RATE-DISTORTION OPTIMIZED APPLICATION-LEVEL RETRANSMISSION USING STREAMING AGENT FOR VIDEO STREAMING OVER 3G WIRELESS NETWORK

Gene Cheung *Wai-tian Tan* *Takeshi Yoshimura*

HP Labs, Japan
Hewlett-Packard Japan, Ltd

HP Labs, Palo Alto
Hewlett-Packard, Ltd

Multimedia Laboratories
NTT DoCoMo, Inc.

ABSTRACT

Feedback adaptation has been the basis for many media streaming schemes whereby the media being sent is adapted according to feedback information about the channel. 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 a 3G wireless network, 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 then show how the introduction of an 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.

1. INTRODUCTION

The goal of this paper is to improve streaming media quality in next generation 3G wireless networks [1]. In particular, streaming media in our context means a piece of media content is delivered from a streaming server in a wired network to a mobile client via a last-hop wireless link. The content is decoded and viewed by the client before the entire content has been downloaded, with possible setup and initial buffering delay. Media streaming is compliant with the 3GPP packet streaming service (3GPP-PSS) [2], where the server uses RTP for media transport while the clients send feedback to their respective servers via RTCP.

One common objective of media adaptation is congestion control whereby video sources reduce their transmission rates in reaction to wired network congestion [3]. For paths involving wired and wireless links, it has been shown [4] that end-to-end feedback information alone is ineffective for congestion control purposes since it is not possible to identify where losses occur. For instance, if losses occur in the wireless link due to poor wireless condition, it is not helpful if the sources 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 [4] — a network proxy located at the junction of the wired and wireless network sends *statistical feedbacks* (RTCP reports in particular) back to the sender to help the sender determine the proper action. However, the limited information contained by such statistical feedbacks are often insufficient for other purposes such as optimized retransmissions of this paper.

Another limitation of using only end-to-end feedback is the long time for the feedback to arrive. It turns out that in the case of today's 3G wireless network, typical one-way delay of radio links is quite large — on the order of 100ms — without link layer retransmission [5]. Thus, the actual end-to-end delay can be quite large, especially with wireless link-level retransmission. Such long delay severely impedes the effectiveness of feedback information for the purpose of congestion control and otherwise.

Both problems above can be solved simultaneously using a special agent called Streaming Agent (SA) [6] located at the junction of the wired network and wireless link. Unlike the RTP monitoring agent [4] which provides only statistical feedbacks such as average round trip time (RTT) and packet loss rate, SA sends *timely feedbacks*, such as acknowledgment packets (ACKs), that tell the sender which packet has/has not arrived at SA correctly and on time. We call such information provided by SA the *wired client state*. 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 be taken faster to alleviate network congestion. Furthermore, by providing the wired client state rather than wired network state induced from statistical feedbacks, SA allows senders to have much more flexibility in media adaptation than is possible with wired network state alone. In this paper, we introduce a specific application retransmission scheme which explicitly exploits the availability of client states provided by SA and demonstrate the potential gains.

The organization of the paper as follows. We first discuss related work in section 2. We then discuss the details of SA in section 3. We briefly discuss the transmission scheme under investigation in section 4. Finally, we present results in section 5.

2. RELATED WORK

SA and RTP monitoring agents [4] are related to the Snoop agents [7] in that they can be used to properly perform congestion control for one-hop wireless networks. However, [7] focused on providing transparency to TCP end-points, while the role of the RTP monitoring agent is to provide enough information to the sender to perform congestion control. SA provides further information — the wired client states, that allow the senders to perform other media adaptations such as optimized retransmission.

Streaming media is a popular topic, and an extensive body of previous research [8] [9] [10] exists focusing on the case when the entire delivery path is abstracted as one packet independent channel. Our work can be viewed as an extension of [9] and [10]: we show how using SA timely feedbacks, end-to-end streaming performance can be increased in a rate-distortion sense.

The general notion of timely feedbacks has long been known to be useful in the streaming literature, and many transmission schemes [8] [10] rely on the availability of such feedbacks. Re-

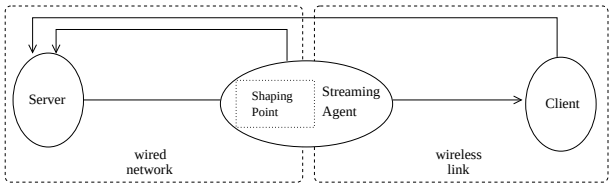

Fig. 1. *Using Streaming Agent (SA) to provide timely feedback.*

cently, the Audio/Video Transport Working Group in IETF has also recognized the importance of timely feedbacks and submitted an Internet draft [11] to modify RTCP reports in current RTP RFC to include such feedbacks.

3. STREAMING AGENT

SA [6] is an enhanced version of the RTP monitoring agent in [4]. It is a network proxy located at the basestation of the transmitting wireless link, or more generally, at the intersection of the wired core network and the transmitting wireless link. During a server-client streaming session such as an RTP session, SA monitors and identifies the stream by examining the RTP headers. SA then periodically sends statistical and timely feedbacks to the sending server. Specifically, statistical feedbacks in the form of RTCP reports are sent in intervals of seconds to minutes, and contain observable statistics such as mean RTT and packet loss rate. Timely feedbacks are sent in sub-second intervals and contain packet acknowledgments or even user-defined event notifications. The exact frequencies of timely and statistical feedbacks can be preset by the network operator, or by the sender prior to the start of the streaming session via other handshake. See Figure 1 for an illustration.

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 is located at the basestation to store packets waiting to be fragmented and transmitted in lower layers. 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 dropping packets prior to packet arrival at SA to avoid eventual queue overflow. Details of the shaping point is discussed in [4].

3.1. Usefulness of SA Feedbacks

SA statistical feedbacks help sender track wired network state — essential for the sender to perform proper congestion control. On the other hand, SA timely feedbacks help sender track wired client state. For example, SA can monitor the RTP sequence number space in the stream and send ACKs when it notices a group of K consecutive packet arrivals.

Ideally, the server wants timely feedback directly from the client to reconstruct the *end client state* — whether each packet has arrived at client correctly and on time. In contrast, wired client state induced from SA timely feedbacks leads only to an estimated end client state at best. Nevertheless, SA feedbacks are desirable for a number of reasons.

First, the current 3GPP-PSS [2] specification has dictated the use of existing RTP and RTCP specifications only. That means one

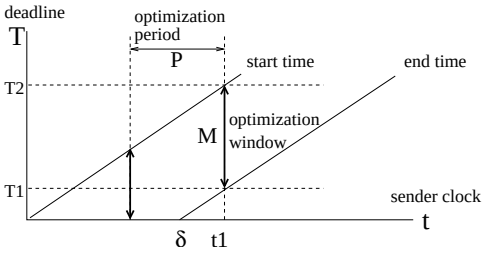

Fig. 2. *Optimization Window*

can rely on 3G handsets to provide only RTCP feedbacks — statistical feedbacks only. Second, mobile client typically has severe power constraint due to limited battery life. Having client sending frequent feedbacks while receiving streaming video may not be desirable.

Even if the client can send timely feedbacks, SA timely feedbacks can still be useful. The reason is that sender can track only a delayed version of the end client state using client timely feedbacks, where the delay is RTT between sender and client. Similarly, the sender tracks a delayed version of the wired client state where the delay is RTT between sender and SA. In the case of today's 3G wireless networks, the wireless link delay is large [5] — on the order of 100ms — compared to the wired network delay. An approximate and recent client state is often more useful than one that is accurate and much delayed.

Moreover, quality of wireless link can be traded off with link delay by setting the number of link-layer retransmissions during session setup [1]. When this parameter is set high, the wireless link becomes low loss and high delay. Using a large parameter in combination with SA [6], the tractable wired client state mimics the true end client state well and is also much faster than the tractable end client state.

4. APPLICATION-LEVEL RETRANSMISSION

4.1. Problem Formulation

To effectively exploit the usefulness of SA timely feedbacks, we derive a rate-distortion optimized application-level retransmission scheme, leveraging on previous work [9] [10], that adapts to SA and client feedbacks. 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 data-units whose delivery deadline (to be discussed) falls within start time $start(t)$ and end time $end(t)$. $start(t)$ brings data-units into the optimization window; by keeping the window small, it keeps the optimization computationally feasible and the instantaneous client buffer small. $end(t)$ expires data-units when they cannot be 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. The optimization is performed every P seconds. See Figure 2 for a plot of data-unit deadline T against sender running time t.

Given the observable network conditions and acknowledgment packets from streaming agent and 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 network and source models.

4.1.1. Network Model

We model the wired part of the network independently from the wireless link. The packet loss experienced by the client will be a combination of the wired and wireless part. We will assume the two parts are time-invariant, memoryless and independent from each other. The two loss probabilities associated with the wired and wireless forward trips are α and β, respectively.

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

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

4.1.2. Source Model

We use the directed acyclic graph based source model introduced in [10] to model our pre-encoded video sequence. Each frame i is abstractly represented by one data-unit (DU_i). For simplicity, we assume a one-to-one mapping of data-unit to RTP packet. Each DU_i is characterized by three numbers: delivery deadline T_i, size in byte B_i and reduction in distortion d_i. DU_i is correctly decoded iff all DU_j, $k \leq j \leq i$ are 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.

4.1.3. Mathematical Formulation

The problem is given a window of data-units, what retransmission scheme to use to transmit the data-units in the window. We derive the optimal retransmission scheme similar to [10]. The scheme is formulated as a minimization of end-to-end distortion subject to a transmission rate constraint. The optimizing variable is $\pi = \{\pi_1 \ldots \pi_M\}$, where π_i is the number of times DU_i is transmitted in an optimization window. By transmission policy, we mean how many times the data-unit is transmitted within a transmission window. The transmission rate constraint R_t can be set using one of several standard congestion control algorithms such as [3]. Mathematically, the optimization is then:

$$\min_{\pi} D(\pi) \quad \text{s.t.} \quad R(\pi) \leq R_t \quad (2)$$

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

$$R(\pi) = \sum_i B_i \pi_i \quad (3)$$

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

$$D(\pi) = D_o - \sum_i d_i \prod_{j \preceq i} (1 - \epsilon(\phi_j, \pi_j)) \quad (4)$$

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.

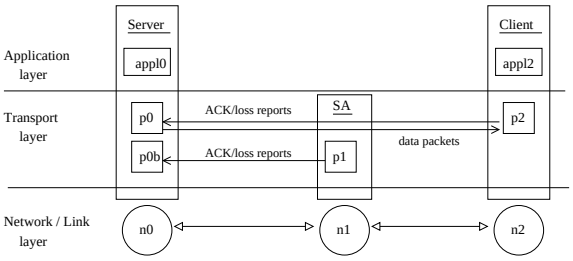

Fig. 3. *Simulation Setup.*

4.2. Solution

To solve (2), we employ a dynamic programming technique inspired by [9]. To simplify the discussion, we will assume for now that the size of the optimization window M is L, and we are optimizing a sequence of 1 I-frame plus dependent $L-1$ P-frames. We first define $\Theta(k, \pi)$, the additional distortion reduction from frame k to L provided by policy vector π given the first $k-1$ frames are correctly decoded. It can be written as:

$$\Theta(k, \pi) = \sum_{i=k}^{L} d_i \prod_{j=k}^{i} (1 - \epsilon(\pi_j)) \quad (5)$$

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

The key observation is that (5) can be written recursively:

$$\Theta(k, \pi) = (1 - \epsilon(\phi_k, \pi_k))\,(d_k + \Theta(k+1, \pi)) \quad (6)$$

The dynamic programming solution can be derived from (6) naturally by defining $\Theta^*(k, r)$ as the optimal $\Theta(k, \pi)$ given the rate for frame k to L is r. Assuming we bound the maximum number of transmissions for a given packet to be N, we have:

$$\Theta^*(k, r) = \begin{cases} -\infty & \text{if } r < 0 \\ \max_{\pi_k = 0}^{N} \{(1 - \epsilon(\phi_k, \pi_k)) \\ \quad (d_k + \Theta^*(k+1, r - B_k \pi_k))\} & \text{o.w.} \end{cases} \quad (7)$$

$\Theta^*(1, R_t)$ can now be solved recursively using (7). To reduce complexity, each time (7) is solved, the solution is stored in the (k, r) entry of a dynamic programming table. This is such that repeated sub-problems are solved only once. The complexity of the algorithm is $O(NR_t^2)$.

Like [9], $O(NR_t^2)$ means the algorithm is pseudo-polynomial. In our case, it means a large running time for large R_t. To reduce the running time, we assume packets of data-units are approximately of equal size, and we express rate limit R_t in number of packets instead of bytes. Correspondingly, we replace $B_i \pi_i$ and $B_k \pi_k$ in (3) and (7) with π_i and π_k, respectively.

5. RESULTS

5.1. Simulation Setup

We performed simulations using Network Simulator 2 [12]. The simulation setup is shown in Fig. 3. It has a transport layer duplex connection (`p0-p2`) from the sender node `n0` to the client node `n2`, and a simplex connection (`p1-p0b`) from the SA node `n1` to the sender node `n0`.

In our simulation, the links `n0-n1` and `n1-n2` have constant delay and uniform loss rates. An instance of the application, `app0`,

sits at sender node and sends packets to the client using the first connection. Each packet has a sequence number in the packet header. There is a filter at the link from `n1` to `n2` that sniffs out the packets targeted to the client and forwards it to `p1`, who sends ACKs back to the sender using the second connection. The sender performs the optimization above based on received ACKs.

5.2. Results

The objective measure we are using is average PSNR, calculated relative to the uncompressed original video sequence. When the receiver is unable to decode frame i, the most recently correctly decoded frame j is used for display for frame i, and so we calculate the PSNR using original frame i and encoded frame j instead. If no such frame j is available, then PSNR is 0.

For real video data, we use H.263 video codec to encode the first 50 frames of the `carphone` sequence into a video stream, encoded at QCIF size, 230kps, 20frames/s and at I-frame frequency of 25 frames. The resulting average PSNR for the compressed stream is 37.01dB.

We assume an 1s delay between server start time and client playback time. The size of the optimization window M is 10-frame time or 500ms. Optimization is performed every 5-frame time or 250ms. For each optimization instance, we assume a sending rate R_t of 6 packets per optimization period.

We evaluate our proposed solution by comparing three different schemes: *no-FB* where no feedback is used, *ClientFB* where only client feedback is used, and *ClientFB + AgentFB* where both client and agent feedbacks are used. Fig. 4 shows the achieved PSNR of the three schemes at different wired and wireless loss rates, at wired and wireless delays of $30ms$ and $200ms$ respectively. Due to the high delay associated with end-to-end feedback, we see that client feedback alone is ineffective, resulting in negligible gain over the *no-FB* scheme. In contrast, the additional use of agent feedback can provide significant quality improvement. Specifically, at low wireless loss rates of 0-1%, the use of agent feedback improve PSNR by 1-2dB over the wide range of wired loss of 1-10%.

Fig. 5 shows the PSNR achieved by the three schemes at different wired and wireless delay when the loss rates at the wired and wireless networks are 5 and 1%, respectively. Again, we see that client feedback alone is effective over the wide range of wired delay between 0 to 50ms, and wireless delay between 100 to 300ms. In contrast, the additional use of agent feedback sustained a PSNR improvement of 1 to 2dB over the same range. As expected, the PSNR improvement decreases as the wired delay increases. Nevertheless, at a relatively high wired delay of 50 ms and relatively low wireless wireless delay of 100 ms, a PSNR difference of 1.28dB is still maintained.

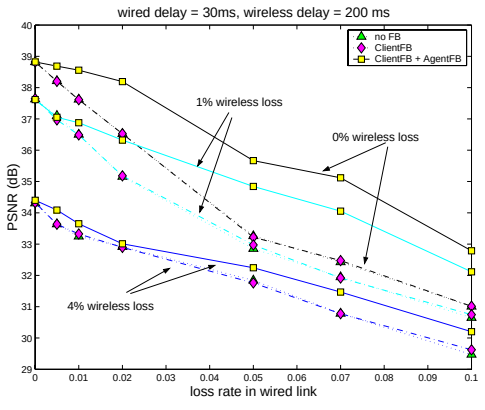

Fig. 4. *Comparison of various feedback schemes at different wired and wireless loss rates.*

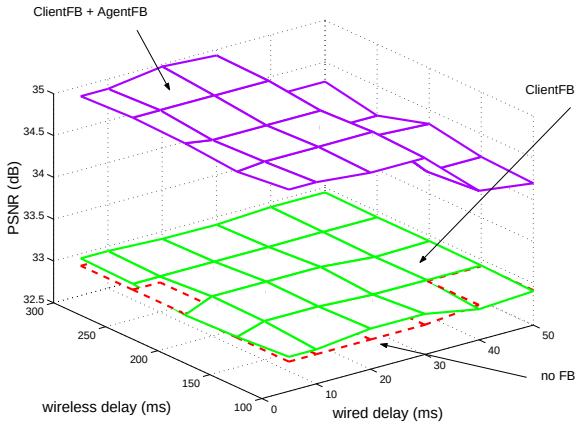

Fig. 5. *Comparison of various feedback schemes at different wired and wireless delay. Wired loss is 5% and wireless loss is 1%.*

6. CONCLUSION

In this paper, we proposed the use of a Streaming Agent at the junction of the wired and wireless networks to provide additional, and timely feedback to the servers. A specific optimized streaming scheme exploiting such feedback is presented to demonstrate potential benefits. Through simulation, it is shown that significant PSNR improvement of 1 to 2 dB can be maintained over a range of operating conditions.

7. REFERENCES

[1] H. Holma and A. Toskala, Eds., *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)*, ftp://ftp.3gpp.org/Specs/2001-03/Rel-4/26_series/26233-400.zip, March 2001.

[3] S. Floyd et. al., "Equation-based congestion control for unicast applications," in *SIGCOMM*, 2000.

[4] T. Yoshimura et. al., "Rate and robustness control with rtp monitoring agent for mobile multimedia streaming," in *IEEE ICC 2002*, April 2002.

[5] G. Montenegro et. al., "Long thin networks," January 2000, IETF RFC 2757.

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

[7] H. Balakrishnan et. al., "A comparison of mechanisms for improving tcp performance over wireless links," in *IEEE/ACM Trans. Networking*, December 1997, vol. 5, no.6.

[8] M. Podolsky, S. McCanne, and M. Vetterli, "Soft arq for layered streaming media," Tech. Rep. UCB/CSD-98-1024, University of California, Berkeley, November 1998.

[9] V. Chande and N. Farvardin, "Progressive transmission of images over memoryless noisy channels," in *IEEE JSAC*, June 2000, vol. 18, no.6.

[10] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," in *submitted to IEEE Trans. MM*, February 2001.

[11] S. Wenger et. al., "Extended rtp profile for rtcp-based feedback," July 2001, IETF Internet-draft: draft-ietf-avt-rtcp-feedback-00.txt.

[12] "The network simulator ns-2," June 2001, release 2.1b8a, http://www.isi.edu/nsnam/ns/.