L} d(i,j) - \sum_{j=i}^{L} d(i-1,j)\,, \quad i \geq 2. \end{aligned} \tag{3}$$

For the experiments performed in Section V, for each test sequence, a matrix $d(i,j)$ was loaded into the server to calculated d_i using (3), and into the client to perform the simple error concealment as stated above.

2) Network Model: While an independent time-invariant packet erasure channel with random delays is proposed in [9], we instead elect to use a simpler model for complexity reason. Essentially, the wired and wireless parts of the network are each modeled by a time-invariant independent packet erasure model with constant delay, where packet loss rates are α and β in the wired network and wireless link, respectively.

Let π_i be the number of transmissions for DU_i in the current optimization instance. Let $\phi_i = \{n_i, a_i, b_i\}$ be the history of DU_i in all previous optimization instances: DU_i's total number transmissions to-date, n_i, and the number of ACKs received from the SA and client, a_i and b_i, respectively. Further, let $\epsilon(\phi_i, \pi_i)$ be the probability that no transmission of DU_i is successful given ϕ_i and π_i. We have

$$\epsilon(\phi_i, \pi_i) = \begin{cases} 0 & \text{if } b_i > 0 \\ \beta^{a_i}[\alpha + (1-\alpha)\beta]^{n_i - a_i + \pi_i} & \text{o.w.} \end{cases} \tag{4}$$

3) Mathematical Formulation: Given a window of data units, the problem is to determine the optimal retransmission scheme for data units in the window. Like [9], the problem is formulated as a minimization of end-to-end distortion subject to a transmission rate constraint. The optimizing variable is set $\boldsymbol{\pi} = \{\pi_1 \ldots \pi_M\}$, where π_i is the number of times DU_i is transmitted in the window. By transmission policy then, we mean how many times each data unit is transmitted for this optimization period of duration P_t. The rate constraint R_t in number of packets has already been discussed in previous section. Mathematically, the problem is

$$\min_{\pi} D(\pi) \quad \text{s.t.} \quad R(\pi) \leq R_t. \tag{5}$$

The rate term $R(\pi)$ is simple and can be written as

$$R(\pi) = \sum_i B_i \pi_i. \tag{6}$$

Given the source model, the distortion term $D(\pi)$ is

$$D(\pi) = D_o - \sum_i d_i \prod_{j \preceq i} (1 - \epsilon(\phi_j, \pi_j)) \tag{7}$$

where D_o is the initial distortion if no frame is decoded correctly, and $\{j \preceq i\}$ denotes the set of data units DU_i depends on for correct decoding.

B. Dynamic Programming (DP) Solution

To solve (5), we employ a DP technique inspired by [8]. To simplify discussion, we assume for now that the size of the optimization window M is L, and we are optimizing a sequence of 1 I-frame plus $L-1$ dependent P-frames. We denote by $\Theta(k,\pi)$ the additional distortion reduction from frame k to L provided by policy vector π given the first $k-1$ frames are correctly decoded. We write

$$\Theta(k,\pi) = \sum_{i=k}^{L} d_i \prod_{j=k}^{i} (1 - \epsilon(\phi_j, \pi_j)). \tag{8}$$

Given $D(\pi) = D_o - \Theta(1,\pi)$, maximizing $\Theta(1,\pi)$ is equivalent to minimizing $D(\pi)$.

The key observation is that (8) can be written recursively as

$$\Theta(k,\pi) = (1 - \epsilon(\phi_k, \pi_k))\,(d_k + \Theta(k+1,\pi)). \tag{9}$$

The DP solution can be derived from (9) naturally by defining $\Theta^*(k,r)$ as the optimal $\Theta(k,\pi)$ given the rate budget for frame k to L is r packets. Assuming we bound the maximum number of transmissions for a given data unit in the current optimization window to be N, we have (10), as shown at the bottom of the page.

$\Theta^*(1,R_t)$ can now be solved recursively using (10). Each time (10) is solved, the solution is stored in the $[k,r]$ entry of a DP table. This is done so that repeated sub-problems are solved only once. The complexity of the algorithm can be bounded by the time required to construct the DP table. Given initial call of $\Theta^*(1,R_t)$, the size of the DP table is $L * R_t$. Each table entry is solved using (10), which is bounded by N. Hence, the complexity of the algorithm is $O(NLR_t)$.

1) Complexity Scaling: Like [8], a complexity of $O(NLR_t)$ means the algorithm is pseudopolynomial. In our case, it means a large running time for large R_t. To reduce running time, we can solve an approximate instance of the problem by performing a rounding operation by K for the sizes of data units B_i's and rate constraint R_t

$$B'_i = \left\lceil \frac{B_i}{K} \right\rceil \qquad R'_t = \left\lfloor \frac{R_t}{K} \right\rfloor. \tag{11}$$

$$\Theta^*(k,r) = \begin{cases} -\infty & \text{if } r < 0 \\ \max_{\pi_k=0}^{N} \{(1 - \epsilon(\phi_k,\pi_k))\,(d_k + \Theta^*(k+1, r - B_k\pi_k))\} & \text{o.w.} \end{cases} \tag{10}$$

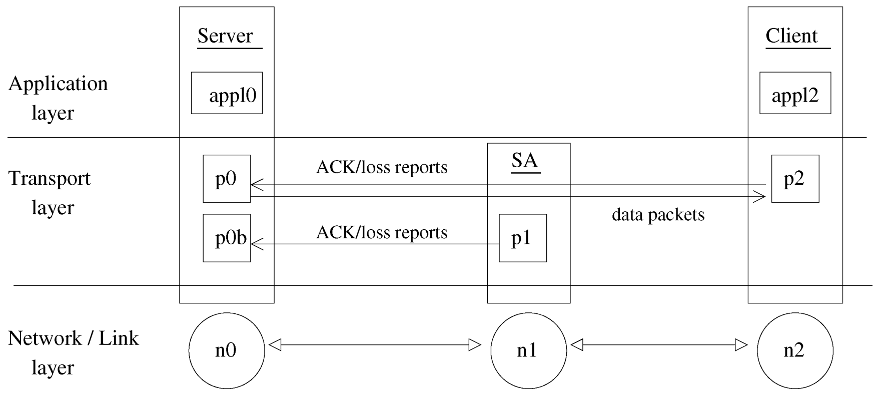


Fig. 5. Simulation setup.

We denote the solution to this optimization instance by π'. Because the size of the DP table has been scaled down by a factor of K due to smaller R'_t, the resulting complexity is $O(NL(R_t/K))$. The error bound from the optimal solution can be found *a posteriori* by first solving a different instance of the problem by performing the following rounding operation:

$$B''_i = \left\lfloor \frac{B_i}{K} \right\rfloor \qquad R''_t = \left\lceil \frac{R_t}{K} \right\rceil. \tag{12}$$

We denote the solution to this optimization instance by π''. The error of solution π' from optimal solution π^* can be bounded as follows:

$$|D(\pi') - D(\pi^*)| \leq |D(\pi') - D(\pi'')|. \tag{13}$$

The proof can be found in the Appendix. Notice that the *a posteriori* bound in (13) is similar to one for the traditional Lagrangian approach for resource allocation [28].

V. Experiments

A. Simulation Setup

We performed simulations using Network Simulator 2 [29]. The setup is shown in Fig. 5. Three nodes were constructed, $n0$, $n1$ and $n2$, representing the three locations of the server, SA and the wireless client, respectively. To connect these nodes, two links were constructed. Link n0-n1, simulating the wired network between the server and SA, had constant propagation delay and uniform loss rate α. For link n1-n2 that simulated the wireless link of rate 144 kbps, we implemented link-layer retransmission as follows: a network layer packet was fragmented and grouped into transport blocks of 180 bytes each that spanned 10 ms. Groups of x transport blocks were 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 was transmitted through the wireless link with success probability β. When link-layer retransmission was used, each transport unit was retransmitted when the sender timed out on the receiver's link-layer ACK. The maximum number of retransmissions was set at 20.

The transport layer had a duplex connection (p0-p2) from the server n0 to the client n2 and a simplex connection (p1-p0b) from SA n1 to the server. p0-p2 was for endpoint data transmission from server to client. p1-p0b was for feedbacks from SA to server; a filter was placed at link n1-n2 to sniff out packets targeted to the client and to forward them to p1, who then sent ACKs to the server.

A server application, app0, sat at sender node and sent packets to the client using the connection p0-p2. Each packet had a sequence number in the packet header indicating the frame it contained; we assume each frame maps to one RTP packet. For each video sequence, a distortion matrix $d(j,i)$ described in Section IV-A1 was generated offline and loaded into both server and client before playback began. Server derived distortion value d_i for each data unit i using (3), and client calculated the visual PSNR during each frame i playback time: if frame i is correctly decoded, then PSNR is $d(i,i)$; if not, then the most recently correctly decoded frame j was used for display for frame i, and we used $d(j,i)$ as PSNR. If no such frame j was available, then PSNR was 0.

B. SA Application: Complexity-Scalable ARQ

We will compare our proposed complexity-scalable ARQ scheme using SA with two schemes. The first is a rate-distortion optimized streaming scheme [9] based solely on client's RTCP feedbacks. We term this scheme *No Agt*. We will see soon that *No Agt* suffers severely from two shortcomings: 1) not having fine-grained client feedbacks means *No Agt* cannot accurately estimate success delivery probability of transmitted packets and 2) not being able to distinguish between wired and wireless loss and wired and wireless delays means *No Agt* performs unnecessary wired network congestion control, sending at rates much lower than necessary.

The second is a proxy scheme proposed in [26], [27] that performs a hybrid sender/receiver-driven rate-distortion optimized streaming optimization. For convenience, we term the proxy scheme *CCG* after the authors. In brief, *CCG* essentially performs a receiver-driven version of [9] at the proxy to request media packets from a streaming server, caches the packets, then performs a sender-driven version of [9] to transmit packets to the client. Recall that we have already pointed out the differences between our agent approach and the proxy approach, including security concern, complexity overhead and soft state property in Section II-C. Nevertheless, we will compare their performance in this section.

We use H.263 Version-2 video codec (TMN10) to encode two 300-frame MPEG test sequences, `container` and `foreman`.

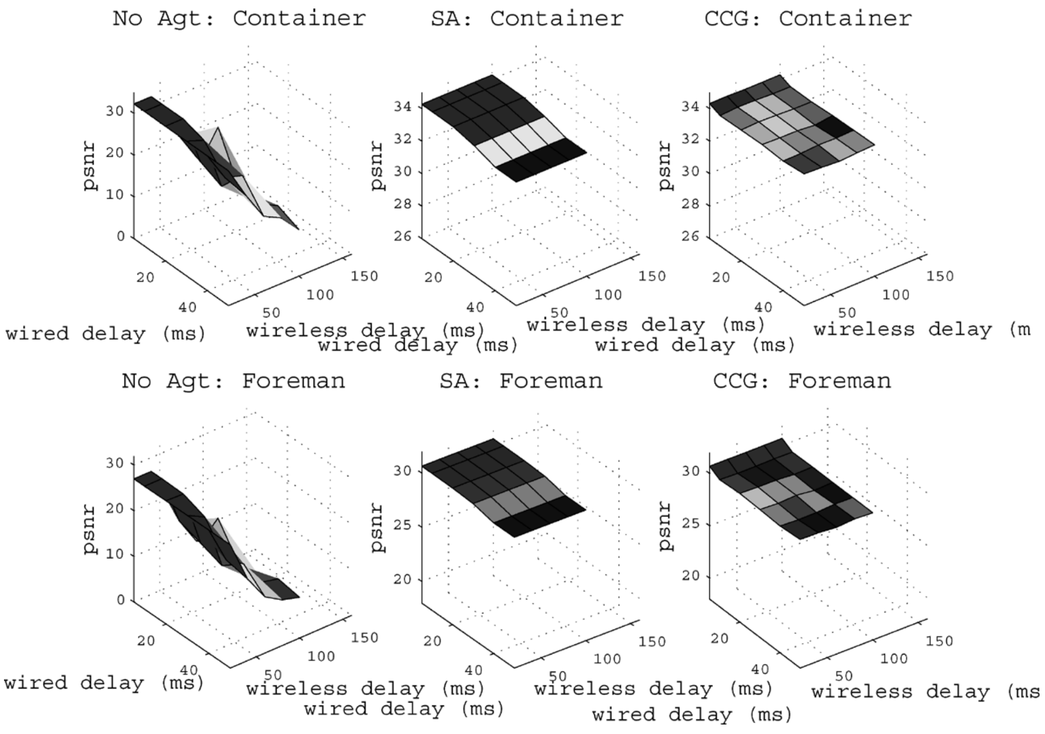


Fig. 6. Comparison of ARQ schemes for $\alpha = 0.03$.

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 PSNR[1] for the compressed streams under noiseless conditions are 38.49 and 32.46 dB, respectively. For each data point, the video sequence was replayed upon completion until 600-second playback time has been reached. This was done for an averaging effect. For the wireless link, acknowledgment mode is set up as described in Section V-A, with wireless loss rate fixed at 0.04.

For complexity-scalable ARQ using SA, termed *SA* below, we assume a 2-s delay between server start time and client playback time. The size of the optimization window M is 10-frame time. Optimization is performed every T_s seconds, i.e., we assume a sending rate R_t of 1 packet per optimization period. This means the procedure of rounding by a factor of K, discussed in Section IV-B1, is not necessary, as $R_t = 1$ is already the smallest integer possible. For *CCG*, we assume a 1-s buffering delay at the proxy, and a 1-s delay buffering delay at the client. This is roughly equivalent to the 2-s delay employed at the client for the *SA* case. The proxy also had an optimization window of ten frames, and it performed the same congestion control protocol (1) as *SA* as described in Section IV-A.

We varied the interleaving depth at the wireless link (wireless delay) as well as the wired network delay, and plotted Fig. 6 for the case when wired network packet loss $\alpha = 0.03$. We first see that by performing the proper congestion control, the performances of *SA* and *CCG* were far better than *No Agt* for any reasonable wireless delay value. We see that both *SA* and *CCG* are relatively flat with respect to wireless delay as compare to wired delay. The reasons are twofold: 1) link-layer retransmissions up to 20 times are already performed at the wireless link, hence the increase in wireless delay does not hamper retransmission performance and 2) in addition to affecting ARQ performance at the streaming server, increase in wired delay decreases the wired network bandwidth using congestion control protocol (1). Note however, that the performance of *SA* and *CCG* are remarkably similar, with the maximum performance differences between them being only 0.257 and 0.531 dB for the `foreman` and `container` sequences, respectively.

We performed the same experiment for the case when wired network packet loss rate $\alpha = 0.05$. The results are shown in Fig. 7. We see that similar trends can be observed. In this case, the maximum performance differences between *SA* and *CCG* are marginal: 0.279 and 0.295 dB for the `foreman` and `container` sequences sequences, respectively.

VI. Related Standardization Efforts

As the development of the streaming agent is engendered in the constrained environment of 3GPP-PSS, which is itself a collection of international standards, the many related evolving standards will have an important influence on the longevity of the applicability of the streaming agent. Given this is the case, it makes sense for us to discuss a few standard evolving trends here in this section.

The Internet Draft "Extended RTP Profile for RTCP-based Feedback" [17] discussed extensively how to extend existing RTCP protocol to include timely feedbacks. So it is conceivable that in the future version of 3GPP-PSS, the 3GPP-PSS compliant wireless client can and will send timely feedbacks to the streaming server. Nevertheless, the remaining reasons discussed in Section III-A still hold to support the existence and usefulness of SA, namely: 1) power consumption may still prevent client from sending frequent timely feedbacks to the server on

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

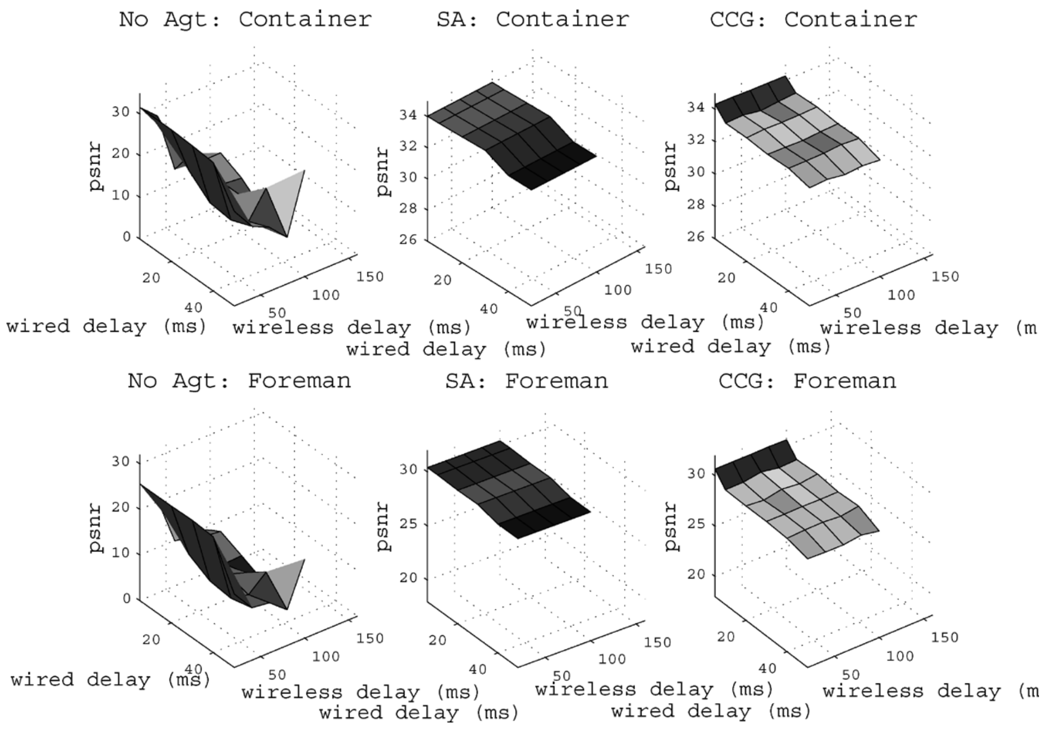


Fig. 7. Comparison of ARQ schemes for $\alpha = 0.05$.

a per packet basis and 2) if link-layer retransmissions are used to eliminate most of the wireless losses, SA feedbacks still represent a much faster yet accurate version of client feedbacks. Assuming the wireless delays remain in the hundreds of milliseconds in the near future, the timeliness of SA feedbacks over client feedbacks remains significant.

Also specified in [17] are the three types of feedback messages: transport layer, payload-specific, application layer. Both payload-specific and application layer messages require the feedback sender to look inside the RTP payload to determine if feedback is necessary. While the application at the client can provide these feedbacks, as a network agent, SA is required to look only at packet headers of IP packets, and hence SA does not have that knowledge and cannot provide these types of feedbacks. Nevertheless, timely packet-level transmission status remains useful to a hose of applications, and doing so without application-specific agent implementation modification. Note also, as we alerted earlier in Section II-C, that SA does sidestep any end-to-end security issues by never touching the payload. Hence, SA can be used even if the payload is encrypted as described in the secure RTP [30].

VII. Conclusion and Future Work

In this paper, we proposed the use of a Streaming Agent at the junction of the wired and wireless networks to provide additional timely feedbacks to the streaming servers. We discussed one specific streaming optimization, complexity-scalable application-level retransmission, that exploit such feedback to demonstrate potential benefits. Through simulation, it is shown that significant PSNR improvement can be maintained over nonagent-based systems.

For future work, extensions of the streaming agent that provide more services is possible. Given the network agent already monitors media flows, it is perhaps sensible for it to perform network policing, for example, to make sure it does not operate at a higher sending rate than it deserves. Information collected at the network agents can also be used for core network analysis for possible network-wide optimizations.

Appendix

In this section, we prove the error bound (13) of the approximate solution from the optimal solution using the approximation algorithm in Section IV-B. We denote the optimal solution to the original optimization problem (5) by π. Additionally, we denote the optimal solution to (5), with the rate of each data unit B_i and constraint parameter R_t rounded according to (11) and (12), by π' and π'' , respectively.

Note that π' is in the solution space of the original optimization (5) as well

$$\begin{aligned}\sum_{i=1}^{L} B_i' \pi_i' &\le R_t' \\ \sum_{i=1}^{L} \left\lceil \frac{B_i}{K} \right\rceil \pi_i' &\le \left\lfloor \frac{R_t}{K} \right\rfloor \\ \sum_{i=1}^{L} \frac{B_i}{K} \pi_i' &\le \frac{R_t}{K} \\ \sum_{i=1}^{L} B_i \pi_i' &\le R_t. \end{aligned} \tag{14}$$

Since the rounding does not affect the calculation of the objective measure, we know by the optimality of π that

$$D(\pi) \le D(\pi'). \tag{15}$$

Similarly, it can be shown that π is in the solution space of the optimization problem (5) with rate rounding (12). By optimality of π'', we have

$$D(\pi'') \leq D(\pi). \tag{16}$$

Combining (15) and (16), we can conclude

$$|D(\pi') - D(\pi)| \leq |D(\pi') - D(\pi'')| \tag{17}$$

and thereby established the error bound relating the approximate π' to the optimal π.

REFERENCES

[1] H. Holma and A. E. Toskala, *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications*: Wiley, 2001.

[2] 3GPP TS 26.233 Transparent End-to-End Packet Switched Streaming Services (PSS); General Description (Release 4) (2001, Mar.). [Online]. Available: ftp://ftp. 3gpp.org/Specs/2001-03/Rel-4/26_series/26 233-400.zip

[3] H. Schulzrine, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Application,", IETF RFC 1889, 1996.

[4] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Stockholm, Sweden, Aug. 2000.

[5] T. Yoshimura, T. Ohya, T. Kawahara, and M. Etoh, "Rate and robustness control with RTP monitoring agent for mobile multimedia streaming," in *Proc. IEEE Int. Conf. Communication*, New York, Apr. 2002.

[6] M. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for Layered Streaming Media," Univ. California, Berkeley, Tech. Rep. UCB/CSD-98-1024, 1998.

[7] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2000.

[8] V. Chande and N. Farvardin, "Progressive transmission of images over memoryless noisy channels," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 850–860, Jun. 2000.

[9] P. Chou and Z. Miao, ""Rate-Distortion Optimized Streaming of Packetized Media," Microsoft Research Technical Report,", Tech. Rep. MSR-TR-2001-35, 2001.

[10] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya, "Long Thin Networks,", IETF RFC 2757, 2000.

[11] G. Cheung and T. Yoshimura, "Streaming agent: A network proxy for media streaming in 3g wireless networks," in *Packet Video Workshop*, Pittsburgh, PA, May 2002.

[12] M. Margaritidis and G. Polyzos, "Mobiweb: Enabling adaptive continuous media applications over 3g wireless links," *IEEE Pers. Commun. Mag.*, vol. 5, no. 6, pp. 36–41, Dec. 2000.

[13] M. Gunter and T. Braun, "Internet service monitoring with mobile agents," *IEEE Network Mag.*, vol. 16, no. 3, pp. 22–29, May/Jun. 2002.

[14] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Networking*, vol. 5, no. 6, pp. 756–769, Dec. 1997.

[15] H. Inamura, G. Montenegro, R. Ludwig, A. Gurtov, and F. Khafizov. (2002) TCP over Second (2.5g) and Third (3g) Generation Wireless Networks. IETF. [Online]. Available: draft-ietf-pilc-2.5g3g-07

[16] B. Liu, D. Goeckel, and D. Towsley, "TCP-cognizant adaptive forward error correction in wireless networks," in *Proc. INFOCOM*, New York, NY, Jun. 2002.

[17] J. Ott, U. Bremen, S. Wenger, S. Fukunaga, N. Sato, K. Yano, A. Miyazaki, K. Hata, R. Hakenberg, and C. Burmeister. (2002) Extended RTP Profile for RTCP-Based Feedback. IETF. [Online]. Available: draft-ietf-avt-rtcp-feedback-02.txt

[18] J. Rosenberg and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction,", IETF RFC 2733, 1999.

[19] 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.

[20] P. Chou and A. Sehgal, "Rate-distortion optimized receiver-driven streaming over best-effor networks," in *Packet Video Workshop*, Pittsburg, PA, Apr. 2002.

[21] 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 *Proc. GLOBECOM*, San Antonio, TX, Nov. 2001.

[22] 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.

[23] H. Matsuoka, T. Yoshimura, and T. Ohya, "Design, implementation and performance measurement of multimedia streaming protocol (MSP)," in *Proc. Asian Int. Mobile Computing Conf. (AMOC'2002)*, May 2002.

[24] J. Chakareski and P. Chou, "Application layer error correction coding for rate-distortion optimized streaming to wireless clients," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 3, Orlando, FL, May 2002, pp. 2513–2516.

[25] J. Chakareski, P. Chou, and B. Aazhang, "Computing rate-distortion optimized policies for streaming media to wireless clients," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Apr. 2002, pp. 53–62.

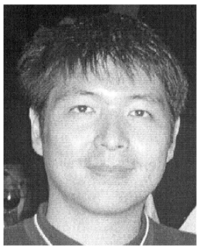
[26] J. Chakareski, P. Chou, and B. Girod, "Computing rate-distortion optimized policies for hybrid receiver/sender driven streaming of multimedia," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2002.

[27] ——, "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.

[28] Y. Shoham and A. Gersho, "Efficient bit allocation for an aibitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, Sep. 1988.

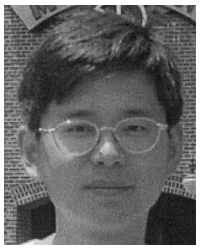
[29] (2001) The Network Simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[30] R. Blom, E. Carrara, D. McGrew, M. Naslund, K. Norrman, and D. Oran. (2001) The Secure Real Time Transport Protocol. IETF. [Online]. Available: draft-ietf-avt-srtp-01.txt



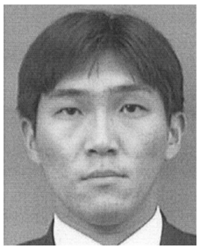
Gene Cheung (M'00) received the B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1998 and 2000, respectively.

Since August 2000, he has been a Member of Technical Staff at Hewlett-Packard Laboratories, Tokyo, Japan. 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 Hewlett-Packard Laboratories, 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.