PROXY-DRIVEN RATE-DISTORTION OPTIMIZED VIDEO STREAMING OVER WIRELESS NETWORK USING ASYNCHRONOUS CLOCKS

*Danjue Li * Chen-Nee Chuah * Gene Cheung$^+$ S.J.Ben Yoo**

*Department of ECE
University of California, Davis

$^+$ HP Laboratories, Japan
HP Japan Ltd.

ABSTRACT

Video streaming over wireless network is challenging due to node mobility and high channel error rate. In this paper, we propose a joint server/peer, proxy-driven video streaming architecture to support high quality video streaming service over IEEE 802.11-based wireless networks. The video proxy tracks peer mobility pattern, coordinates among multiple senders and performs rate-distortion optimized streaming. We formulate the joint server /peer streaming as a combinatorial optimization problem and introduce the concept of *asynchronous clocks* to decouple the problem into three steps: first deciding the candidate set of potential senders among the wireless peers based on their mobility pattern, then selecting the sender in each optimization instance using asynchronous clocks, and finally applying point-to-point rate-distortion optimization framework between the selected sender-receiver pair. In addition, we consider two different system setups, *no cache* and *limited cache*, to investigate cache effect on the streaming performance. To design more realistic simulation models, we use empirical results from corporate wireless networks to model peer mobility. Simulation results show that our proposed joint streaming scheme has better performance than the server-only streaming scheme. Simulation results also show that having cache in the proxy can potentially improve the streaming performance.

1. INTRODUCTION

Advances in wireless communication technologies and mobile devices have made ubiquitous Internet access increasingly prevalent in everyday life. IEEE 802.11 [1]-based wireless local area network (LAN) is emerging as an attractive solution for providing network connectivity in businesses, educational institutions, and public places like airports, shopping malls, cafes, etc. Although most current uses of WLAN are for data transmission, the desire for supporting delay-sensitive applications such as voice-over-IP (VoIP) and video streaming is apparent.

In contrast to data transmission, audio/video communication generally is more tolerant to packet losses, but has a more stringent delay requirement. Any packet which arrives late will be useless for decoding and be equivalent to a packet loss. Interdependency of video data can cause error propagation and further degrade the video quality. Furthermore, compared with wired links, wireless links are more error-prone and unpredictable. End-host mobility, time-varying channel fading and physical layer corruption in WLAN can all cause the end-to-end perceived video quality to fluctuate greatly. To use WLAN to deliver high quality video streams, a robust streaming scheme which can cope with high packet losses, potential node mobility, and bandwidth variation is needed.

The goal of this paper is to support high quality video streaming service over 802.11 wireless LAN. We build upon our previous work [2], in which we proposed a joint server/peer, receiver-driven video streaming scheme. In [2], the receiver was responsible for tracking potential senders (servers and peers) and applying multiple point-to-point rate-distortion optimized streaming to enhance the overall video quality. However, the heavy computation load at the client would hamper implementation in small, portable devices due to the limitations on power consumption and processing capabilities. To address this issue, in this paper, we propose a proxy-driven scheme that moves the optimization task to the proxy. We also relax the assumption made in our previous work [2] that the receiver has the full knowledge of the video content kept in the mobile nodes (MNs) and the remote server. The limited memory space in the MNs and the inherent characteristics of video streaming applications make it unrealistic to assume the entire video content are available at neighboring MNs all the time. In this paper, we use the proxy attached to the access point to perform content discovery and keep track of the availability information of the desired video file in each mobile node.

Our contributions in this paper are threefold. First, we propose a novel proxy-driven architecture for streaming video over wireless LAN. The architecture leverages the neighboring MNs and the remote media server to form a joint sender group for streaming video content to a single wireless client. The proxy sets up a content table to keep track of the location of video files among wireless peers. Moreover, the proxy coordinates among different senders and performs rate-distortion optimized streaming. Compared with the sender-driven approach, our solution avoids the synchronization problem among multiple senders and can easily adapt to instantaneous changes in network conditions.

Second, we use the proxy to keep track of peer mobility patterns. Mobility patterns are defined using the distribution of the session duration and the revisit frequency of peers. Using the empirical results in [3], we design the simulation model to evaluate our scheme.

Third, we formulate the proxy-driven streaming scheme as a combinatorial optimization problem. To solve this combinatorial optimization problem, we decouple it into three steps: first using the mobility patterns of MNs to decide the membership of the joint sender group, then selecting a sender (server or peer MNs) for one transmission opportunity, and finally performing point-to-point rate-distortion optimized streaming for the selected sender-receiver pair. To solve the sender selection problem, a set of asynchronous clocks are introduced in the proxy, each clock being responsible for one particular sender-receiver pair. Working frequency of the clock is adapted based on corresponding network conditions. We implement and evaluate our scheme using network simulator (NS) ns-2.27 [4].

The rest of this paper is organized as follows. Section 2 presents a brief survey on related works. Section 3 describes our proxy-driven video streaming architecture and models the system. Section 4 characterizes user mobility in WLAN and presents the methodology we use to evaluate our design. The main analytical formulations and results are presented in Section 5. Section 6 presents our simulation results using NS. In Section 7, we conclude the paper and discuss future research.

2. RELATED WORK

There have been some solutions in past literature to support robust video transmission over wireless networks. Qiao et al. [5] proposed a two-step adaptive hybrid Automatic Repeat Request (ARQ) scheme using Reed-Solomon (RS) codes. Xu et al. [6] combined various error control strategies, MAC layer FEC/ARQ, application layer FEC and priority queuing, with FGS scalable video coding to achieve reliable communication over IEEE 802.11b WLANs. Based on their previous work [6], Krishnamachari et al. proposed a cross-layer approach [7] to enhance the robustness and efficiency of scalable video transmission by performing joint optimization of the various protection schemes, which include application layer FEC, bandwidth-adaptive compression using scalable coding, and adaptive packetization strategies. Shan and Zakhor [8] combined application layer adaptation and link layer packetization to achieve robust wireless video transmission based on the packet loss ratio, frame type, frame delivery deadlines, etc. Majumdar et al. proposed a hybrid FEC/ARQ scheme to increase the robustness of the video streaming in WLAN [9]. All these works focus on solving the problem of bandwidth adaptation and robustness to packet-losses in WLAN purely from the joint source/channel coding point of view. In contrast, we address this problem by applying rate-distortion optimized streaming framework.

Using path diversity for video streaming from multiple sources to enhance the streaming performance has also been explored in the previous literature. Nguyen et al. [10] presented algorithms to stream video from multiple servers. Miu et al. [11] proposed a system to tackle the quality degradation of streaming video over IEEE 802.11b wireless networks by leveraging the path diversity between multiple access points and the mobile client. However, none of them is rate-distortion optimized. Chou et al. [12] discussed a novel peer-to-peer (P2P) multicast streaming mechanism to stream a flash crowd over the Internet. Instead, we are concerned with optimizing streaming on a single client for applications like movie previews or video-on-demand.

To the best of the authors' knowledge, the work of Chou et al. [13] is the first to systematically define and solve the point-to-point rate-distortion optimized video streaming problem. Our work leverages their idea heavily. We extend the concept of asynchronous clocks introduced in our previous work [2] and combine it with the rate-distortion optimized streaming scheme [13] for the implementation in *multi-source diversity streaming*. Among all works along this line of research, what Chakareski et al. have done are the most closely related to our work. First they focused on single path streaming and proposed a hybrid receiver/sender driven streaming schemes [14] while our work focuses on simultaneously streaming video from multiple paths. Later, they considered path diversity for media streaming in a receiver-driven rate-distortion optimized framework under a constraint on the expected overall transmission rate from the servers to the client [15]. Alternatively,

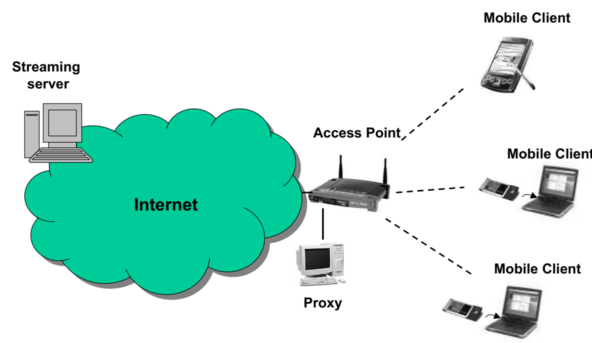

Fig. 1. Streaming on demand from joint sender group to a client through a proxy server

we consider M rate constraints for M distinct delivery paths from M senders. Moreover, they focused on general static path diversity streaming problem. The number of paths used during the video transmission was fixed during the transmission. It did not consider the dynamic membership of senders. In our work, we cope with the dynamic membership problem of senders caused by the mobility of wireless peers.

3. PROXY-DRIVEN VIDEO STREAMING OVER WIRELESS LAN

In this section, we describe our proxy-driven video streaming architecture that coordinates multi-source streaming to clients, discuss source and channel models, and state the rate control schemes for video streaming applications.

3.1. System Architecture

Figure 1 illustrates our proposed proxy-driven video streaming system, where the client will stream video content from the remote media server and peers in the WLAN simultaneously. Different from the infrastructure/ad-hoc based system introduced in [2], we only use the infrastructure communication mode of IEEE802.11 interface card in this paper. As illustrated by Figure 1, a proxy will sit at the access point (AP) of the WLAN, gather the video information from the server and peers and stream the information to the client.

When a client node wants to receive a video stream, it sends a request for media to the proxy. Instead of forwarding the request directly to the media server, the proxy will send a message to the media server to ask for a *rate-distortion preamble* of the desired presentation, containing the directed acyclic graph (DAG) representation of the media source. The preamble is necessary to perform rate-distortion optimized packet scheduling at the proxy. DAG source model will be described in details in Section 3.2.1. Meanwhile, the proxy will broadcast messages to mobile nodes (MNs) in the WLAN to conscript willing senders of the desired video content, who will then reply affirmatively. Based on those replies and the mobility patterns of MNs, the proxy will determine the candidate sender group (including server and MNs). Sender selection is discussed in Section 3.1.1.

Compared with receiver-driven approach [2], the proxy-driven approach has the following advantages. First, since the proxy is associated with the access point, it is easier for the proxy to keep track of the mobility pattern of each MN. The receiver-driven approach requires the receiver to send out separate probing messages to track the mobility of its peers, which increases the communication overhead. Since access points will periodically send out beacons and get the probe responses from MNs, the proxy-driven approach can leverage those responses to check whether MNs in the sender group are still alive. Second, the proxy-driven approach takes over the optimization task from the client, which can potentially be computationally intensive. It can reduce the energy consumption in the client, thus making the scheme feasible for implementation even in small, portable mobile equipment such as mobile phones and PDAs.

3.1.1. Membership Decision of the Candidate Sender Group

The mobility of MNs makes memberships of the candidate sender group dynamic, which can potentially lead to high variation of the video quality perceived by the client. In order to smooth the video quality variation, we try to include mobile nodes with less mobility while constructing the candidate sender group. The proxy will keep track of the mobility pattern of MNs and try to predict how frequently the MN will join/leave the WLAN and once it joins the WLAN, how long it will stay connected. Those MNs which stay associated with the AP longer and are less likely to leave/join the WLAN frequently will have higher priorities to be selected as senders. In order to evaluate the membership decision scheme, we will use the empirical results from [16] to design our simulation.

3.1.2. No Cache vs. Limited Cache

There are two different communication modes based on cache availability in the proxy. We will analyze these two cases separately.

When there is no cache available in the proxy, the proxy cannot store any packet from senders. All the requested packets from senders will go directly to the client without leaving a copy in the proxy. Hence any packet lost from a sender to the client via the proxy can only be recovered by another request for packet by the proxy to a willing sender. Client will send an acknowledgment packet to the proxy for every correctly received packet.

When cache of limited size is available in the proxy, the proxy will cache packets it receives from senders. Let the size of the cache be K in RTP packets as illustrated in Figure 2. In this scenario, the proxy will first request packets from members of the joint sender group until it correctly receives them. Once the proxy gets the requested packets, it will cache them and perform (re)transmission to the client. Client will send an acknowledgment packet to the proxy for every correctly received packet.

For the limited cache scenario, since the cache size is finite, we can only cache limited number of packets. When a packet arrives from a sender to the proxy and the cache is full, contention takes place. To resolve cache contention, a comprehensive cache management strategy is needed.

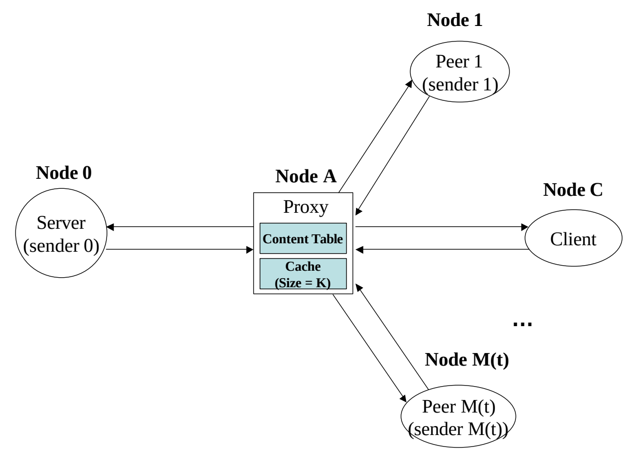

Fig. 2. System model for joint server/peers video streaming

3.2. Preliminaries

We discuss the source and channel model used to model our video streaming system in this section. The models will be used in Section 5 to formulate the streaming optimization problem.

3.2.1. Source Model

We use the directed acyclic graph (DAG) introduced in [13] to model the video source. Each frame i is represented by a data unit DU_i. Like [13], data unit is the smallest granularity we will consider in the optimization. Constants that are associated with each data unit DU_i include: the size n_i, the decoding time T_i, and the distortion reduction D_i. The size n_i is the size of the data unit in RTP packets. The decoding time T_i is the time by which the data unit DU_i must arrive at the client. D_i is the distortion reduction if data unit DU_i arrives at the client on time and is decoded.

3.2.2. Channel Model

Consider a network with a media server (node 0), a proxy (node A), a client (node C) and a set of peers (node k, $k \in \{1, ..., M(t)\}$). $M(t)$ is the size of candidate peer group in the given optimization instant t. We denote the channel between node k and the proxy as CH_k, $k \in \{0, ..., M(t), C\}$. Each channel includes one forward path and a backward path. We define the forward path as the path from the proxy to the node k, and the backward path as the reverse. We model the forward path and the backward path using an independent time-invariant packet erasure channel with random delay. We define the packet loss as one minus the probability that a packet transmitted by the proxy (or node k) is successfully received by the node k (or the proxy). Hence, for CH_k, we can characterize its forward path by random loss ϵ_k^F and delay densities $f_k^F(x) = \gamma_k^F e^{-\gamma_k^F x}$, $x \geq 0$, and γ_k^F is a constant determined through measurements [17]. Then the packet sent by the proxy at time T will be received correctly by node k by time T' with the following probability:

$$p_k^F(T'-T) = (1-\epsilon_k^F)\int_0^{T'-T} f_k^F(x)dx \qquad (1)$$

Similarly, the backward path for CH_k can be characterized by ϵ_k^B and delay densities $f_k^B(x) = \gamma_k^B e^{-\gamma_k^B x}$, $x \geq 0$. The packet sent by the node k at time T will be received correctly by the proxy at time T' with the probability $p_k^B(T'-T)$, which has the same form as (1).

Thus, for CH_k, the probability that a request (or data unit) sent by the proxy at time T to the node k will result in a data unit (or an acknowledgment) successfully arriving in the proxy by time T' will be:

$$p_k(T'-T) = (1-\epsilon_k^F)(1-\epsilon_k^B)\int_0^{T'-T} f_k^F(x) * f_k^B(x)dx \qquad (2)$$

where $*$ denotes convolution.

3.3. Rate Control for Video Streaming

In order not to overload the links and potentially cause congestion problem, we have to perform rate/congestion control. Equation based rate control [18], also known as TCP-friendly rate control (TFRC), is a widely used rate control scheme over wired networks. However, TFRC works poorly for wireless networks because it cannot distinguish between packet loss due to channel error versus contention-induced collisions (i.e., congestion) [19].

In this paper, we perform TFRC for streaming video from the media server since it will traverse the Internet. We also implement TFRC for each wireless connection between the peers and the proxy, and the sending rate Ω_j from sender-receiver pair j is determined as the following:

$$\Omega_j = \frac{L}{\mu_j\sqrt{2\alpha_j/3} + t_{RTO}(3\sqrt{3\alpha_j/8})\alpha_j(1+32\alpha_j^2)}, \qquad (3)$$

where L is the packet size in bytes/packet, μ_j is the round-trip time, α_j is the loss event rate perceived by sender j and t_{RTO} is TCP retransmission timeout value. According to the work of Floyd et al. [18], it is reasonable and practical to estimate t_{RTO} using $t_{RTO} = 4\mu_j$.

Using one TFRC connection for wireless connection between the proxy and the peer may result in under utilization of the wireless bandwidth, especially when the bulk of the packet loss is due to physical channel errors. Chen and Zakhor [20] proposed the use of multiple simultaneous TFRC connections for a given wireless streaming application between the same two endpoints to increase throughput. However, such extension is out of the scope of this paper and we will explore other rate-control mechanisms in our future work.

3.4. Asynchronous Clocks

In order to maximize utility of available bandwidths, we utilize a concept called *asynchronous clocks* developed in [2]. The idea is to set up a clock j at the proxy for each sender-receiver pair j that wakes periodically with period $\Delta_j = L/\Omega_j$, assuming packet size L is the same for all senders. When a clock j wakes up, it signals a data unit transmission is granted between sender-receiver pair j. Requesting/transmitting at frequency $1/\Delta_j$ would mean bandwidth j established using (3) is maximally utilized.

For the no-cache scenario, because the transmission path includes both the sender-proxy connection and the proxy-client connection, the proxy will until a clock awakes in both the first leg (sender-proxy pair j) *and* the second leg (proxy-client) before sending a request to sender j. If more than one first leg clock awakes while waiting for the second leg clock, then the proxy must pick one sender corresponding to one of the awaken first leg clocks for data unit transmission request. After selecting a sender j, the proxy will then decide which data unit to request from sender j.

For the limited cache scenario, if a first leg clock j wakes up, the proxy will immediately request a data unit transmission from sender j. If the second leg clock wakes up, the proxy will immediately select a data unit from all available data units in the cache for transmission to the client. In both the no-cache and the limited cache scenario, the proxy will decide which data unit to request/transmit in a rate-distortion optimal fashion. The data unit selection problem is formalized as an optimization problem and is discussed in details in Section 5.

4. CHARACTERIZING MOBILITY IN A WIRELESS LAN

To design more realistic simulation scenarios for wireless networks, understanding user mobility and network usage characteristics is critical. There are numerous studies on the performance of the wireless infrastructure and the user access and mobility patterns. Tang et al. [21] examined a twelve-week trace of a building-wide wireless LAN and analyzed the user behavior (when and how intensively people use the network and how much they move around), overall network traffic and load characteristics, and traffic characteristics from a single user point of view. Balazinska et al. [3] studied user mobility patterns in a large corporate environment and introduce new metrics, persistence and prevalence to model user mobility. Chinchilla et al. [16] found that many users spend a large fraction of their time associated with a single AP. Users tend to visit the same AP frequently and stay for long periods of time. This observation justifies the feasibility of our design, i.e. streaming video from the server and mobile peers simultaneously. In our paper, we will use the empirical results of Balazinska et al. [3] to design our simulation model and evaluate our scheme. To model the mobility of an individual mobile user, we need to compute two metrics: session duration and revisit interval.

4.1. Session Duration

In our architecture, session duration refers to the amount of time that a user stays associated with an access point (and hence served by our proxy) before moving to another access point or leaving the network. For MN j, we denote the session duration as V_j. Our definition of session duration is equivalent to the persistence metric in Balazinska et al's work. [3], which is shown to follow a power law distribution with exponential 1.78, i.e.,

$$P(X = x) \sim x^{-1.78}.$$

4.2. Revisit Interval

After a MN leaves the AP, it is likely that it will come back and visit the AP again. Revisit interval is the amount of time before the next visit of the user, which also shows how likely it is for a MN to visit an AP that is has visited recently. Balazinska et al. [3] presented a prevalence metrics to measure the fraction of time that a user spends with a given AP. In this paper, we will derive the revisit interval based on the prevalence distribution given by Balazinska et al [3].

Let R_j be the revisit interval. The prevalence of MN j is $Prev_j$ and the prevalence probability distribution follows a power law with a low exponents [3], i.e. $P(Prev_j = x) \sim 0.001x^{-1.75}$. Using $Prev_j$ and V_j, the revisit interval R_j is defined by $R_j = V_j/Prev_j$.

5. RATE-DISTORTION OPTIMIZED TRANSMISSION

In this section, we formulate the proxy-driven streaming problem as a combinatorial optimization problem.

5.1. Problem Formulation

Suppose there are N data units in a selected optimization window under consideration for (re)transmission. As previously discussed, an optimization instance appears when an asynchronous clock j wakes up at time t_o [2], presenting an immediate transmission opportunity for sender-receiver pair j. For the non-caching scenario, that actually means a token has been collected at both the first leg (sender-proxy pair j) and the second leg (proxy-client pair), at which time the sending of a transmission request to sender j is granted to the proxy at time t_o. The received data unit will be forwarded to the client immediately upon arrival at the proxy. For the limited caching scenario, the awakened clock can be either in the first or second leg. If it is the first leg, it signals a transmission request to sender j is granted to the proxy. The optimization window will contain data units that have not yet arrived at the proxy correctly. If it is the second leg, it signals a data unit transmission to the client is granted to the proxy. The optimization window will then only contain data units that have already arrived at the proxy and are available in the cache. In any of the three cases, the remaining question is *which* data unit should be requested/transmitted among N data units in the optimization window. We formulate that problem here in this section.

For each data unit $DU_i, i \in \{1, 2, \ldots, N\}$, we define a transmission policy π_i, $\pi_i \in \Pi$, where Π corresponds to a family of transmission schedules, which dictates when and how the data unit should be requested/transmitted. Let $\pi = \{\pi_1, \ldots, \pi_N\}$ be the transmission vector for N data units. Let π_i be defined as $\pi_i = \{H_i, c_i\}$, where H_i is the transmission history of DU_i, defined by $H_i = \left\{(t_i^{(1)}, s_i^{(1)}), \ldots, (t_i^{(l_i)}, s_i^{(l_i)})\right\}$. l_i is the number of previous transmission attempts, $t_i^{(k)}$ and $s_i^{(k)}$ are the timestamp and sender ID of attempt k, respectively. Sender ID $s_i^{(k)}$ identifies the sender to which the packet was sent at attempt k.

Transmission decision c_i determines if DU_i is decided for (re)transmission for sender j when clock j awakes. Hence, c_i is (t_o, j) if DU_i is selected. Otherwise, c_i is the empty set $\emptyset$.

5.1.1. *Scenario One: No cache in the proxy*

In the *no-cache* scenario, the requested data unit will be forwarded immediately to the client by the proxy. Moreover, the proxy has to resend a request to one of the potential senders for retransmission instead of performing local retransmission from the proxy. Following the discussion of channel modeling in Section 3.2.2, the probability that a request packet sent by the proxy to sender j at time T will result in a data unit successfully arrives in the client by time T' is:

$$Q_j(T'-T) = (1-\epsilon_j^F)(1-\epsilon_j^B)(1-\epsilon_C^F)\int_0^{T'-T} f_j^F(x)*f_j^B(x)*f_C^F(x)dx \tag{4}$$

Using $Q_j(T'-T)$ and the source model defined in Section 3.2.1, the probability of successfully receiving data unit i, $q_i(\pi_i)$, is:

$$q_i(\pi_i) = 1 - \left(1 - Q_{c_i}(T_i - t_o)\right)\prod_{k=1}^{l_i}\left(1 - Q_{s_i^{(k)}}(T_i - t_i^{(k)})\right) \tag{5}$$

where $Q_{c_i}(T_i - t_o) = 0$ if c_i is $\emptyset$.

5.1.2. *Scenario Two: Limited cache in the proxy*

As we discussed earlier, the awakened clock can correspond to either the first or the second leg. If it is the first leg, we will only consider data units that have yet been successfully arrived at the proxy. For a data unit DU_i that that has not yet correctly arrived at the cache, π_i will be the tranmission request policy from proxy to senders. We can define the probability that DU_i will arrive at the proxy before its deadline as:

$$q_i(\pi_i) = 1 - \left(1 - p_{c_i}(T_i - t_o)\right)\prod_{k=1}^{l_i}\left(1 - p_{s_i^{(k)}}(T_i - t_i^{(k)})\right) \tag{6}$$

If the awakened clock corresponds to the second leg — proxy-client pair, then we will consider only N of all data units that have safely arrived at the proxy and are available in the cache. For a data unit DU_i that is in the cache, π_i will be the transmission policy from the proxy to the client. We can define the probability that DU_i will arrive at the client before its deadline as:

$$q_i(\pi_i) = 1 - \left(1 - p_C^F(T_i - t_o)\right)\prod_{k=1}^{l_i}\left(1 - p_C^F(T_i - t_i^{(k)})\right) \tag{7}$$

Now we can deduce the expected distortion of the group of N data units in the optimization window under the transmission request policy vector π based on the probability q_i for each data unit DU_i and the DAG source model:

$$D(\pi) = \{D_0 - \sum_{i=1}^{N} D_i \prod_{l \preceq i} q_l(\pi_l)\} \tag{8}$$

where $l \preceq i$ denotes the set of DU_l's that precede or equal to DU_i in DAG, D_0 is the overall expected distortion for the group given no data unit is received. Note that for no-cache scenario and for proxy-client pair in the limited-scenario, $D(\pi)$ corresponds to the expected distortion at the *client*, while for the sender-proxy pair j in the limited-cache scenario, $D(\pi)$ corresponds to the expected distortion at the *proxy*.

5.2. Solution

5.2.1. *Point-to-Point RD optimized streaming*

When an asynchronous clock j expires, we will request a data unit transmission from sender j and the problem will be simplified to a point-to-point RD optimized streaming problem. Following the discussion of [13], the optimal data unit DU_i for transmission is the one with the largest $\lambda_i = \lambda'_i S_i / n_i$, where λ'_i and S_i are the increase of successful delivery likelihood given one transmission is sent at the optimization instant and data sensitivity, respectively. λ'_i and S_i can be defined as following:

$$\lambda'_i = q_i(\pi_{i,1}) - q_i(\pi_{i,0}) \quad (9)$$

$$S_i = \sum_{k \succeq i} D_k \prod_{\substack{l \preceq k \\ l \neq i}} q_l(\pi_l) \quad (10)$$

where $\pi_{i,1} = \{H_i, (j, t_o)\}$ is the transmission policy of DU_i given a transmission request is sent to j at time t_o, and $\pi_{i,0} = \{H_i\}$ is the policy of DU_i given no request is sent at time t_o. See [13] for more details.

5.2.2. *Cache Management*

We consider cache flushing strategy to maintain cache space availability for incoming data units. The cache will be flushed when one of the following cases happens: i) an ACK is received from client; ii) a cached data unit expires; and, iii) contention takes place due to arrival of a data unit from sender while the cache is full. In the first two cases, the data unit ACKed or expired will be deleted from the cache immediately. In the third case, we compare λ value of data units (discussed in Section 5.2.1). Lowest λ-valued data units will be dropped until cache overflow is avoided.

6. SIMULATION STUDY

To evaluate our proposed proxy-driven, multiple-sender video streaming scheme, we use the network simulator NS-2.27 to implement different transmission strategies and compare the streaming performance. We characterize the received video quality using Peak Signal-to-Noise Ratio (PSNR). We generate realistic mobility patterns for the wireless nodes, as described in Section 4, and measure the effect of the mobility on the system performance. In addition, we evaluate how the ability to cache data units at the proxy affect the streaming performance.

6.1. Simulation Framework

We simulate a hybrid network environment that consists of two parts: wireless LAN and wired Internet. Our topology includes one server node, one proxy node and three MNs, i.e. one client node and two peer nodes. The proxy node is co-located with the wireless access point and communication with all mobile nodes using IEEE 802.11b infrastructure mode. On the other hand, the server is connected to the proxy (wireless access point) over wired network.

The bit error rate (BER) for the wireless channel is set to 10^{-5}. Both forward and backward packet loss rate (PLR) for wired link are 3%. The round trip time (RTT) is 20ms for the wireless links and 100ms for the wired links. The transmission range of the WLAN is set to be 275m. We use the mobility model given in Section 4 to generate the realistic mobility patterns, i.e., how long the MN will stay in the range of the access point and how frequently the MN will revisit the access point once it is disconnected from the access point. The actual movement of the MN in the WLAN follows the Random Waypoint model introduced in [22]. The bandwidth of the link between the server and the proxy node is 0.5 Mbps. We assume that in the presence of background traffic, the available bandwidth for streaming service between the server and the proxy is at most 150 kbps. The wireless channel can support data rate up to 1 Mbps, which will be shared among all wireless stations using the same channel. In the simulation, we set the maximum wireless bandwidth available for streaming service as the following: 450 kbps for the proxy-to-client connection and 150 kbps for all peer-to-proxy connections.

Two 300-frame standard video sequences, *foreman* and *container*, are used to drive the simulation. These video sequences are encoded using H.263 version2 at QCIF, with 30 frames per second and 120kps. The I-frame frequency is 1 in 25 frames. For each sequence, PSNR between the original frame i and the reconstructed frame j is calculated for every combinational i and j for $i \leq j$ and saved as a matrix $dArray[i, j]$. The matrix is then used by the simulator to estimate client's performance based on the successfully received data unit.

6.2. Streaming Schemes

To show the benefits of multi-path streaming and explore the two design choices, i.e., *no-cache* vs. *limited cache*, we have implemented the following four streaming schemes:

- Joint Server/Peer with Limited Cache (JSPLC)
- Joint Server/Peer with No Cache (JSPNC)
- Server-only with Limited Cache (SOLC)
- Server-only with No Cache (SONC)

Both JSPLC and JSPNC use our proposed scheme where multiple video sources, including a server on infrastructure-based network and neighboring wireless nodes, can be selected to stream video content to a particular wireless client. Both follow the 3-step solution presented in Section 5, where the proxy not only performs content discovery and identify potential senders, but also requests the video content from multiple senders on behalf of the client before relaying the content to the client. In JSPLC, there is a finite-size cache at the proxy to temporarily store data units it receives from various senders before forwarding them to the client. On the other hand, JSPNC does not use a cache at the proxy, i.e., all data units requested from the sender will be forwarded directly to the client through the wireless access point (the proxy).

In SOLC and SONC, there is only one potential sender, i.e., the server. The proxy will requests video data from the sender on behalf of the client, before relaying the content to the client. SOLC uses a cache at the proxy to temporarily cache some of the more important data units, while waiting for transmission from the proxy to the client. SONC does not use a cache and requires that all data units to be streamed directly from the server to the client.

6.3. Simulation Results

In this section, we will present the simulation results from the perspectives of multi-source diversity, mobility levels, and cache effect.

6.3.1. *Multi-Source Diversity*

In this paper, path diversity is defined as streaming the same video sequence from multiple senders. To evaluate how the multi-source diversity can affect the streaming performance, we compare the

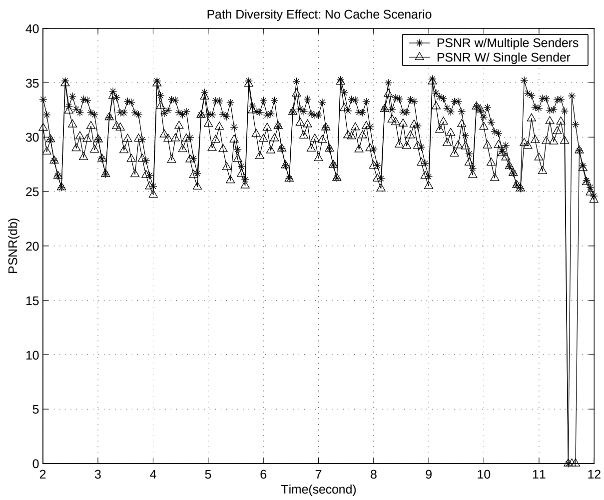

Fig. 3. Effect of multi-source diversity when the connection to the server is not the bottleneck('*container*')

Sequence	*Container*	*Foreman*
PSNR w/o Loss (dB)	34.32	30.71
PSNR w/ Loss Using JSPLC (dB)	31.61	22.60
PSNR w/ Loss Using JSPNC (dB)	30.34	20.85
PSNR w/ Loss Using SOLC (dB)	31.04	21.65
PSNR w/ Loss Using SONC (dB)	28.64	19.11

Table 1. Average streaming performance with and without limited cache

performance of JSPNC and SONC scheme when there is no cache at the proxy.

In the first set of simulation, we use the same simulation setting outlined in Section 6.1. Figure 3 shows the typical performance of the two schemes, JSPNC using multiple senders (server and peers) and SONC using only one sender. In this specific run, we can see that streaming video from neighboring peers (in addition to the server) can improve the received video quality and reduce performance degradation perceived by the receiver.

We repeated the same experiments 40 times, and compute the mean PSNR for each case by averaging the performance over the run-time and 40 runs. Table 1 shows the mean PSNR of the two sequences, *foreman* and *container*, for the four different streaming strategies in the lossy scenario. The mean PSNR in the loss-free scenario is also included for comparison. From Table 1, we can see that JSPNC achieves better performance than SONC for both video sequences. The proposed joint server/peer streaming approach leverages the available source diversity between different potential senders to perform multi-path streaming, leading to better received video quality at the client.

Such multi-path streaming can offer additional advantages. For example, when the remote server that the client is connected to through the wireless access point is overloaded or when there is congestion on the path to the server, the client can avoid quality degradation by requesting the data units from its neighboring peers through ad hoc wireless connections. Second, peer-to-peer streaming with neighboring nodes may incur less delays. To illustrate this, we change the available link capacity between the server and the proxy to 0.1 Mbps, which is less than the required streaming rate to the client and hence representing a scenario where the path to the server is congested. We reran the simulation using this new setting and plot how the PSNR varies over time in Figure 4. The results clearly shows how JSPNC can maintain good performance through multi-path streaming, while SONC suffers great performance degradation.

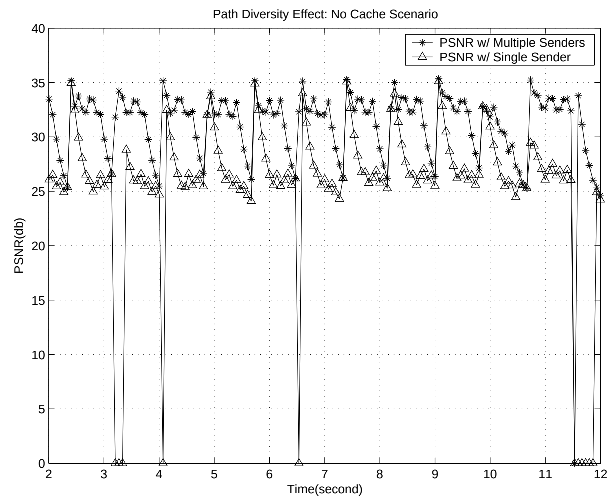

Fig. 4. Effect of multi-source diversity ('*container*') when the connection to the server is the bottleneck

Sequence	*Container*	*Foreman*
PSNR w/ Low Mobility (dB)	31.02	22.83
PSNR w/ Medium Mobility (dB)	30.34	20.85
PSNR w/ High Mobility (dB)	29.63	20.20

Table 2. Average PSNR for Joint Sender-Peer with No Cache (JSPNC) scheme with different levels of peer mobility

6.3.2. Mobility Patterns

In a wireless setting, it is common for mobile nodes (potential senders) to join and leave unexpectedly. Therefore, peer mobility will affect the received video quality for joint sender/peer streaming approach. In this set of simulations, we will perform sensitivity analysis on how the peer mobility affects the streaming performance. We vary the prevalence metric introduced in Section 4 to generate different levels of node mobility as follows. The peer node is said to have *high mobility* when $Prev_j(t) \in [0, 0.25)$, *medium mobility* when $Prev_j(t) \in [0.25, 0.5)$, and low mobility when $Prev_j(t) \in [0.50, 1]$. Again, we average the performance over 40 iterations of a 10-second simulation.

Table 2 shows the video quality perceived by the client under different levels of node mobility when the *Joint Server/Peer with No Cache (JSPNC)* strategy is used. Higher node mobility results in lower mean PSNR. This justifies our design rationale to keep track of peer mobility at the proxy and use this information to determine whether a peer should be included in the candidate sender group (Section 3.1.1).

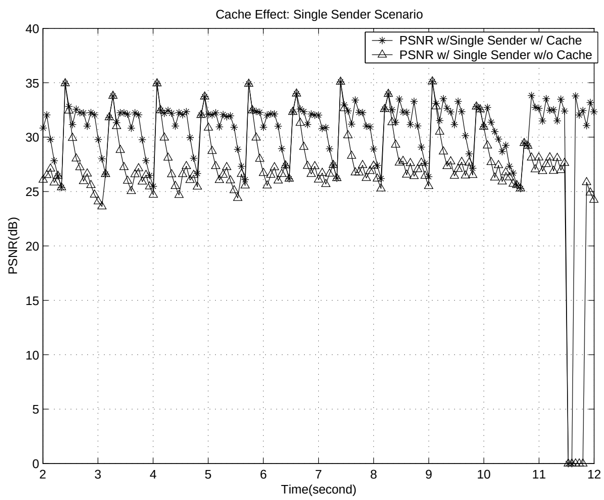

Fig. 5. Effect of cache availability when single source is available ('*Container*')

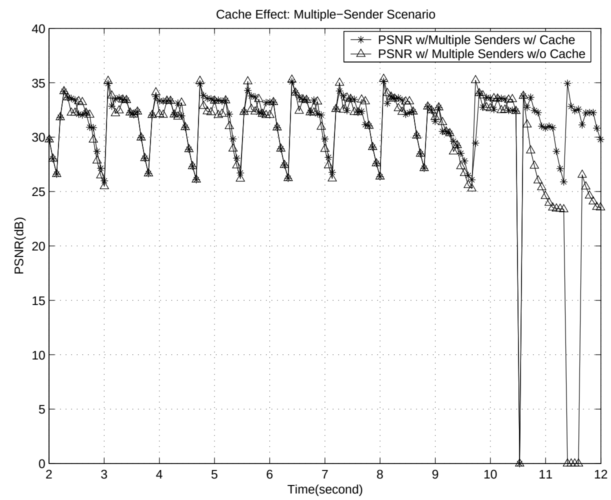

Fig. 6. Effect of cache availability when multi-source strategy is used ('*Container*')

6.3.3. Cache in the proxy

The availability of cache in the proxy can affect the streaming performance in both the single- and multiple-sender cases. In this paper, we measure the cache size in terms of the number of RTP packets. The importance of data units is determined by their respective λ's, and the more important data units are always cached at the proxy. Figure 5 and Figure 6 show the streaming performance when limited cache (i.e., with finite size) is employed in the proxy. Figure 5 compares two schemes: SOLC using limited cache and SONC with no cache. For both SOLC and SONC, there is only one sender, i.e., the media server. In this specific run, we can see that using cache in the proxy can improve the received video quality since it enables local retransmission over the lossy wireless link between the proxy and the client.

To study the effect of cache availability on the performance of multi-source (joint server/peer) streaming schemes, we simulate two schemes: JSPLC using limited cache, and JSPNC with no cache. In Figure 6, we observe that using limited cache also helps improve the streaming performance for multi-source streaming strategy.

We repeated the same experiments 40 times, and compute the mean PSNR perceived by the client to compare the streaming performance of various schemes with and without cache at the proxy. Again, we consider two different sender strategies, server only and multi-source streaming. Table 1 shows the mean PSNR of two standard video sequences, *container* and *foreman*, for the four schemes: JSPLC, JSPNC, SOLC, and SONC. From Table 1, we can see that using limited cache can improve the average streaming perform in both single sender and multi-source scenarios.

7. CONCLUSION AND FUTURE WORK

In this paper, we propose a joint server/peer proxy-driven video streaming scheme to optimize the received video quality from the perspective of a single wireless client, where the proxy coordinates among multiple senders, keeps track of peer mobility patterns and performs the rate-distortion optimization. We formulate the path diversity streaming scheme as a combinational optimization problem and combine the concept of asynchronous clocks and the rate-distortion optimization framework to solve it. To investigate the cache effect on the streaming performance, we model the system in two different ways, *no cache* and *limited cache*. Our NS simulations use the realistic mobility model from the corporate wireless networks to generate the mobility patterns. Numerical investigations through the NS simulations show that our joint scheme can provide better performance than the single sender streaming scheme. The simulation results also show that having the cache in the proxy will potentially increase the streaming performance. Currently, we focus on the performance of single wireless client. As part of our future work, we will address the performance of multiple wireless clients. Moreover, we will investigate the aggregated rate control for streaming over wireless network and how different caching strategies impact the performance as well.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under the CAREER Award No. 0238348 and UC Micro program with matching funds from Hewlett Packard. We are also grateful to Lihua Yuan in the RUBINET group at UC Davis for his assistance in implementing the algorithms using NS-2.

8. REFERENCES

[1] "IEEE STD. 802-11, IEEE 802.11b/d3.0 Standard for Wireless Medium Access Control(MAC) and Physical Layer(PHY) Specification," AUGUST 1999.

[2] D. Li, G. Cheung, C. Chuah, and S.J. Ben Yoo, "Joint server/peer receiver-driven rate-distortion optimized video streaming using asynchronous clocks," in *IEEE ICIP2004*, Oct. 2004.

[3] M. Balazinska and P. Castro, "Characterizing mobility and network usage in a corporate wirelss local-area network," in *First International Conference on Mobile Systems, Applications, and Services (iMobiSys)*, MAY 2003.

[4] "Network simulator," http://www.isi.edu/nsnam/ns/.

[5] D. Qiao and K. G. Shin, "A two-step adaptive error recovery scheme for video transmission over wireless networks," in *IEEE INFOCOM*, June 2000.

[6] X. Xu, M. van der Schaar, S. Krishnamachari, S. Choi, and Y. Wang, "Adaptive error control for fine-granular-scalability video coding over ieee 802.11 wireless lans," in *ICME*, Baltimore, Maryland, USA, July 2003.

[7] S. Krishnamachari, M. van der Schaar, S. Choi, and X. Xu, "Video streaming over wireless LANs: A cross-layer approach," in *The 13 th International Packetvideo Workshop*, Nantes, France, April 2003.

[8] Y. Shan and A. Zakhor, "Cross layer techniques for adaptive video streaming over wireless networks," in *ICME*, Lausanne, Switzerland, August 2002.

[9] A. Majumdar, D. G. Sachs, I. V. Kozintsev, K. Ramchandran, and M. M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs," in *IEEE Transcations on Circuits and Systems for Video Technology*, June 2002, vol. 12, no.6.

[10] T. Nguyen and A. Zakhor, "Distributed video streaming over the internet," in *SPIE Multimedia Computer and Networking*, Jan. 2002.

[11] A. Miu, J. G. Apostolopoulos, W. Tan, and M. Trott, "Low-latency wireless video over 802.11 networks using path diversity," in *ICME*, Baltimore,USA, July 2003.

[12] V. N. Padmannabhan, H. J. Wang, and P. A. Chou, "Resilient peer-to-peer streaming," Microsoft Research Technical Report MSR-TR-2003-11, March 2003.

[13] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," February 2001, Microsoft Research Technical Report MSR-TR-2001-35.

[14] J. Chakareski, P. A. Chou, and B. Girod, "Computing rate-distortion optimized policies for hybrid receiver/sender driven streaming of multimedia," in *Asilomar Conference on Signals, Systems, and Computers*, November 2002.

[15] J. Chakareski and B. Girod, "Server diversity in rate-distortion optimized media streaming," in *ICIP*, Sept. 2003.

[16] F. Chinchilla, M. Lindsey, and M. Papadopouli, "Analysis of wireless information locality and association patterns in a campus," in *IEEE INFOCOM*, March 2004.

[17] A. Gurtov and S. Floyd, "Modeling wireless links for transport protocols," in *ACM SIGCOMM Computer Communication Review*, April, pp. 85–96.

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

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

[20] Minghua Chen and Avideh Zakhor, "Rate control for streaming video over wireless," in *INFOCOM*, March 2004.

[21] D. Tang and M. Baker, "Analysis of a local-area wireless network," in *ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom)*, Boston, Massachusetts, USA, august 2000.

[22] T. Camp, J. Boleng, and V. Davies, "Survey of mobility models for ad hoc network research," in *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking:Research, Trends and Applications*, 2002, vol. 2, no. 5, pp. 483–502.