VIDEOCONFERENCING OVER AN INTERMEDIATE-PROXY

Ali C. Begen and Yucel Altunbasak

School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA USA

{acbegen, yucel}@ece.gatech.edu

ABSTRACT

While the recent developments in access technologies such as DSL and cable enable end-users to communicate with each other in the means of video and voice, there still exist several hurdles in the sustainability and reliability of Internet communication services. In this paper, we focus on improving the performance of the delivery solutions for interactive media in the best-effort Internet. A promising approach in satisfying the stringent delay/loss requirements of interactive media transmission is to benefit from configurable proxies. In this study, we introduce an intermediate-proxy solution for videoconferencing applications running over the networks with large delays. By the Internet experiments between the U.S. and Europe, we demonstrate the effectiveness and potential benefits of the proposed approach.

1. INTRODUCTION

The primary role of ubiquitous networking, in particular of the Internet, is to disseminate the information in a timely manner and provide an inexpensive communication platform to its users. As the access technologies provide high bandwidths at economically-attractive prices and by the help of the advancements in audiovisual signal processing, a larger number of people are communicating interactively through networks every day. However, there still exist hurdles in terms of sustainability and reliability in these communication services, particularly in the applications that require strict interactivity.

As several research studies previously reported, effective audiovisual communication demands appropriate media-aware and application-specific solutions, without which its full benefits will not be realized. Poor approaches often lead to a set of associated system performance implications: degraded performance as perceived by the clients, possible network collapses due to high-bandwidth nature of video applications and poor performance observed by other network flows due to the unresponsiveness of such applications. Yet, intelligent approaches still cannot guarantee to function properly and deliver the desired quality because of the inevitable impairments of the best-effort Internet.

As an illustrative example, let us consider a videoconferencing session running between two end-users. In videoconferencing applications, packets that are not delivered within 200 ms usually disturb the video and result in intermittent and unintelligible video. An empirical study on RealVideo [1] reports that modem users can experience a jitter of 200 ms (or more) with a probability of 50%. That is, even if we ignore the propagation delays, the large delay variations will hinder the interactivity resulting in choppy frames in half of the session. For the broadband subscribers such as DSL, this probability reduces to 20%, but still has a high value. Although these discouraging results were obtained by experiments that were conducted by users who were diversely located around the world, they are representative to characterize a videoconferencing session running between end-users residing on different continents. A more recent study [2] that analyzed the real-time video streaming experience of dial-up modem users, also shows that these users experience round-trip delays larger than 600 ms on the average. Combined with the high packet loss rates, such large delays unfortunately render interactive applications, and even real-time streaming applications, impractical for a large group of Internet users.

Evidently, large delay variation reduces the chance of delivering the packets successfully before their decoding deadlines. In addition, it becomes more difficult for the receiver to exactly know if a missing packet is actually lost or excessively delayed. Although the receiver can try to infer this information by observing the packet arrival times [3], under a strict delay tolerance, deciding on a retransmission for a missing packet is a challenging problem. A retransmission request at an early stage can be redundant, whereas a late attempt will probably be useless. Naturally, as the delay tolerance increases or the one-way delay decreases, the receiver will likely give more accurate and well-timed retransmission decisions, which probabilistically increase the chance of timely delivery of the packet and improve the expected video quality. In their work [4], Chou and Miao address this problem and try to solve it by using a Markov decision process framework. However, under a delay tolerance of 200 ms, a packet can almost never find a retransmission opportunity if one-way delay and delay variation are already large.

Delay is a feature of the underlying physical network. Hence, we cannot do much to reduce it on the end-to-end basis. However, we can virtually eliminate the adverse effects of large delays by introducing an *intermediate-proxy* (I-Proxy) on the path between the end-users. Basically, the I-Proxy acts as a base, from which the missing packets reported by the receiver can be retransmitted without waiting for the retransmission from the sender. Likewise, if the I-Proxy does not receive a packet within a pre-defined time period, it can request a retransmission from the sender, eliminating the delays that would be incurred in the case of a receiver-initiated retransmission request. In other words, the I-Proxy divides the network with a large end-to-end delay into two networks with shorter end-to-end delays. This way, one can enable a retransmission opportunity for the packets, which would not have that chance otherwise.

In the literature, proxies have been long considered for Web-

0-7803-8554-3/04/$20.00 ©2004 IEEE.

caching to reduce the network traffic and improve the latency observed by the end-users. More recently their use is also considered for multimedia streaming applications [5, 6]. In their work [7], Hartanto *et al* study proxy-caching strategies for continuous media in interactive streaming applications. In particular, they develop caching strategies based on the request patterns for the proxies connected to the clients via LAN. In [8], the authors study the problem of minimizing bandwidth consumption by jointly optimizing server scheduling and caching strategies. All these studies utilize proxies for caching and reducing the network load. However, in this paper our focus is on using proxies in order to provide better error-protection for time-critical media content.

2. THE APPROACH

Consider the system configuration depicted in Fig. 1, where there are two end-users, denoted by client A and B, and an intermediate-proxy (I-Proxy). Clients A and B capture and encode the video in real-time. After packetizing the video, they transmit the packets to the I-Proxy and the I-Proxy forwards them to the other end. Meanwhile, the I-Proxy also caches any forwarded packet for a short amount of time. For the sake of clarity, let us focus on the video transmission from client A (sender) to client B (receiver).

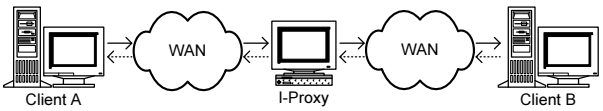

Fig. 1. Illustration of videoconferencing over an I-Proxy.

As mentioned previously, by observing the packet arrival times the receiver can request a retransmission for the packets that are late for a certain amount of time. Since this is a two-way videoconferencing session, any retransmission request is piggybacked on the video packets transmitted on the reverse direction as a feedback message. When the I-Proxy receives a retransmission request and if it has the packet in its cache, *i.e.*, the request is a hit, it immediately retransmits the requested packet to the receiver and clears the retransmission request from the packet. The explicit advantage of this *early retransmission* is the savings in the recovery time for the missing packet; it avoids unnecessary delays that would be incurred in case of a retransmission by the sender. Less explicitly, the I-Proxy also reduces the amount of network resources consumed during the retransmission. Particularly, this approach is useful when the packets often get lost between the I-Proxy and the receiver, or the round-trip delay between the I-Proxy and the receiver is relatively small.

There might be some packets that get lost between the sender and the I-Proxy as well. Eventually, the receiver will observe these missing packets and ask for a retransmission if enough time remains to their decoding deadlines. However, similar to the receiver, by observing the packet arrival times the I-Proxy can infer these lost packets earlier than the receiver. In such a situation, the I-Proxy requests a retransmission from the sender. The sender then retransmits the missing packet and the I-Proxy forwards it to the receiver. This proactive approach, so-called *fast retransmission*, avoids unnecessary waiting for the receiver to observe the missing packets and request a retransmission. In other words, by the help of the I-Proxy, the sender is informed about the missing packets at an earlier stage, which allows us to take the necessary actions on time. Recall that without an I-Proxy we would have to wait for the receiver to report a missing packet and by that time, most probably it would be late for a retransmission attempt because of the insufficient remaining time to the decoding deadline.

Having given an overview of our approach, we next discuss the details of our implementation.

3. IMPLEMENTATION ISSUES

Throughout our presentation, to make the analysis tractable we will use a simple terminology and notation. We associate three properties with each video packet. For a packet l, we denote its decoding deadline, arrival time to the I-Proxy and arrival time to the receiver by $t_{DTS,l}$, a_l^P and a_l^R, respectively. Depending on the video rate and packet size, the sender transmits the video packets equally-spaced in time. We denote this inter-packet spacing by ΔT in ms.

3.1. Retransmission Request Policy:

In a jitter-free transmission medium, equally-spaced packets are expected to arrive equally-spaced at the destination. However, it is the jitter what disturbs the inter-packet spacing, and hence, renders the retransmission decisions difficult. A retransmission request is initiated when a packet does not arrive within a certain amount of time. This certain amount of time is referred to as the retransmission timeout (RTO) and is estimated by observing the consecutive packet arrivals and delay variation. RTO plays an important role in deciding on whether the missing packet is lost or delayed.

In [9], the authors studied the performance of several RTO estimators on a real-time streaming application running between a video server and dial-up modem users. However, the nature of their application is quite different from ours, *e.g.*, while they employ a one-way streaming application with a startup delay of 2.7 seconds, our application is an interactive one and requires the delivery of the packets in 200 ms. This usually leaves us with the time enough for only one or at most two retransmissions. Hence, our RTO estimator should be more aggressive in estimating RTO. At the same time, it should also be as network-friendly as possible in order not to disturb the transmission of new packets and other flows in the network.

In parallel to the results reported in [9], our observations on the actual Internet packet traces show that the latest forward-trip time (FTT) sample is an important parameter in estimating the subsequent forward-trip delay. Our traces also indicate that the inter-packet spacings (observed at the destination) generally tend to increase when the congestion gets aggravated and tend to decrease as the congestion is alleviated. This suggests us to monitor the variation in the inter-packet spacing. We estimate the expected value for the next inter-packet spacing, $\Delta T_n(l+1)$, based on the current observation, $\Delta T_c(l)$. Specifically, we use the following estimator:

$$\Delta T_n(l+1) = \alpha \times \Delta T_c(l) + \beta \times \Delta T, \quad (1)$$

where

$$\Delta T_c(l) = a_l^{P,R} - a_{l-1}^{P,R}. \quad (2)$$

In (1), α and β are some constants used to adjust the tolerance of the estimator to the jitter. Implicitly, they determine the trade-off between the redundant and late retransmissions.

The retransmission request policy works as follows: After a packet arrival, if the subsequent packet does not arrive in ΔT_n ms, the missing packet is identified as lost and a retransmission

is requested. However, if the packet is eventually received, there will be an ambiguity whether it is due to the first transmission or the retransmission. To circumvent this ambiguity, we do not use the arrival times of the retransmitted packets in our RTO estimator. Furthermore, in contrast to TCP, which increases its RTO value in case of a timeout, we keep the value of ΔT_n unchanged until a single-transmitted packet arrives, in which case we recompute its value.

The retransmission request policy explained above is employed by both the receiver and the I-Proxy with possibly different values for α and β. In this paper, we will not elaborate on finding the optimal values of α and β. Interested readers are referred to [9, 10] for a further discussion on RTO estimators.

3.2. Caching Strategy:

As previously discussed in Section 2, video packets are transmitted in both directions via the I-Proxy. During this transmission, the I-Proxy caches any received packet for a possible retransmission. However, if the practicability of the system is considered, it is clear that the idea of caching each received packet does not scale well. That is, the cached packets should be replaced by the new ones in order to limit the storage space required by each video session. Recall that our target application is videoconferencing, in which packets have very short lifetimes. Any packet becomes essentially useless and can be removed from the cache when the display time of the last frame that depends on this packet (usually the last frame in a GOP) passes. For example, if the transmitted video has a GOP size of ten frames and the frame rate is 20 f/s, then the critical time to hold the packets in the cache is at most 500 ms. Subsequently, the storage requirement for one-way transmission can be computed by $0.5 \times R$ (Kbit), where R is the video rate in Kbps. Considering that the typical video rate varies between 40 - 512 Kbps, we can calculate the storage cost for a single session (two-way video transmission) as 5 - 64 KB. Since these storage requirements are far smaller compared to the storage capacity of a conventional proxy, several sessions can be run concurrently through a single I-Proxy.

4. EXPERIMENTAL RESULTS

In this section, we present experimental results to evaluate the performance improvements gained in a delay-sensitive application by employing an I-Proxy. For this purpose, we ran a videoconferencing session of 30 minutes between a client connected to Georgia Tech network and a broadband client connected to an ISP in Istanbul, Turkey. The I-Proxy was located in Bilkent University in Ankara, Turkey. The corresponding configuration is depicted in Fig. 2. In the rest of the section, for clarity we will refer to the clients as C_{GT} and C_{IST}, respectively.

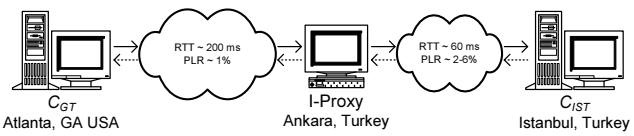

Fig. 2. Experimental setup.

Before running a videoconferencing session, we first conducted initial experiments to measure the path characteristics such as packet loss rate, delay and jitter. These experiments were run between each client and the I-Proxy. We measured low packet loss rate (around 1.0%) and almost constant FTT (around 100 ms), *i.e.*, low jitter, between C_{GT} and the I-Proxy, whereas we observed higher packet loss rates (2-6%) and jitter between C_{IST} and the I-Proxy although C_{IST} was both physically and network-wise closer to the I-Proxy (mean FTT is 30 ms).[1] We suspect that this result is mainly because of the better connectivity of both Georgia Tech and Bilkent campuses. The corresponding distributions of round-trip time (RTT) samples are given in Fig. 3. Note that our experimental setup favors the I-Proxy approach. However, in some less-favorable cases it may not be easy to determine the ideal location for the I-Proxy. We will investigate this problem further in our future work.

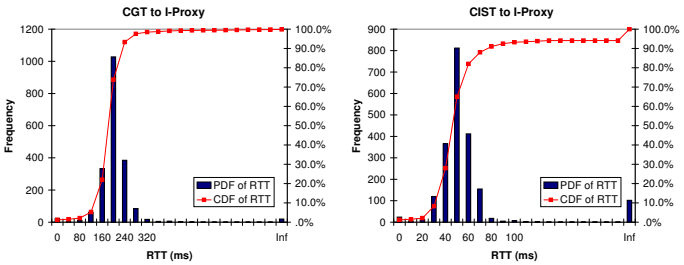

Fig. 3. RTT distributions for the paths between C_{GT} and the I-Proxy, and between C_{IST} and the I-Proxy. We indicate the packet loss rate as the packets are delayed infinitely.

In the experiments, in accordance with the bandwidth limitations of C_{IST} we fixed the video rate to 120 Kbps. We used a standard H.264 codec to encode the test sequence FOREMAN (176×144) at a frame rate of 10 f/s. The clients transmitted 1500-byte video packets over UDP, which resulted in inter-packet spacing of 100 ms. During the transmission, the packets that could not be delivered in 200 ms were not displayed and counted as an unsuccessful packet. However, the packets arriving after their decoding deadlines were still used to decode subsequent predictively-coded frames. In order to quantify the quality improvements, we will compare the cases with and without the I-Proxy. We will present our results in terms of both the percentage of successful packets and average video quality.

4.1. Direct Video Transmission between C_{GT} and C_{IST}

In this experiment, both clients transmitted the video packets directly to each other. Although this method avoids the extra delays incurred when relaying over the I-Proxy, the observed delay statistics did not differ much on the end-to-end basis; we measured the mean RTT as 250 ms. Since this high RTT value rendered any retransmission attempt impractical, each packet had only one transmission opportunity. By the end of 30-minute videoconferencing session, we determined the percentage of successful packets as around 91% for both clients, which actually produced a choppy video. The remaining packets were either lost or missed their decoding deadlines. In terms of video quality, both clients achieved merely around 31 dB.

The main result of this experiment is that late/lost packets are inevitable when there is only one transmission opportunity in the best-effort Internet and without any error-protection method the video quality suffers.

[1] On the backward transmissions, *i.e.*, from the I-Proxy to the clients, we observed similar characteristics to the ones obtained in forward direction.

4.2. Video Transmission from C_{GT} to C_{IST} via I-Proxy

When we examine the RTT distributions given in Fig. 3, we observe more favorable characteristics between C_{GT} and the I-Proxy compared to the ones between C_{IST} and the I-Proxy. This implies that the packets mostly get lost between the I-Proxy and C_{IST}. Considering the short RTT ($\backsim$ 60 ms) between them, the lost packets can be recovered by the I-Proxy before their decoding deadlines pass. Armed with this retransmission capability, our goal is to determine the potential quality gain over the success rate of 91%.

Recall that C_{IST} has to estimate the RTO and decide on a retransmission for a missing packet. For this purpose, we first adopted the estimator given in (1) with $\alpha = 0.8$ and $\beta = 0.5$. Although these values kept the number of redundant retransmission requests small [2], they largely overestimated the RTO. Consequently, retransmission requests were delayed and the subsequent attempts were unsuccessful in delivering the packets in a timely fashion. In particular, with $\alpha = 0.8$ and $\beta = 0.5$, 93% of the total packets were received successfully and only 10% of the retransmission requests were redundant. As the second case, we repeated the same experiment with $\alpha = 0.9$ and $\beta = 0.3$. In this case, we achieved a success rate of 97% at the expense of 25% redundant retransmission request rate. With these parameters, the displayed video at C_{IST} had an average quality of 35.2 dB.

4.3. Video Transmission from C_{IST} to C_{GT} via I-Proxy

Next, let us focus on the video transmission from C_{IST} to C_{GT}. As mentioned above, because of the uneven path characteristics, the majority of the packet losses are experienced between C_{IST} and the I-Proxy. If the I-Proxy does not monitor the packets coming from C_{IST} and take the necessary actions on time, then C_{GT} will eventually have to identify the lost packets and ask for a retransmission. However, because of the large delay between C_{GT} and C_{IST}, these retransmission efforts will be essentially useless. To overcome this problem, the I-Proxy should monitor the packet arrivals, identify the lost packets and send a retransmission request to C_{IST}. To this effect, the I-Proxy estimated the RTO by using (1). With the values of $\alpha = 0.9$ and $\beta = 0.3$, we measured the achieved success rate as 98% and the average video quality as 36.1 dB. In the overall, 17% of the retransmission requests made by the I-Proxy were redundant.

We summarize our results in Table 1. The results clearly demonstrate that we can maintain more stable video experience and improve the average quality drastically by employing an I-Proxy between the end-users.

Table 1. Experimental results for the FOREMAN sequence.

	Success Rate	Average Quality	% of Red. ARQ
Without I-Proxy			
$C_{GT} \rightarrow C_{IST}$	90.8%	30.8 dB	NA
$C_{IST} \rightarrow C_{GT}$	91.5%	31.3 dB	NA
With I-Proxy			
$C_{GT} \rightarrow C_{IST}$	97.1%	35.2 dB	25%
$C_{IST} \rightarrow C_{GT}$	97.9%	36.1 dB	17%

[2] A retransmission request is considered to be *redundant* if any of the previous transmission attempts becomes a success.

5. CONCLUSIONS

In this paper, we presented practical uses of proxies in delay-sensitive video applications. In contrast to complex packet scheduling algorithms, our methodology is relatively straightforward yet effective. Since we do not advocate any particular congestion or routing mechanism in the underlying network, this methodology does not require any native network support. Hence, the worldwide service providers can easily provide I-Proxy services to improve the experience of their customers when they communicate over large distances.

Of course, when the implementation issues are considered, relaying packets over an I-Proxy poses several challenging problems. In particular, the selection of an appropriate I-Proxy plays an important role in improving the video quality. As a future work, we will consider the architectures, mechanisms and policies by which an effective I-Proxy selection can be accomplished.

Acknowledgments

The authors would like to thank Dr. Nail Akar and Dr. Ezhan Karasan of Bilkent University for their support in this project. This work is supported by NSF under NSF CAREER award CCR-0133221.

6. REFERENCES

[1] Y. Wang, M. Claypool, and Z. Zuo, "An empirical study of realvideo performance across the internet," in *ACM SIGCOMM Internet Measurement Workshop*, 2001.

[2] Dmitri Loguinov and Hayder Radha, "End-to-end internet video traffic dynamics: Statistical study and analysis," in *IEEE Int. Conf. Computer Communications (INFOCOM)*, 2002.

[3] Dina Katabi and Charles Blake, "Inferring congestion sharing and path characteristics from packet interarrival times," *Technical Report MIT-LCS*, 2001.

[4] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, 2001, submitted.

[5] Y. Wang, Z. Zhang, D. Du, and D. Su, "A networkconscious approach to endtoend video delivery over wide area networks using proxy servers," in *IEEE Int. Conf. Computer Communications (INFOCOM)*, 1998.

[6] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *IEEE Int. Conf. Computer Communications (INFOCOM)*, 1999.

[7] F. Hartanto, M. Reisslein, and K. W. Ross, "Interactive video streaming with proxy servers," *Information Sciences, an International Journal*, 2001.

[8] Chitra Venkatramani, Olivier Verscheure, Pascal Frossard, and Kang-Won Lee, "Optimal proxy management for multimedia streaming in content distribution networks," in *ACM NOSSDAV*, 2002.

[9] Dmitri Loguinov and Hayder Radha, "On retransmission schemes for real-time streaming in the internet," in *IEEE Int. Conf. Computer Communications (INFOCOM)*, 2001.

[10] Rajarshi Gupta, Mike Chen, Steven McCanne, and Jean Walrand, "A receiver-driven transport protocol for the web," *Telecommunication Systems Journal*, vol. 21, no. 2-4, pp. 213–230, December 2002.