Playback Adaptation with Transport-Protocol Awareness for Wireless Video Streaming

Antonios Argyriou and Vijay Madisetti

School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250, USA
Email: {anargyr,vkm}@ece.gatech.edu

Abstract— **This paper introduces a new idea for implementing playback adaptation algorithms suitable for video streaming applications in wireless IP networks. The proposed algorithm depends on the accurate estimation of the expected latency of lost video packets. The latency is estimated according to the transport protocol being used, which in our case is TCP. The client uses this estimate in order to adjust the playback rate according to a new algorithm so that both buffer underflows and overflows can be avoided. The proposed algorithm in non-intrusive to the protocol stack since it does not require any modifications to the transport protocol. To validate the basic principles behind our idea, we provide a set of comparative experimental results with other streaming systems in terms of the rate of underflow events and the actual PSNR.**

I. Introduction

Video streaming is one of the most popular applications for wireless and mobile networks. Considerable research efforts have targeted this area so that despite the intricacies of the wireless medium [1], [2], the user perceived quality is high. However, the problem is not that simple. The heterogeneity of the several existing access networks today complicates even more this task. Therefore, it is important for a media streaming system to be adaptive so that it can offer the best possible playback quality given the previously mentioned problems.

In this paper, we are concerned with an adaptive video streaming mechanism that is based on the regulation of the playback rate at the client [3], [4]. With this mechanism, the client buffers packets during a good channel state and keeps a constant playback rate, while it drops the playback rate when the channel conditions are unfavorable. Therefore, buffer underflow events are minimized at the cost of reduced frame rate. This mechanism allows playback adaptation exclusively at the client without intervention of the streaming server. A considerable number of similar systems can be found in the literature [2]. However, the majority of these mechanisms assume a communication model that uses a simple multiplexing protocol like UDP, and therefore the focus shifts on the accurate model of the underlying wireless channel.

While TCP has long been considered unsuitable for video streaming [2], it is used by the most popular video steaming applications like WindowsMedia [5], QuickTime [6], and RealPlayer [7]. In this paper our goal is to design a playback adaptation mechanism for the wireless client when the transport protocol is TCP. In order to realize our goal we adopt a slightly different approach than the current practice. In order to select the optimal playback rate at any time instant, we initially focus on the development of a latency model for TCP. With the latency model the client can estimate the ability of the sender to deliver packets when events such as handoffs or random packet drops take place. Subsequently, the optimal playback rate at client is regulated so that the playback buffer occupancy is constant. Our approach could be classified as a cross-layer mechanism, since it utilizes knowledge from the transport layer to define the desired application behavior. However, the crucial difference of our work is that this principle is realized implicitly without modifying the transport protocol.

II. The Video Streaming System

Figure 1 presents a high level view of our system model, which consists of a streaming server, the mobile client and the network.

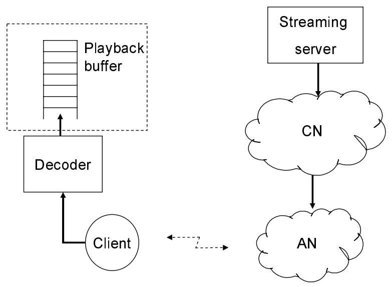

Fig. 1. System model where the core and access networks are independent.

The Video Source: The video server streams a video sequence that is encoded offline by an H.263 encoder. The encoded bitstream contains intra frames (I), predicted frames (P), and bidirectionally predicted frames (B). The encoded sequence is packetized into RTP packets, which are sent to the transport protocol which in our case is TCP. The streaming scenario we want to capture, assumes that there is a unicast end-to-end session with the data flowing from the server to the client. The server transmits packets to the network which will either arrive at the client or get lost.

The Network: Our network model is slightly different from existing work, since we wanted to also account for the effect of handoffs. We model the access network (AN) as a three-state Markov chain [8], where the three states are good, bad and handoff. This implies that the handoff induced packet losses are distinguished from other packet losses due to wireless errors in the access network. The duration of the successive good, bad, and handoff states are independent and identically distributed (i.i.d) with an exponential cumulative distribution function with mean T_G, T_B, and T_H respectively. Concerning the core network, it is modeled as a two-state Markov chain since it approximates with sufficient accuracy packet loss in the Internet, and the two states are lost (L) and received (R). Based on the assumption of three-state Markov model, the average packet loss rate because of wireless errors will be given by:

$$P_B = \frac{\pi_{GB} + \pi_{HB}}{\pi_{GB} + \pi_{BG} + \pi_{HB} + \pi_{BH}} \tag{1}$$

The transition probabilities are calculated using maximum likelihood estimators. For example $\hat{\pi}_{GB} = \frac{n_{GB}}{n_G}$, where n_{GB} is the number of times in the ACK messages that B state follows G state, n_{BG} is the number of times G follows B, and n_G is the number of times a good state is followed by a good. Concerning the calculation of the other transitional probabilities, they can be obtained similarly. In this way we obtain the average packet loss rate because of handoffs and wireless errors. Because the core network is independent, the total end-to-end packet loss rate will be:

$$P_{e2e} = P_L + (1 - P_L)(P_B + P_H) \tag{2}$$

III. Latency Model

While the analysis of TCP behavior during a handoff event is complex, two basic sideffects take place and we want to capture their effect on the latency of specific packets. Figure 2 will be used for explaining TCP behavior. Let R, Y, and X

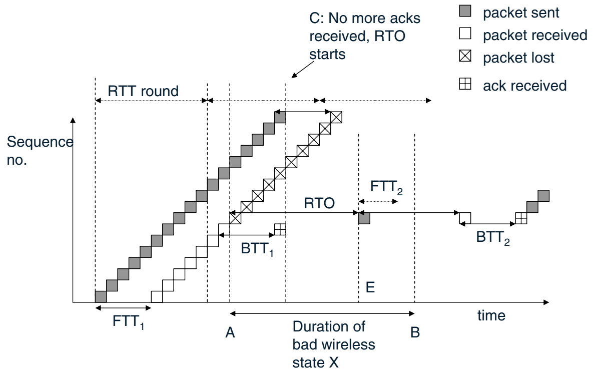

Fig. 2. Packet-level TCP behavior at the sender during IP layer handoff.

denote the random variables of the RTT, the RTO, and the handoff duration respectively. From figure 2 we can see that when the one way latency FTT or L, is smaller than X two cases arise:

Case 1, $X > t_E - t_A$: This condition means that the MH will still be in the handoff state, while the RTO expires the first time (time instant t_E in figure 2). So if we want to express the probability that the AN will be in handoff state for time ϵ after handoff we can write

$$P\{S(t_E + \epsilon) = G | S(t_E) = H\} = P_g(\epsilon) \tag{3}$$

This holds due to the memoryless property of the distribution of X, causing thus the channel state at time instant ϵ, to be independent from the state at t_A. Therefore:

$$P_g(\epsilon) = P\{S(\epsilon) = G\} = \frac{T_G}{T_H + T_G + T_H}(1 - e^{-\epsilon/T_G}) \tag{4}$$

The term $P_g(Y)$ expresses the probability for the channel to be good state after time Y, when the RTO expires. The average TO duration, for every possible RTO value, will be equal to the probability that the channel is in good state at that specific RTO renewal (given that it was bad before) times the value of the RTO. So this value will be:

$$\begin{aligned} L_{TO}(Y) &= \sum_{i=1}^{6} 2^{i-1} Y \times P_g(iY) \prod_{j=0}^{i-1} (1 - P_g(jY)) \\ &+ 64Y P_g(64Y) \prod_{i=0}^{6} (1 - P_g(iY)) \end{aligned} \tag{5}$$

The product term in the previous equation expresses the probability that the AN was in handoff state, when the TO expired in the previous time instants before i. In addition, after the first six consecutive TOs the value of the TO will be fixed to $64Y$. The last term on the above equation captures this event.

Case 2, $X < t_E - t_A$: In this case, we can see from figure 2, that it will also be $t_C < t_E$. This means that the sender will not experience a TO, but instead a TD and so it will fast retransmit the first missing packet. The average latency introduced due to this event will be

$$L_{TD} = \int_{t=0}^{T_H} (t + RTT) P_g(t) dt \tag{6}$$

Therefore, the total latency for TCP can be expressed from the previous two equations:

$$\begin{aligned} L_H^{TCP} &= \sum_{l=0}^{\infty} P[l < L] = \int_{t=Y}^{T_H} L_{TO}(t) f_X(t) dt \\ &+ \int_{t=0}^{T_H} (t + RTT) P_g(t) dt \end{aligned} \tag{7}$$

The first term in the above equation follows from the assumption that the disruption time due to handoff X, is exponentially distributed with a mean equal to T_H and a p.d.f $f_X(t)$.

Concerning the core network induced delay L_N, that occurs mainly due to buffering at the routing infrastructure, it has been shown that it could be modeled as a shifted Gamma distribution [9]. We will follow this latency distribution in this paper, so that the derivation of an analytical closed form solution is possible. We denote the p.d.f. as f_{L_N}. Several possible analyses can be preformed in order to model more accurate the core network performance, but this research is out of the scope of this paper.

IV. Playback Adaptation Algorithm

In this section we will answer how the previously derived latency estimate can be used by the playback algorithm at the client. During initialization, the client requests from the server a media file, which is progressively sent and added to the playback buffer. The size of the playback buffer at the client is denoted as B. When the backlog in the playback buffer reaches the ideal b^* bytes (in our case was set to 50%), playback starts at a rate of c bytes/sec, while the current playback buffer occupancy is denoted as b.

From figure 3 we can understand precisely the main concept behind the proposed model-based playback adaptation algorithm. In this figure, the receiver and playback curves are denoted as $R(t)$ and $P(t)$ respectively. The client calculates the average packet latency every intra-frame period t_f, and derives an expected number of received bytes. The aforementioned task can be performed over a series of w intra-frame periods, leading to tradeoffs between accuracy and processing cost. Our objective now is to maintain the buffer occupancy

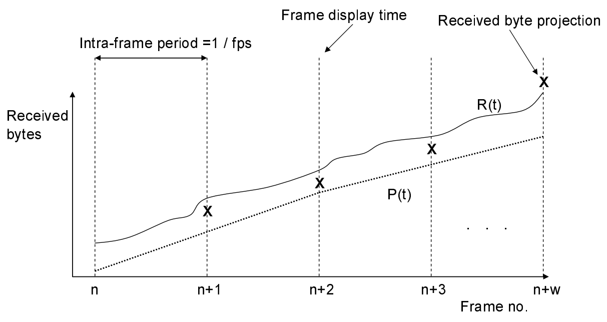

Fig. 3. Playback adaptation with lookahead supported by the analytical TCP latency model.

at b^* so that both underflows and overflows are avoided. Now, over the period of $w-1$ periods the buffer should ideally contain $(w-1)b^*$ bytes. The expected number of bytes that the buffer will be lacking over the period of $w-1$ intra-frame periods will be equal to:

$$E[d(n,w)] = (w-1)b^* - \sum_{n+1}^{n+w} \frac{t_f(i)}{L(i)} \tag{8}$$

where $\frac{t_f(i)}{L(i)}$ gives the expected number of delivered bytes in the period i. Note that the previous equation can provide both a positive and a negative value representing a lack or surplus in bytes respectively. Based on this projection, we define a new playback adaptation policy at the client. The playback rate will be reset for the next frame $(n+1)$ as follows:

$$c(n+1) = \begin{cases} c(n) - (b^* - b) - E[d(n,w)] & \text{if } b < b^* \\ c(n) - E[d(n,w)] & \text{if } b = b^* \\ c(n) + (b - b^*) - E[d(n,w)] & \text{if } b > b^* \end{cases}$$

For example when there is a surplus in the received bytes $(b > b^*)$, the playback rate is increased by this surplus minus the projection for the next w frames.

V. Experiments

The network testbed for our experiments consists of a client-server configuration that are linux boxes while the middlebox is freeBSD machine that acts as a router. We used the middlebox with the Dummynet software [10], for emulating the packet losses due to wireless errors, handoffs in the access network, and buffer overflows in the core network routers. The sequences FOREMAN, and COASTGURAD were used for the video streaming experiments. The capacity of the bottleneck link between the two routers was set to 250Kbps. All the sequences were encoded at 128Kbps with a target frame rate of 15 fps. The results were obtained by running the same scenario 100 times and averaging the PSNR values of the same experiments.

In the first experiment, we evaluated the effect of the handoffs on the performance of the playback buffer. In figure 4, the x-axis depicts the average value for the handoff duration (T_H), and in the y-axis the normalized buffer underflow rate. We compare our approach, with a playback adaptation strategy reported at [4], and we also set the playback buffer size to 200 packets. The playback adaptation algorithms reported at [3], [4] are based on the assumption that the underlying channel is a two-state Markov chain. This model has been shown to be very effective in capturing the behavior of packet losses in wireless fading channels. However our channel model captures also the effect of sustained handoffs.

Results for TCP can be seen in figure 4, and demonstrate that the proposed playback adaptation algorithm can lead to a lower number of buffer underflow events when compared to playback adaptation algorithms identified in the literature [3]. We also present in the same figure a version of the algorithm, where the server sents TCP parameter estimates to the client in order to improve prediction. It is interesting to note that when the handoff has small duration, then the version of the protocol where the server is also active, does not correspond to large benefits. However, when the handoff duration is increased the mobile client cannot make precise estimates concerning the optimal playback rate. The proposed buffer adaptation for TCP is proven more essential than initially expected, due to TCP's rapid throughput decrease.

At the core of our performance evaluation, are the experiments that stress test the heterogeneous path model, and essentially, the effect of heterogonous packet losses. Results for peak signal-to-noise ratio (PSNR) are depicted in figure 5. Since we evaluate a playback algorithm, merely presenting

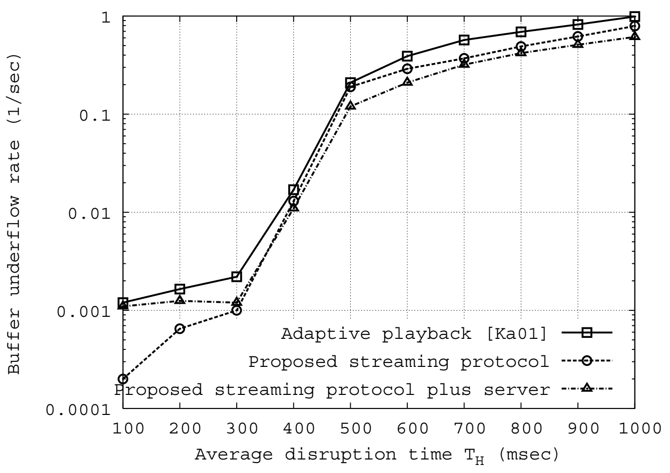

Fig. 4. Buffer underflow rate for TCP streaming experiment with a WLAN→WLAN handoff.

PSNR at the decoder does not make sense. Therefore, we performed this experiment by setting a lower bound of 10 frames per second at the decoder. This means that the decoder was displaying frames even if the playback buffer was empty in order to maintain this bound. By considering this, we show in this figure the effects for a fixed wireless packet loss rate of $P_W = 0.001$, and varying packet loss on the core network P_L. We compare our protocol with the wireless video adaptation strategy developed at [11], where the authors of that study consider only wireless errors. The advantage of the proposed algorithm, is that adaptation is performed at a finer granularity.

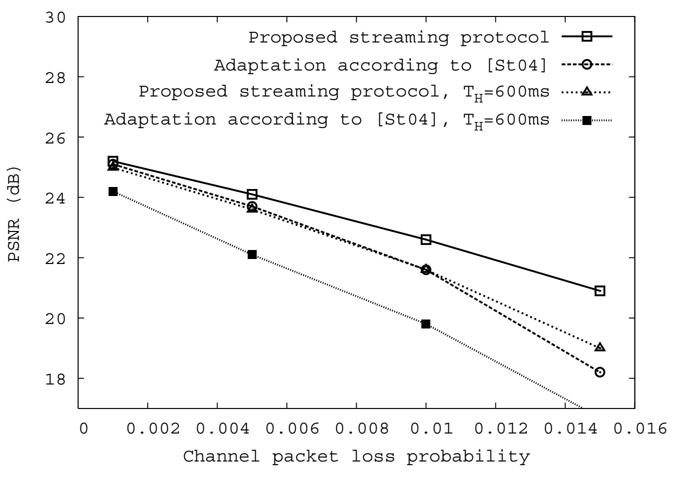

Fig. 5. PSNR for TCP streaming of the sequence FOREMAN.

VI. Conclusions

The contribution of this paper is that we introduce the new idea of utilizing knowledge about the behavior of a transport protocol for the design of media streaming algorithms. We developed a simple latency model for driving the design of an algorithm that controls playback adaptation at a mobile and wireless client. The proposed algorithm can be easily implemented on top of the TCP protocol, since it does not require any modifications of the protocol stack. Performance evaluation revealed the protocol's ability to maintain a low playback buffer underflow rate and significantly improve the perceived video quality.

References

[1] I. Stojmenovic, *Handbook of Wireless Networks and Mobile Computing*. John Wiley and Sons, February 2002.

[2] Y. Wang, J. Ostermann, and Y.-Q. Zhang, *Video Processing and Communications*. Prentice Hall, 2002.

[3] M. Kalman *et al.*, "Adaptive Media Playout for Low-Delay Video Streaming Over Error-Prone Channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 6, p. 841851, June 2004.

[4] M. Kalman, E. Steinbach, and B. Girod, "Adaptive playout for real-time media streaming," in *IEEE International Conference on Image Processing (ICIP)*, October 2001.

[5] "Windows media," http://www.microsoft.com.

[6] "Quicktime," http://www.apple.com.

[7] "Real media," http://www.realnetworks.com.

[8] L. N. Kanal and A. R. K. Sastry, "Models for channels with memory and their applications to error control," *IEEE Proceedings*, vol. 66, p. 724744, July 1978.

[9] J. Chakareski and B. Girod, "Rate-distortion Optimized Packet Scheduling and Routing for Media Streaming with Path Diversity," in *IEEE Data Compression Conference (DCC)*, 2003.

[10] "Dummynet," http://info.iet.unipi.it/ luigi/ip_dummynet/.

[11] T. Stockhammer *et al.*, "Streaming Video Over Variable Bit-Rate Wireless Channels," *IEEE Transactions on Multimedia*, 2004.