Smart Media Striping Over Multiple Burst-Loss Channels

Gene Cheung, *Senior Member, IEEE*, Puneet Sharma, *Member, IEEE*, and Sung-Ju Lee, *Senior Member, IEEE*

***Abstract*—We consider a community of multi-homed wireless devices, where each device has both a wireless wide area network (WWAN) interface to connect to the Internet and a wireless local area network (WLAN) interface to connect to its neighbors. Suppose users in the community are interested in receiving the same piece of delay-sensitive media content, and are willing to share their network resources. It is obvious that the community can benefit from the bundling of WWAN links and achieve higher aggregate bandwidth that is not possible with a single user with a single WWAN connection. What is not obvious is that by inverse multiplexing or striping packets across multiple WWAN channels, one can also improve the goodput of delay-sensitive media traffic by striping FEC and ARQ packets across available channels. In this paper, we analyze the potential benefits of striping media traffic and develop algorithms that take advantage of these benefits to optimize the delivery of delay-sensitive media streams to a wireless multi-homed device community. Results show dramatic improvement over naïve striping schemes such as weighted round robin both in terms of packet loss ratio, and in terms of peak signal-to-noise ratio for H.264 video streaming.**

***Index Terms*—Multimedia streaming, packet striping, wireless networks.**

I. Introduction

It is now commonplace for modern wireless devices to be multi-homed—each having both a wireless wide area network (WWAN) interface to connect to the Internet via a cellular network basestation, and a wireless local area network (WLAN) interface to connect to the neighboring and similarly configured wireless devices. While WWAN links such as 3G networks' UMTS [1] remain comparatively limited in bandwidth, slow in transmission and burst-loss-prone in packet delivery, WLAN links such as 802.11x are, in contrast, plentiful, fast, and reliable. Though WLAN links can provide media-streaming capable high-speed Internet connectivity, it requires the availability of an access point connected to a high-speed, (mostly) wired connection. The users have to rely on bandwidth limited WWAN connections in the areas where the coverage of public access points is absent. A collaborative resource sharing approach has been proposed as a complimentary mechanism for high-speed access in such conditions [2]. Aggregated bandwidth channels can be realized only when hosts willingly collaborate by sharing their communication channels. Willingness to collaborate is not an issue for a single user with multiple mobile devices (e.g., cell phone, PDA, laptop, etc.) forming a community, (i.e., personal area network), nor might it be an issue for colleagues or acquaintances. But there need to be incentives for collaboration between hosts owned by multiple parties with little or no pre-existing relationship. Clearly, if many community members seek access to the same content (e.g., multicast video) then the members will be well-motivated to take advantage of faster download or streaming.

Suppose then a *community* of multi-homed wireless users are interested in receiving the same piece of *delay-sensitive* content such as a video stream, and are willing to share their network resources to achieve their common goal. To maximize the usage of the community's available WWAN links, one can first divide the incoming packet stream into smaller substreams at a gateway located at the junction of wired and wireless WAN networks, and inverse-multiplex or *stripe* them across the community's WWAN links. Upon receiving packets of a substream, each user will forward them to others in the community for stream recomposition via its high-speed WLAN links. It is obvious that such striping framework for a community of wireless multi-homed users benefits from the aggregation of the community's WWAN bandwidths, enabling streaming of high bandwidth content that is not otherwise possible for an individual user with a single bandwidth-limited WWAN link. See Fig. 1 for an illustration. Similarly, a device with multiple WWAN connections can bundle its multiple low bandwidth channels together to reap benefits of bandwidth aggregation.

Such striping framework can not only provide higher bandwidth for streaming applications, but the additional channels can also be used for error correction. Error correction is of paramount importance here, given that the WWAN connections are prone to burst loss and that the delay sensitivity of streaming traffic allows only a very limited time window for error correction. It turns out that intelligent assignment of error correction in a striping scenario, either forward error correction (FEC) or retransmissions (ARQ), can greatly improve the timely delivery (goodput) of delay-sensitive media traffic. For FEC, similar to a single channel packet interleaver, striping FEC packets across multiple channels can avoid decoding failure due to a single burst loss. Yet unlike the interleaver, striping avoids excessive transmission delay of long interleaving in a single channel. We call this striping benefit the *interleaving effect*. For ARQ, given a packet's delivery deadline, striping empowers one with the ability to select among multiple WWAN channels for packet transmission, each with different channel characteristics in delay and loss. One can then judiciously select a channel

Manuscript received November 9, 2006; revised May 2, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yucel Altunbasak.

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

P. Sharma and S.-J. Lee are with the Mobile & Media Systems Lab, Hewlett-Packard Laboratories, Palo Alto, CA 94304 USA (e-mail: puneet.sharma@hp.com; sungju.lee@hp.com).

Digital Object Identifier 10.1109/JSTSP.2007.902069

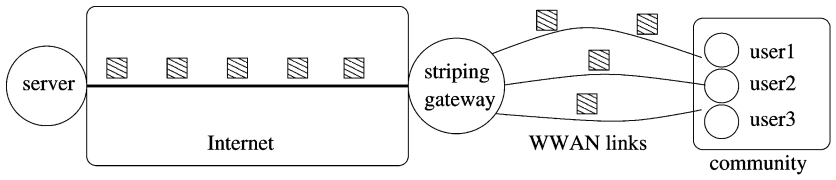


Fig. 1. Overview of the packet striping system.

that optimizes the packet's chance of survival—one that maximizes its successful transmission probability *and* its chance for retransmission if the current transmission fails. We call this striping benefit the *channel selection effect*.

The goal of the paper is to closely examine the potential benefit of the interleaving effect and the channel selection effect for striping of delay-sensitive packets. In particular, our contributions are the following.

1) To pinpoint the performance of a striped FEC block, we mathematically derived expressions for the packet loss ratio of an FEC block striped across multiple independent burst-loss channels.
2) To exploit the potential of the interleaving effect, we devised a heuristic-based fast-converging greedy algorithm that stripes an FEC block across multiple independent burst-loss channels.
3) To exploit the potential of the channel selection effect, we devised an ARQ-based algorithm that stripes incoming and previously lost delay-sensitive packets across lossy channels in a bandwidth-limited system.
4) To exploit the potential of the interleaving effect in a bandwidth-limited system, we devised an FEC-based algorithm that selects the appropriate FEC block for incoming delay-sensitive media traffic and stripes them across multiple burst-loss channels.
5) We combined the ARQ-based and FEC-based algorithms into an hybrid algorithm that selects the right mixtures of FEC and ARQ and stripes them across multiple channels in a bandwidth-limited system. We devised an appropriate penalty function to drive the system towards optimal behavior.

The rest of the paper is organized as follows: Section II discusses background and related work. Section III discusses the modeling of burst-loss channels and basic definitions. Section IV derives the effective packet loss ratio (PLR) when FEC Reed-Solomon code $\mathrm{RS}(n,k)$ is applied to a single bursty channel. Section V derives PLR when $\mathrm{RS}(n,k)$ is striped over a set of m independent burst-loss channels under a particular mapping. A fast heuristic algorithm that finds a good FEC mapping is also developed. Striping on bandwidth-limited, bursty channels is analyzed, and optimization algorithms are designed in Section VI for the ARQ-based algorithm and in Section VII for the FEC-based and hybrid FEC/ARQ algorithms. Section VIII presents two important enhancements towards real-time implementation for striping of multimedia traffic: i) a recursive procedure to optimize multiple input packets at a time using developed algorithms for a single packet; and ii) a two-tier dynamic programming implementation that reduces the computational complexity at the cost of solution quality. Experimental results that compared our derived striping schemes with common striping schemes in the literature such as weighted round robin—including video streaming experiments that used MPEG test sequences as inputs to the striping system—are presented in Section IX. Finally, concluding remarks are presented in Section X.

II. Background

As shown in Fig. 1, striping is the mapping of a single flow to multiple channels. While fair load sharing among multiple channels is a concern, effective traffic mapping of delay-sensitive media packets onto the channels for optimized performance (delay-bounded goodput) is also critical—this is the sole focus of this paper. In particular, we assume packets entering the striping gateway are each marked with a *delivery deadline*, before which time the packet must be delivered to the clients or the packet is rendered useless. Delivery deadlines are the only application level details exposed to the striping gateway; it is our goal to show that even with this simple level of abstraction, it is sufficient for the striping gateway to dramatically improve the delivery of delay-sensitive packets without resorting to more computationally intensive cross-layer optimizations like real-time transcoding [3].

It is clear that the receiving end of the striping gateway must resynchronize out-of-order delivery packets; we assume the existence of reassembly mechanisms that handle reordering of packets. Applications such as media streaming use receiver side buffers that can also be used for packet reordering. We additionally assume the packet size and the transmission rate are constant. The wireless channels are always available, although they will sometimes be lossy. In other words, the disappearance of the channels due to mobility of the end hosts is not considered.

There are related works in different areas: striping over wireless channels, modeling wireless channels, and media streaming over single wireless channel or multiple wired paths. However, very few model striping delay-sensitive media packets across multiple wireless channels. We review the three categories in order.

Adaptive inverse multiplexing for Cellular Digital Packet Data (CDPD) wireless networks is proposed in [4]. In this scheme the packets are split into fragments of size proportional to the observed throughput of component links. The fragment size of each link is dynamically adjusted in proportion to the measured throughput. The bandwidth of mobile users with multiple interfaces is aggregated at the transport layer in pTCP (parallel TCP) [5]. pTCP establishes virtual TCP connection for each interface and performs striping based on congestion window size of each virtual TCP connection. A scheduling algorithm for aggregating bandwidth for real-time applications is detailed in [6]. The authors propose a Earliest Delivery Path First (EDPF) scheduling algorithm that is channel and application aware and minimizes the cost of striping traffic over multiple wireless channels of a device. The commuter Mobile Access Router (MAR) [7] leverages wireless WAN connection diversity to provide high speed Internet access to mobile users. Instead of using the WAN connections of the users, it relies on preprovisioning the MAR with different WAN connections, limiting the aggregation to the already exiting links.

Modeling the wireless channel behavior has been an active research area. Wireless channel is modeled using the traces in [8]. Bursty errors are modeled using two-state Markov chain and two variations. The length of errors is shown to have two exponential curves and the length of error-free packets has a combination of two Pareto distributions and one exponential curve. TCP throughput over bursty losses is analyzed in [9]. It models TCP's fast retransmit and timeout mechanism's impact on TCP performance. The authors argue that the timeouts have a large effect on TCP throughput. TCP throughput over random losses is studied in [10]. It shows that random losses degrade TCP performance significantly when the product of loss rate, the normalized asymmetry, and the square of the bandwidth-delay product is large. TCP performance in wireless channels with random and bursty losses is modeled using a continuous time finite state Markovian Chain in [11]. TCP over Rayleigh fading wireless channels, along with ARQ-based link level recovery are considered in [12]. This work shows that for end-to-end paths that are composed with both wired and wireless links, link level recovery schemes improve TCP performance. Type-II hybrid ARQ over wireless bursty channels is analyzed with 16-state Markov chain in [13]. Allocation of packets on parallel channels to improve the error protection for best-effort traffic has been studied earlier in [14], [15].

Streaming over lossy channels creates another challenge as packets are delay-sensitive. Using a burst-loss model, performance analysis of a MPEG-2 streaming system using FEC over a single lossy channel is presented in [16]. Optimal MPEG-2 source rate and FEC packet rate for minimizing video distortion is derived. Streaming packet scheduling over wireless channels has been investigated in several papers. An opportunistic scheduling is proposed in [17] where the channel state and the utility function are considered. Their goal is to minimize delay and also enforce fairness. FEDD (Feasible Earlier Due Date) scheduling is proposed in [18]. This policy selects the packet whose expiry is the earliest and the channel is in good state. The authors have shown that FEDD performs better than the longest-queue-first scheduling. A frame-based and a motion-texture discrimination-based scheduling algorithms are proposed in [19]. Packets are scheduled based on deadline thresholds, which are assigned to video packets based on importance of packets. A scheme proposed in [20] also uses packet priority when scheduling packets. Channel condition is also factored-in for their scheduling algorithm. Point-to-point rate-distortion optimized packet scheduling in lossy channels is thoroughly analyzed in [21]. It is shown that rate-distortion optimized scheduling of the entire session can be solved by isolating error-cost optimized transmission of a single data. EDBS (Expected Runtime Distortion Based Scheduling) for layered streaming video in lossy channels is presented in [22]. Using a fast greedy algorithm, it estimates the importance of each packet and schedules the packets based on the importance. FEC and ARQ performances in continuous streams over bursty channels are compared in [23]. This study shows that ARQ schemes perform better in most cases.

Streaming real-time traffic over multiple paths is also a well-studied subject (see [24] and references within), but most of these work consider paths in the wired Internet where significant burst loss events on delivery paths are not common as in wireless channels.

In our previous work, we have examined performance of striping over multiple burst-loss channels with constant delays [25] and random delays [26]. A recent work [27] proposed two modifications: i) an ad-hoc weighting function to modify the objective function in order to drive the striping system away from pathological `local` minimum; and ii) a two-tier dynamic programming technique to speed up the implementation of the developed striping algorithms. Our current work presents a series of refinements upon previous work. First, the independent assumption of data and parity packets previously used for the calculation of packet loss ratio of a Reed–Solomon code $\mathrm{RS}(n,k)$ striped over a set of m channels is removed, making the new calculation more accurate. Second, a new greedy search algorithm called `local` for finding a good FEC distribution of $\mathrm{RS}(n,k)$ packets over m channels is discussed. We will show that `local` is point-by-point better than other greedy search algorithms we have previously developed. Third, instead of estimating the queuing time of a channel i by counting the number of packets currently in the outgoing queue of the channel, the queuing delay of a channel i is more accurately estimated by recording the time when the most recent packet that entered the queue would exit and free up the queue. Fourth, the selection of the strength of FEC—n and k in $\mathrm{RS}(n,k)$—is restricted to ones whose ratios of total packet to source packet n/k do not exceed the ratio of aggregate channel packet rate to incoming packet rate. Experiments show that the combination of these two refinements eliminates the need for ad-hoc weighting functions [27] that drives the system away from poorly performing `local` minima. Fifth, optimization algorithms are generalized from one packet to all packets at the head of the incoming queue at optimization instance. This is needed as media data like a video frame is often segmented into multiple packets, each having the same delivery deadline, and it is imperative that all packets arrive at the client, not just the first one. Finally, we demonstrate the efficacy of the striping system in the context of a H.264 video streaming scenario and show its performance in peak signal-to-noise ratio (PSNR).

III. Channel Model Basics

We begin our study with an introduction of our network loss model and definitions of basic terms that will be used for analysis and derivation of striping algorithms in later sections. Note that the derivation in Setions III and IV was first presented in [28]; for the sake of completeness we will nevertheless present our notations which differ slightly.

Given the burst-loss nature of wireless links, we model losses in each channel using a two-state Markov chain (Gilbert model), shown in Fig. 2. A correct (incorrect) packet delivery event is denoted by 0 (1).

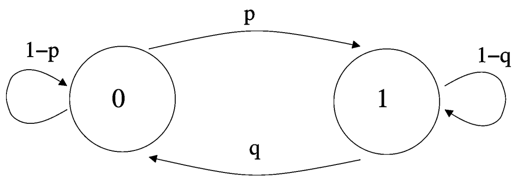


Fig. 2. Gilbert loss model.

We next define basic terms similar to those introduced in [16]. Let p and q be the Gilbert model parameters. Let $p(i), i \geq 0$, be the probability of having *exactly* i consecutive correctly delivered packets between two lost packets, following an observed lost packet, i.e., $p(i) = \Pr(0^i|1)$. Let $P(i)$ be the probability of having *at least* i consecutive correctly delivered packets, following an observed lost packet, i.e., $P(i) = \Pr(0^i|1)$. $p(i)$ and $P(i)$ can be written mathematically:

$$p(i) = \begin{cases} 1-q, & \text{if } i = 0 \\ q(1-p)^{i-1}p & \text{otherwise} \end{cases} \tag{1}$$

$$P(i) = \begin{cases} 1, & \text{if } i = 0 \\ q(1-p)^{i-1}, & \text{otherwise} \end{cases} \tag{2}$$

$$q(i) = \begin{cases} 1-p, & \text{if } i = 0 \\ p(1-q)^{i-1}q, & \text{otherwise} \end{cases} \tag{3}$$

$$Q(i) = \begin{cases} 1, & \text{if } i = 0 \\ p(1-q)^{i-1}, & \text{otherwise} \end{cases}. \tag{4}$$

$q(i)$ and $Q(i)$ are complementarily defined functions: $q(i) = \Pr(1^i0|0)$ and $Q(i) = \Pr(1^i|0)$.

We next define $R(m,n)$ as the probability that there are *exactly* m lost packets in n packets, following an observed lost packet. It can be expressed recursively using earlier definitions as shown in (5) at the bottom of the page.

We additionaly define $r(m,n)$ as the probability that there are *exactly* m loss packets in n packets *between* two lost packets, following an observed lost packet. Similarly, $r(m,n)$ can be expressed recursively as shown in (6) at the bottom of the page.

Finally, we define $\bar{r}(m,n)$ as the probability that there are *exactly* m lost packets in n packets, following an observed lost packet and preceding a successfully received packet.

$$\bar{r}(m,n) = R(m,n) - r(m,n) \tag{7}$$

We define the complementary function $S(m,n)$, as the probability of having *exactly* m correctly received packets in n packets following an observed correctly received packet. See (8), shown at the bottom of the page. $s(m,n)$ and $\bar{s}(m,n)$ are defined counterparts to $r(m,n)$ and $\bar{r}(m,n)$.

IV. FEC for One Burst-Loss Channel

Various error correction and retransmission schemes can be used for improving the data delivery in high loss environments. In this paper, we consider and evaluate the performance of two such schemes, namely, forward error correction (FEC) and automatic repeat request (ARQ). In this section we model the impact of FEC on data delivery ratio over one bursty loss channel.

Given the network model and definitions introduced in the previous section, we can now derive the expected packet loss ratio (PLR) of FEC code—α_{RS} of Reed-Solomon code $\text{RS}(n,k)$ in particular—on a burst-loss channel. Reed-Solomon code is commonly used in practice for FEC packet-level recovery systems with delay constraints [29]–[31]. Fig. 3 shows an example of an RS(5, 3) code. Note, however, that our analysis holds valid for all other maximum distance separable codes besides RS. Our choice of RS stems both from its wide acceptance and its many available fast implementations, including Newton's Interpolation [32]. As shown in [33], the complexity of Newton's Interpolation is $(k-1)((k)/(2)+u)$, where $u = n-k$ at the encoder and u is the number of lost data packets at the decoder. It is negligible for the small ranges of n and k we use in the to-be-discussed FEC algorithms in Section VII.

Recall $\text{RS}(n,k)$ is correctly decoded if any k packets of the group of k data and $n-k$ parity packets are correctly received. First, we condition on the status of the last transmitted packet (loss/success), i.e., the state of the channel as shown as `condition 1` in Fig. 3, giving us two conditional probabilities, $\alpha_{\text{RS}\,|\,1}$ and $\alpha_{\text{RS}|0}$, respectively. α_{RS} can then be expressed as

$$\alpha_{\text{RS}} = \pi * \alpha_{\text{RS}|1} + (1-\pi) * \alpha_{\text{RS}\,|\,0} \tag{9}$$

where $\pi = (p)/(p+q)$ is the raw PLR of the channel.

To find $\alpha_{\text{RS}|1}$, we consider the k data packet block and the $n-k$ parity packet block separately. We condition on the status

$$R(m,n) = \begin{cases} P(n), & \text{for } m = 0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i)R(m-1,n-i-1), & \text{for } 1 \leq m \leq n \end{cases} \tag{5}$$

$$r(m,n) = \begin{cases} p(n), & \text{for } m = 0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i)r(m-1,n-i-1), & \text{for } 1 \leq m \leq n \end{cases} \tag{6}$$

$$S(m,n) = \begin{cases} Q(n), & \text{for } m = 0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} q(i)S(m-1,n-i-1), & \text{for } 1 \leq m \leq n \end{cases} \tag{8}$$

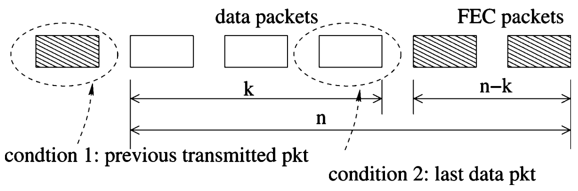

data packets
FEC packets
k
n−k
n
condtion 1: previous transmitted pkt
condition 2: last data pkt

Fig. 3. FEC encoding of data packets.

of the last (kth) data packet; given the kth data packet is lost or received, we use $R(\cdot,\cdot)$ or $S(\cdot,\cdot)$ for probability calculation of the trailing $n-k$ parity packet block.

Conditioning on the event when the kth data packet is lost (`condition 2` in Fig. 3), we consider all cases when any number i of the remaining $k-1$ data packets are lost. Each case i will have a loss ratio of $(i+1)/(k)$, assuming there are $\geq n-k+1$ total loss packets including the $n-k$ parity packets. Similar analysis conditioning on the event when the kth data packet is successfully received completes the derivation for $\alpha_{\mathrm{RS}|1}$:

$$\begin{aligned}\alpha_{\mathrm{RS}|1} = &\sum_{i=0}^{k-1}\left(\frac{i+1}{k}\right) r(i,k-1)\\ &\times \sum_{j=[n-k-i]^+}^{n-k} R(j,n-k)\\ &+\sum_{i=1}^{k-1}\left(\frac{i}{k}\right)\bar{r}(i,k-1)\\ &\times \sum_{j=[n-k+1-i]^+}^{n-k} S(n-k-j,n-k) \qquad (10)\end{aligned}$$

where $[x]^+$ is the positive part of x. Following similar analysis for $\alpha_{\mathrm{RS}|0}$ we get

$$\begin{aligned}\alpha_{\mathrm{RS}\,|\,0} = &\sum_{i=0}^{k-1}\left(\frac{i+1}{k}\right)\bar{s}(k-1-i,k-1)\\ &\times \sum_{j=[n-k-i]^+}^{n-k} R(j,n-k)\\ &+\sum_{i=1}^{k-1}\left(\frac{i}{k}\right) s(k-1-i,k-1)\\ &\times \sum_{j=[n-k+1-i]^+}^{n-k} S(n-k-j,n-k). \qquad (11)\end{aligned}$$

V. Striping FEC for Multiple Burst-Loss Channels

Having derived the PLR of a given $\mathrm{RS}(n,k)$ on a single burst-loss channel, we now extend the analysis to derive the PLR of a particular "stripe" of a given $\mathrm{RS}(n,k)$ on m burst-loss channels. We call the mapping or "stripe" of k data and $n-k$ parity packets to m bursty channels an *FEC distribution*. We denote such mapping function as $\mathrm{g} : (k, n-k) \rightarrow (\mathbf{u},\mathbf{v}), \mathbf{u},\mathbf{v} \in \mathcal{I}^m$. It is a mapping of two scalars to two vectors of length m, where $u_i(v_i)$ represents the number of data packets (parity packets) assigned to channel i.

Let random variable X be the number of unrecoverable data packets at the receiver in k data packets in an $\mathrm{RS}(n,k)$ code. Let Y, Z be the number of correctly transmitted data packets and parity packets, respectively. X, Y and Z are related as follows:

$$X = \begin{cases} k-Y & \text{if } Y+Z \leq k-1 \\ 0 & \text{o.w.}\end{cases} \qquad (12)$$

If given joint probability mass function (pmf) of Y and Z, $\mathbb{P}(Y,Z)$, we can find the expectation of X as

$$\begin{aligned}E[X] &= E[k-Y\,|\,Y+Z\leq k-1]\mathbf{P}(Y+Z\leq k-1)\\ &= \sum_{y=0}^{k-1}(k-y)\mathbf{P}(Y=y, Z\leq k-1-y)\\ &= \sum_{y=0}^{k-1}(k-y)\sum_{z=0}^{k-1-y}\mathbf{P}(Y=y,Z=z). \qquad (13)\end{aligned}$$

To find $\mathbf{P}(Y,Z)$, we first define random variables $Y_i \leq u_i$ and $Z_i \leq v_i$ as the number of correctly transmitted data packets and parity packets in channel i, respectively. We can then write

$$Y = \sum_{i=1}^{m} Y_i, \quad Z = \sum_{i=1}^{m} Z_i \qquad (14)$$

For each channel i, joint pmf of Y_i and $Z_i, \mathbf{P}_i(Y_i, Z_i)$, can be written as one of the two following forms. If $u_i = 0$, then $\mathbf{P}_i(0, Z_i)$ is simple:

$$\mathbf{P}_i(0, Z_i = z) = \pi_i R_i(u_i - z, v_i) + (1-\pi_i) S_i(z, v_i) \quad (15)$$

If $u_i \geq 1$, then $\mathbf{P}_i(Y_i, Z_i)$ is

$$\begin{aligned}&\mathbf{P}_i(Y_i = y, Z_i = z)\\ &\quad = \mathbf{1}(u_i > y)[\pi_i r_i(u_i-1-y, u_i-1) R_i(v_i - z, v_i)\\ &\qquad + (1-\pi_i)\bar{s}_i(y, u_i-1) R_i(v_i - z, v_i)]\\ &\qquad + \mathbf{1}(y>0)[\pi_i \bar{r}_i(u_i - y, u_i - 1) S_i(z, v_i)\\ &\qquad + (1-\pi_i) s_i(y-1, u_i-1) S_i(z, v_i)] \qquad (16)\end{aligned}$$

where $y = 0,\ldots,u_i, z = 0,\ldots,v_i$ and $\mathbf{1}(\mathbf{c}) = 1$ if clause $\mathbf{c}$ is true, and $= 0$ otherwise. Since Y and Z, are both sums of random variables, we derive joint pmf of $\mathbf{P}(Y,Z)$ using probability generating function (pgf) $\mathbf{G}_{Y,Z}(\xi,\zeta)$:

$$\begin{aligned}\mathbf{G}_{Y,Z}(\xi,\zeta) &= E[\xi^Y \zeta^Z]\\ &= \sum_y\sum_z P(Y=y, Z=z)\xi^y\zeta^z\\ &= E[\xi^{Y_1+\cdots+Y_m}\zeta^{Z_1+\cdots+Z_m}]\\ &= E[\xi^{X_1}\zeta^{Z_1}]\ldots E[\xi^{X_m}\zeta^{Z_m}]\\ &= \mathbf{G}_{Y_1,Z_1}(\xi,\zeta)\ldots\mathbf{G}_{Y_m,Z_m}(\xi,\zeta).\end{aligned}$$

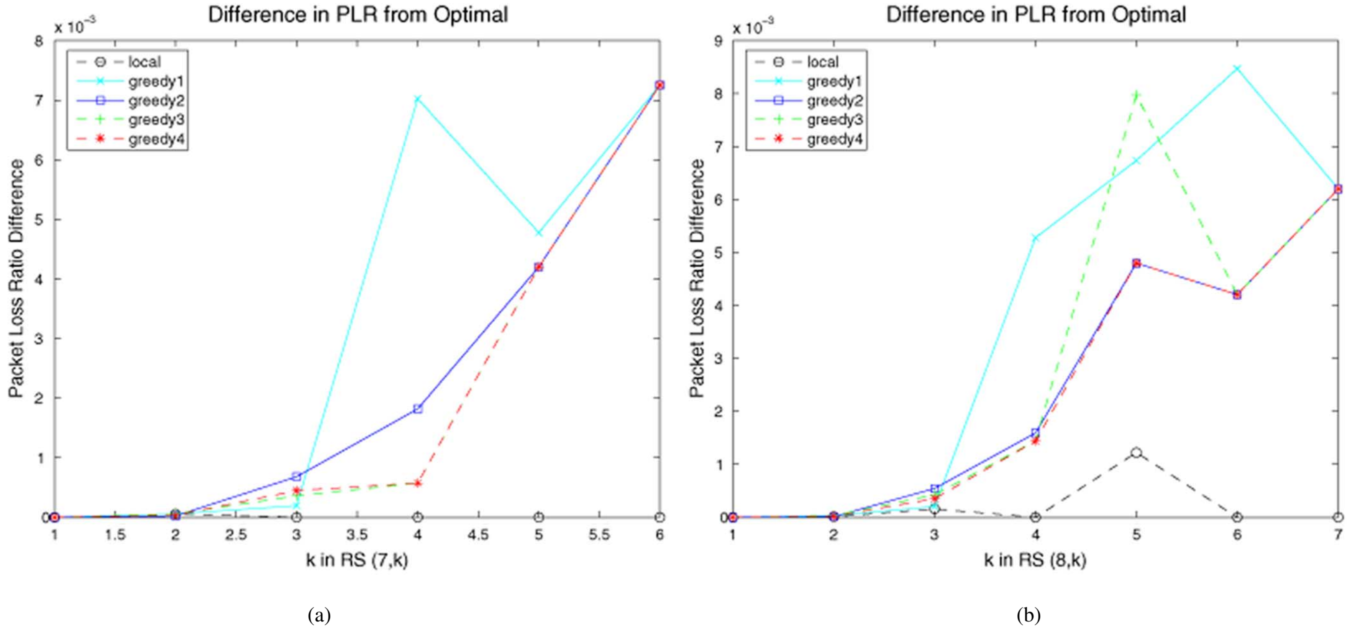


Fig. 4. PLR for different FEC distribution search algorithms. (a) PLR for RS(7, k). (b) PLR for RS(8, k).

Hence joint pgf $\mathbf{G}_{Y,Z}(\xi,\zeta)$ is simply a product of joint pgfs $\mathbf{G}_{Y_i,Z_i}(\xi_i,\zeta_i)$'s. We recover joint pmf $\mathbf{P}(Y,Z)$ from joint pgf $\mathbf{G}_{Y,Z}(\xi,\zeta)$ as follows:

$$\mathbf{P}(Y=y,Z=z)=\frac{1}{y!}\frac{1}{z!}\frac{d^y}{d\xi^y}\frac{d^z}{d\zeta^z}\mathbf{G}_{Y,Z}(\xi,\zeta)\bigg|_{\xi=0,\zeta=0}. \tag{17}$$

We can now derive $E[X]$ using (13). We denote $\pi(\mathrm{g})$ as $E[X]/k$—PLR given mapping g for $\mathrm{RS}(n,k)$.

A. Fast FEC Distribution Search Algorithms

Given $\mathrm{RS}(n,k)$, the number of unique mappings of k data packets to m channels can be shown to be exponential in m. Together with the mappings of $n-k$ parity packets to m channels, the total number of unique FEC distributions grows faster than exponential growth rate. For large values of m and k, exhaustively searching through all possible FEC distributions is impractical. In such cases, we need a computation-efficient algorithm to find a good FEC distribution.

We now explore practical *greedy* algorithms to select good FEC distributions. A greedy algorithm is an algorithm that iteratively makes the most profitable selection locally at each turn until an ending condition is met. The first greedy algorithm greedy1 first allocates one data packet to the *optimum* channel—channel in which adding the additional packet will result in the smallest PLR. It then allocates one parity packet to the optimum channel, then the rest of the data packets one at a time to the optimum channel, and then the rest of the parity packets. greedy2 allocates one data packet to the optimum channel, all the parity packets one at a time to the optimum channel, and then the rest of the data packets. greedy3 allocates data and parity packets alternatively to optimum channel when possible. greedy4 allocates data and parity packets alternatively in small bundles, proportional to the ratio of data to parity packets.

Taking a different approach, local begins with an initial FEC distribution (to be discussed), then iteratively finds and applies the most profitable data/parity packet movement—one where the reallocation of one data/parity packet from one channel to another would result in the largest decrease in PLR. local continues the packet reallocations until no profitable packet movement can be found. Obviously, such greedy local search depends heavily on the initial FEC distribution; we use two extreme initializations—one where all packets are assigned to the single channel with lower raw PLR, and another where all packets are evenly distributed among all channels—and use the lower of the two resulting PLRs as the performance point of local.

To compare these greedy algorithms, we set the parameters of three burst-loss channels as (0.05,0.45), (0.03,0.27) and (0.05,0.4), and we calculated PLR for these algorithms for $\mathrm{RS}(n,k)$, $1\leq k<n\leq 8$. The resulting average effective PLRs over the possible FECs are shown in Table I. We compare their performance with the optimal FEC distribution, found by exhaustive search optimal. We observe that local is by far the best greedy performer. In fact, when we plot the difference in effective PLR compared with optimal in Fig. 4 for $\mathrm{RS}(7,k)$ and $\mathrm{RS}(8,k)$, we see that local is point-by-point better than all other greedy algorithms. We conjecture the reason as the following: $\mathrm{RS}(n,k)$ inherently behaves much differently than $\mathrm{RS}(n-1,k)$ or $\mathrm{RS}(n-1,k-1)$, and hence it is better to start with an initial $\mathrm{RS}(n,k)$ distribution and reallocate packets rather than grow a distribution one packet at a time. Henceforth we will use local as our heuristic for constructing FEC distribution. We will later show in Section IX-A2 that local does in fact perform close to the optimal exhaustive search for all practical purposes. As for complexity, though, in the theoretical worst case local has exponential running time like exhaustive search optimal, we found in the above experiments that local in practice converged quickly in a handful of iterations.

TABLE I
AVERAGE PLR FOR FEC DISTRIBUTION SEARCH ALGORITHMS

Algorithm	greedy1	greedy2	greedy3	greedy4	local	optimal
Avg PLR	0.0183	0.0176	0.0174	0.0177	0.0145	0.0143

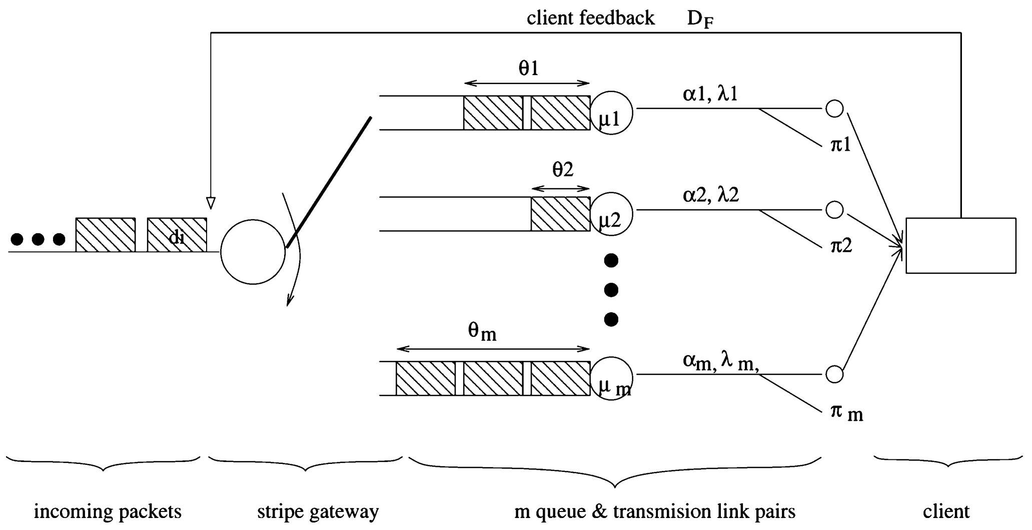


Fig. 5. Bandwidth-limited network model.

VI. STRIPING DELAY-SENSITIVE MEDIA TRAFFIC OVER BANDWIDTH-LIMITED CHANNELS

As mentioned in the Introduction, besides burst losses, realistic WWAN channels are also comparatively bandwidth-limited. Given the delay-sensitivity of the media traffic, it is important to consider not only the packet loss rate but also the delivery deadlines while arriving at the striping schedule. To incorporate the bandwidth-limited dynamics for realtime traffic into our analysis when studying the interleaving and channel selection effects of striping, we expand the Gilbert loss model in Fig. 2 to a bandwidth-limited, burst-loss model with random delays as shown in Fig. 5. Each j of m channels is modeled by a FIFO queue and transmission link pair: a queue with constant service rate μ_j is connected to a transmission link of shifted-Gamma-distributed random variable delay $\gamma_j \sim \mathcal{G}(\kappa_j, \alpha_j, \lambda_j)$ and Gilbert-modeled burst loss of parameters p_j and q_j. A low-bandwidth WWAN channel is modelled by a correspondingly small service rate. We assume the packet gateway records the time t_j at which the most recent packet that entered queue j would exit the queue. The queuing delay experienced by a packet entering queue j at optimization instant t is then $\theta_j = 1/\mu_j + \max(t_j - t, 0)$; new t'_j will need to be subsequently updated for the next optimization instant as: $t'_j = \max(t_j, t) + 1/\mu_j$.

The time required to transmit the packet through queue j is then simply the sum of the queuing and transmission delay: $\theta_j + \gamma_j$. Detailing the transmission delay, a Gamma random variable γ with Gamma shape parameter α and scale parameter λ has the following probability density function (pdf) ([34, p. 117]):

$$g_\Gamma(\gamma) = \frac{\lambda(\lambda\gamma)^{\alpha-1}e^{-\lambda\gamma}}{\Gamma(\alpha)} \quad 0 < \gamma < \infty \tag{18}$$

where $\Gamma(\alpha)$ is the *Gamma function*:

$$\Gamma(\alpha) = \int_0^\infty \tau^{\alpha-1}e^{-\tau}d\tau \quad \alpha > 0. \tag{19}$$

Similarly, the shifted version of the Gamma random variable with shift parameter κ is

$$g_{\Gamma_s}(\gamma) = \frac{\lambda^\alpha(\gamma-\kappa)^{\alpha-1}e^{-\lambda(\gamma-\kappa)}}{\Gamma(\alpha)} \quad \kappa < \gamma < \infty. \tag{20}$$

In addition, we assume the client can inform the striping gateway of a loss event losslessly in constant time D_F.

For input into the striping gateway, we assume the packets in the incoming queue before the striping gateway are labeled with expiration times d_i's. A packet with d_i must be delivered by time d_i or it expires and becomes useless. In other words, at optimization time t, a packet has *packet delay tolerance* $(d_i - t)$—the amount of time the gateway has to deliver the packet to the client. We assume the packets are ordered in the incoming queue by earliest expiration times. We assume striping gateway is activated whenever there is a packet in the incoming queue.

Availability of multiple channels in a striping system allows the use of error correction and packet retransmission, especially in high loss channels. Depending on the channel characteristics and delay tolerance of the real-time traffic, FEC and ARQ can be used to improve the data delivery ratio. We now analyse the impact of FEC and ARQ on delivery of striped delay-sensitive media traffic.

A. ARQ-Based Algorithm

We first develop an *ARQ-based algorithm* to exploit the channel selection effect of packet striping. We assume for now that we are optimizing the first packet at the front of the input queue with expiration time d. Let $f(d'), d' = d - t$, be the probability that a packet with packet delay tolerance d' is timely delivered to the client. Let $f_{\text{ARQ}}(d')$ be the probability that the same packet is timely delivered using (re)transmission (ARQ). Let $f^{(i)}_{\text{ARQ}}(d')$ be the probability that the same packet is timely delivered if channel i is first used for ARQ. Given the client can losslessly inform the gateway of the loss event in time D_F, the packet has a chance for retransmission with a tighter deadline. We can write

$$\begin{aligned} f(d') &= \begin{cases} f_{\text{ARQ}}(d') & \text{if } d' \geq 0 \\ 0 & \text{o.w.} \end{cases} \\ f_{\text{ARQ}}(d') &= \max_{i=1,\ldots,m} f^{(i)}_{\text{ARQ}}(d') \\ f^{(i)}_{\text{ARQ}}(d') &= \int_{\kappa_i}^{d'-\theta_i} g_{\Gamma_s}(\gamma)\left((1-\pi_i)\right. \\ &\quad \left. + \pi_i f(d' - D_F - \theta_i - \gamma)\right) d.\gamma \end{aligned} \tag{21}$$

The interval over which the integral is taken is written as such, because $g_{\Gamma_s}(\gamma)$ is zero for transmission $\gamma < \kappa_i$, and the packet in question will miss its deadline d for $\gamma > d' - \theta_i$.

B. Quantization & Dynamic Programming

As (21) is defined recursively within an integral, it is difficult to solve directly. Hence we first approximate (21) using *quantization*, before using *dynamic programming* to resolve the recursive calls. By quantization, we mean we divide the non-zero area under pdf $g_{\Gamma_s}(\gamma), \gamma \leq d' - \theta_i$, into L evenly spaced regions, where region l has boundaries $[b^{(i)}_{l-1}, b^{(i)}_l)$:

$$\begin{aligned} b^{(i)}_{l-1} &= \kappa_i + \frac{l-1}{L}(d' - \theta_i - \kappa_i) \\ b^{(i)}_l &= \kappa_i + \frac{l}{L}(d' - \theta_i - \kappa_i). \end{aligned} \tag{22}$$

This is illustrated in Fig. 6. It is easy to see that by construction, transmission delays γ's in each region l are upper-bounded by boundary $b^{(i)}_l$. If we quantize all the delays in each region l to $b^{(i)}_l$, each region has probability $\int_{b^{(i)}_{l-1}}^{b^{(i)}_l} g^{(i)}_{\Gamma_s}(\gamma)d\gamma$, and we can approximate (21) to

$$\begin{aligned} f^{(i)}_{\text{ARQ}}(d') \approx \sum_{l=1}^{L} \int_{b^{(i)}_{l-1}}^{b^{(i)}_l} g^{(i)}_{\Gamma_s}(\gamma)d\gamma \\ \times \left[(1-\pi_i) + \pi_i f\left(d' - D_F - \theta_i - b^{(i)}_l\right)\right]. \end{aligned} \tag{23}$$

Notice that the quantized (23) is much easier to solve, because the integral no longer includes the recursive call. Now (23) can be solved recursively with dynamic programming (DP). DP means that each time $f(d')$ is called, the solution is stored in the d'th entry of the DP table $F[]$, so that if a repeated recursive call $f(d')$ is made, the answer can simply be looked up instead.

The complexity of solving $f(d')$ is bounded by the time to solve each entry in the DP table, times the number of entries in the table. Solving $f(d')$ involves m channels and $O(L)$ operations in (23) for each channel, and there are a maximum of d' filled entries in the DP table. Hence the complexity of the algorithm is $O(Lmd')$.

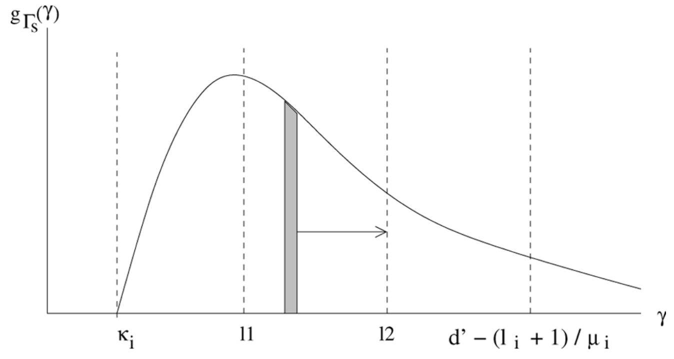


Fig. 6. Illustration of quantization scheme.

VII. Devising FEC Striping Algorithms

We now turn our attention to devising FEC striping algorithms for a set of m bandwidth-limited, burst-loss channels to exploit the interleaving effect of FEC striping. We first derive an FEC-based algorithm in Section VII-A. We then discuss how to appropriately set the Lagrange multiplier value, which controls the volume of parity packets entering the set of queues. Finally, we derive a Hybrid FEC/ARQ algorithm that exploits both the channel selection effect of ARQ striping and interleaving effect of FEC striping at the same time in Section VII-C.

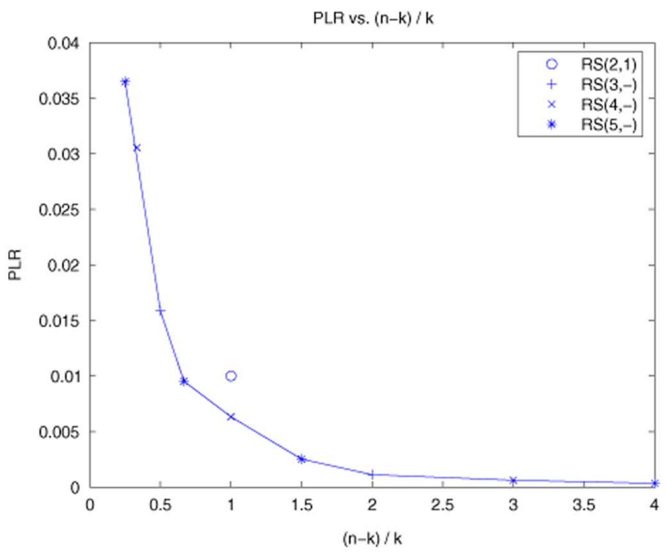
A. FEC-Based Algorithm

We will assume greedy algorithm `local` is always used to find a sub-optimal but good FEC distribution g for a given $\text{RS}(n,k)$ to be deployed on a set of m burst-loss channels. In general, we consider $\text{RS}(n,k)$ while varying n and k for different channel coding strengths and FEC encoding/decoding delays, where n/k is no larger than the ratio of the aggregate channel packet rate to the input packet rate. Let $f_{\text{FEC}}(d'_1), d'_1 = d_1 - t$, be the probability that a packet with expiration d_1 is timely delivered using FEC. To be precise, $f_{\text{FEC}}(d'_1)$ affects all k data packets in $\text{RS}(n,k)$, and so we should maximize the average success probability of all k packets in the head of the incoming packet queue. However, because we assume the packets in the queue are ordered by expiring deadline, we can lower-bound the decoding success probability $f^{\text{g}}_{n,k}(d'_i)$ of k packets with the FEC decoding success probability of the first packet $f^{\text{g}}_{n,k}(d'_1)$. We can now write $f_{\text{FEC}}(d'_1)$ as

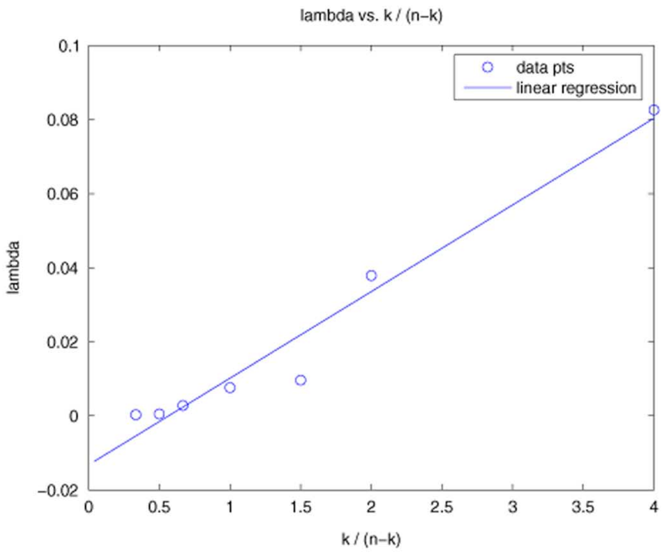
$$\begin{aligned} f_{\text{FEC}}(d'_1) &= \max_{(n,k)} \left[\frac{1}{k}\sum_{i=1}^{k} f^{\text{g}}_{n,k}(d'_i) - \lambda\left(\frac{n-k}{k}\right)\right] \\ &\approx \max_{(n,k)} f^{\text{g}}_{n,k}(d'_1) - \lambda\left(\frac{n-k}{k}\right) \end{aligned} \tag{24}$$

where $f_{\text{FEC}}(d'_1)$ is optimized over a range of n and k.

Notice there is a *penalty* term $\lambda((n-k)/(k))$ in (24). The reason is that using $\text{RS}(n,k)$ invariably increases the traffic volume by $(n-k)/k$ fraction more parity packets. Hence a penalty term is used to regulate the packet volume so that it does not lead to excessive queuing delays; in Section IX-A2, we will demonstrate the importance of the penalty function by comparing the performance of the FEC-based algorithm with



(a)



(b)

Fig. 7. Method of selecting Lagrange multiplier value. (a) Convex hull of FEC. (b) Linear regression of Lagrange multiplier.

and without penalty. The proper selection of the weight of the penalty function—*Lagrange multiplier* λ—is also crucial to the performance of (24); this is the subject of the following section.

$f^{\mathrm{g}}_{n,k}(d'_1)$ in (24) can be approximated as follows: it is the PLR associated with the FEC distribution g of $\mathrm{RS}(n,k)$ over m channels, multiplied by $\Phi^{\mathrm{g}}_{n,k}(d'_1)$—probability that *all* n FEC packets arrive at the receiver in time duration d'_1 given distribution g and queuing delays $\theta'_i s$. It is an approximation because it assumes all n FEC packets must first be transmitted over the delay–varying channel, before packet losses are determined and the FEC block is decoded. Delay of the n packets $\Phi^{\mathrm{g}}_{n,k}(a)$ is defined as follows:

$$\Phi^{\mathrm{g}}_{n,k}(a) = \prod_{i=1}^{m} \prod_{j=1}^{u_i+v_i} \int_{\kappa_i}^{a-\theta_i-(j-1)/\mu_i} g^{(i)}_{\Gamma_s}(\gamma)d\gamma. \tag{25}$$

$f^{\mathrm{g}}_{n,k}(d'_1)$ can now be written as

$$f^{\mathrm{g}}_{n,k}(d'_1) \approx [1-\pi(\mathrm{g})]\Phi^{\mathrm{g}}_{n,k}(d'_1). \tag{26}$$

B. Lagrange Multiplier Selection

At a high level, since the goal of the penalty function $\lambda((n-k)/(k))$ is to regulate the volume of packets in m queues, it makes sense to select λ to be proportional to the total amount of traffic currently in the m queues. So given packet volume w, the question is how to select an appropriate slope d and an y-intercept h in linear equation $\lambda = dw + h$?

Parameters d and h control the sensitivity of the penalty function $\lambda((n-k)/(k))$ to the volume of queue traffic. To derive the appropriate sensitivity, we first trace out each multiplier value λ_i at which optimization (24) switches optimal solutions $\mathrm{RS}(n^o_i, k^o_i)$ to $\mathrm{RS}(n^o_{i+1}, k^o_{i+1})$. As an example, we see in Fig. 7(a) that the performance of each FEC $\mathrm{RS}(n,k), n \le 5$, is plotted on a graph of PLR versus *parity-to-data* ratio $r = (n-k)/k$. As λ varies, the FECs that are optimal solutions to (24) are traced out as the convex hull of the graph. Any two consecutive convex hull points, (PLR_i, r_i) and $(\mathrm{PLR}_{i+1}, r_{i+1})$, will induce a slope $\lambda_i = (\mathrm{PLR}_{i+1} - \mathrm{PLR}_i)/(r_i - r_{i+1})$, which is the value at which (24) will switch from solution (PLR_i, r_i) to $(\mathrm{PLR}_{i+1}, r_{i+1})$. If we now plot these slopes as a function of *data-to-parity* ratio $1/r = k/(n-k)$, as shown in Fig. 7(b), we see an almost linear relationship. The line essentially shows how drastically λ-value must change to effect a corresponding change in data-to-parity ratio given optimization (24) is used. This is the sensitivity we are seeking for. The only task left is to find a line of best fit that describes the relationship between λ and data-to-parity ratio. To that end, we use a well-known linear regression technique in [35], where for a given set of data points $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, the parameters of line of best fit $y = dx + h$ are

$$\begin{aligned} h &= \frac{\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i}{N \sum x_i^2 - \left(\sum x_i\right)^2} \\ d &= \frac{N \sum x_i y_i - \sum x_i \sum y_i}{N \sum x_i^2 - \left(\sum x_i\right)^2} \end{aligned} \tag{27}$$

where each summation is taken from $i = 1$ to N. To summarize, we find the parameters d and h of linear equation $\lambda = dw + h$ as follows.

i) Find performance data points (PLR_i, r_i) of PLR vs. parity-to-data ratio for various candidate FECs, $\mathrm{RS}(n,k)$.
ii) Trace the convex hull of the performance graph.
iii) Using convex hull points $(\lambda_i, 1/r_i)$'s, derive appropriate d and h using (27).

C. Hybrid FEC/ARQ Algorithm

To combine the channel selection effect of ARQ striping and interleaving effect of FEC striping, we can combine the ARQ and FEC algorithms into one hybrid algorithm. $f(d'_1)$ is then simply the larger value of the two possible choices—(re)transmission or FEC:

$$f(d_1') = \begin{cases} \max[f_{\mathrm{ARQ}}(d_1'), f_{\mathrm{FEC}}(d_1')] & \text{if } d_1' \geq 0 \\ 0 & \text{o.w.} \end{cases} . \quad (28)$$

Unlike (26), the FEC decoding success probability given FEC distribution g, $f^{\mathrm{g}}_{n,k}(d_1')$, is now defined recursively to permit retransmission (reFEC) if initial FEC decoding fails:

$$f^{\mathrm{g}}_{n,k}(d_1') = \int_0^{d_1'} [(1-\pi(\mathrm{g})) + \pi(\mathrm{g}) f(d_1' - D_F - \gamma)] \phi^{\mathrm{g}}_{n,k}(\gamma) d\gamma \quad (29)$$

where $\phi^{\mathrm{g}}_{n,k}(\gamma) = (d\Phi^{\mathrm{g}}_{n,k}(\gamma))/(d\gamma)$ is the probability that $\mathrm{RS}(n,k)$ is ready for decoding after exactly γ time duration. As done in Section VI-B for the ARQ-based algorithm, in order to separate the integral from the recursion in (29), we need to first divide the nonzero area under pdf $\phi^{\mathrm{g}}_{n,k}(\gamma)$ into L quantization regions. We first define the largest minimum delay, $D_{\max}$, experienced by any packet in $\mathrm{RS}(n,k)$ given FEC distribution g due to queuing and shifts in Gamma distributions:

$$D_{\max} = \max_{i=1,\ldots,m} \left[\theta_i + \frac{u_i + v_i - 1}{\mu_i} + \kappa_i \right]. \quad (30)$$

It is clear $\phi^{\mathrm{g}}_{n,k}(\gamma) = 0$ for $\gamma < D_{\max}$. The largest amount of time permissible to transmit all packets is of course d_1'. Hence to quantize the area under $\phi^{\mathrm{g}}_{n,k}(\gamma)$ into L regions, each region l with boundaries $[a_{l-1}, a_l)$, we get:

$$[a_{l-1}, a_l) = \left[\frac{l-1}{L}(d_1' - D_{\max}) + D_{\max}, \frac{l}{L}(d_1' - D_{\max}) + D_{\max} \right). \quad (31)$$

To calculate $\int_{a_{l-1}}^{a_l} \phi^{\mathrm{g}}_{n,k}(\gamma) d\gamma$—the probability that $\mathrm{RS}(n,k)$ is ready for decoding in interval $[a_{l-1}, a_l)$—we simply subtract the probability that all n packets arrive by a_{l-1}, $\Phi^{\mathrm{g}}_{n,k}(a_{l-1})$, from the probability that all n packets arrive by a_l, $\Phi^{\mathrm{g}}_{n,k}(a_l)$. We can now write $f^{\mathrm{g}}_{n,k}(d_1')$ as follows:

$$f^{\mathrm{g}}_{n,k}(d_1') \approx \sum_{l=1}^{L} [(1-\pi(\mathrm{g})) + \pi(\mathrm{g}) f(d_1' - D_F - a_l)] \times \left[\Phi^{\mathrm{g}}_{n,k}(a_l) - \Phi^{\mathrm{g}}_{n,k}(a_{l-1}) \right]. \quad (32)$$

VIII. Real-Time Implementation of the Hybrid FEC/ARQ Algorithm for Media Streams

There are two remaining concerns for the Hybrid FEC/ARQ algorithm developed in Section VII-C when implementing it for real-time striping of media stream. First, as mentioned in Section II, media data like a video frame is often segmented into multiple packets, each with the same delivery deadline, and all packets must be delivered on time or none will be useful for the client decoder. This means the striping gateway must have the ability to optimize a group of packets in the head of the queue at the same time. Second, it is clear that the hybrid FEC/ARQ algorithm is computation-intensive, and a fast implementation is needed. In this section we address these two issues in order.

A. Optimizing Packet Group at Head of Input Queue

To optimize delivery of N packets at the heard of input queue, we modify our Hybrid FEC/ARQ algorithm (28) as follows. We pass on an additional argument c to function $f(d_1')$, where c indicates the number of packets we need to optimize. A recursive call $f(d_1', N)$ will return the optimal answer. In more details, when we test simple transmission on channel i for the first of c packets, we additionally call $f(d_1', c-1)$ to calculate PLR for the remainder of the of packets. Mathematically, for ARQ we modify (21) and (23) to (33)-(34), shown at the bottom of the page.

Two details are worthy of note here. First, in (33), we recurse on $f()$ to find the average PLR over c packets only if there are more packets in input queue to consider ($c \geq 2$). Second, regardless of the current value of c, recursion call on $f()$ in (34) has argument c reduced to 1. This prevents the averaging of PLR in lower level of recursive calls in the recursion tree as done in the first line of (33).

FEC-based recursions (24) and (32) need to be modified accordingly as well. The necessary modifications are similar and hence are omitted here.

B. Two-Tier Dynamic Programming Implementation

To reduce the computation complexity of the Hybrid FEC/ARQ algorithm (28), we employ a two-tier dynamic programming implementation. The first tier of dynamic programming, like the dynamic programming tables used for ARQ-based algorithm of (23) in Section VI-B, is used when (28) is solved for the first time. Because (28) recursively calls $f()$ with smaller arguments repeatedly, computed value of $f(a)$ can be stored in the ath entry of dynamic programming (DP) table $F[]$, so that future recursive calls of same argument can

$$f_{\mathrm{ARQ}}(d_1', c) = \begin{cases} \max_{i=1,\ldots,m} \left(\frac{1}{c}\right) f^{(i)}_{\mathrm{ARQ}}(d', c) + \left(\frac{c-1}{c}\right) f(d', c-1) & \text{if } (c \geq 2) \\ \max_{i=1,\ldots,m} f^{(i)}_{\mathrm{ARQ}}(d', c) & \text{o.w.} \end{cases} . \quad (33)$$

$$f^{(i)}_{\mathrm{ARQ}}(d_1', c) = \sum_{l=1}^{L} \int_{b^{(i)}_{l-1}}^{b^{(i)}_l} g^{(i)}_{\Gamma_s}(\gamma) d\gamma \times \left[(1-\pi_i) + \pi_i f\left(d' - D_F - \theta_i - b^{(i)}_l, 1\right) \right]. \quad (34)$$

TABLE II
MODEL PARAMETERS FOR PACKET LOSS EXPERIMENTS

channel	p	q	μ	α	λ	κ
1	0.05	0.45	$30ms/pkt$	4	0.2	50
2	0.03	0.27	$30ms/pkt$	4	0.2	50
3	0.05	0.4	$25ms/pkt$	4	0.16	50

be simply looked up instead of re-computed. Further, we can restrict the size of the DP table to a limit H entries—hence placing an upper bound on the execution time. To do so, we must derive an index a' into the table by first dividing the argument a of $f(a)$ by constant K to place or retrieve a value into or from the table; $K \in \mathcal{R}$ can be selected so that all possible arguments a's map just inside the available space H:

$$K = \frac{a_{\max}}{H-1} \quad a' = \left\lfloor \frac{a}{K} \right\rfloor \tag{35}$$

where $a_{\max}$ is the largest possible argument for (28). Because $f()$ is monotonically non-decreasing by definition, the rounding down operation provides a lower bound when calculating $f()$ recursively using the table.

The second tier of dynamic programming is used when parameters of the network models remain unchanged from packet to packet. Observe that the algorithm is computed based only on the following: i) survival time d' of the first packet in the head of queue; ii) number of packets N in the head of queue; and, iii) queuing delays θ_i's in the outgoing channels. Each time $f(d')$ is computed using (28), the solutions should be stored in entry $[d'][N][[\theta_1][\theta_2][\theta_3]$ of a DP table Soln (assuming the number of channels is 3). When a future packet arrives with survival time d', number of packets in input queue N,and queue delays θ_1, θ_2 and θ_3, the striping engine can have its solution simply looked up in Soln. Similar dividing and rounding operation by constant factor K_2 can be done for queuing delays θ_i's as well to further reduce complexity at the cost of solution quality.

IX. PERFORMANCE EVALUATION

To test the developed striping algorithms, we implemented `muns` (MUlti-path Network Simulator) in C on `linux`, with Gilbert losses, and constant queuing delays and shifted-Gamma distributed transmission delays, as shown in Fig. 5. The network model parameters assumed are shown in Table II. We studied the performance of our algorithms for two kinds of data traffic. In the first set of results we used a constant bit rate data stream. Second set of results is based on two H.264 encoded video streams.

A. Constant Rate Traffic

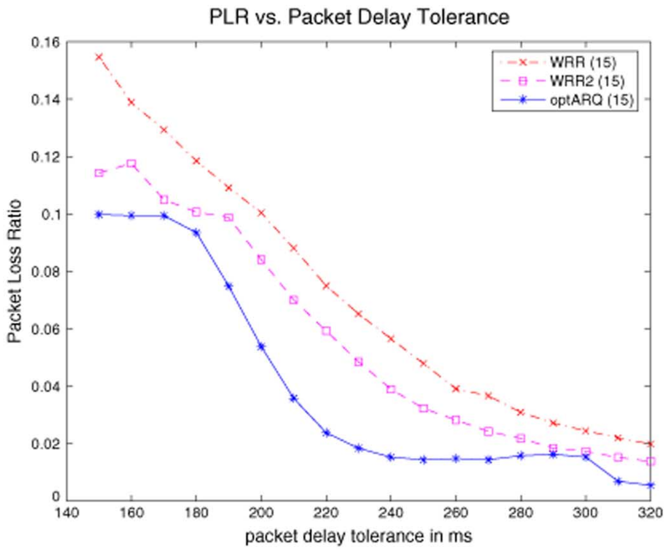
In this section, we use the packet loss ratio as the metric for evaluating the various striping algorithms for constant bitrate data source. For each data point of PLR, 300 000 packets were inputed for an averaging effect.

1) ARQ-Based Algorithm: We first experimentally examine the channel selection effect of striping; we compare the performance our optimal ARQ scheme `optARQ` in (21) with weighted round-robin WRR, which randomly assigns incoming input packets to channels with probabilities proportional to the relative sizes of the channel bandwidths, and biased weighted round robin `WRR2` which is like `WRR`, but only chooses channels where the packet in the head of the queues has a non-zero probability of successful transmission. Fig. 8 shows the resulting PLR of the three schemes as a function of packet end-to-end delay tolerances in ms. Quantization was set to $L = 10$ to solve (23); L was set large enough so that quantization effects are negligible. We conducted two trials, with input packet spacing of 15 ms and 16 ms respectively, resulting in input packet volume of 66.7 pkts/s and 62.5 pkts/s respectively. We see that `optARQ` outperformed `WRR` and `WRR2` for the entire range of packet tolerance delay for both trials. In particular, at the threshold value of delay tolerance of 220 ms where the timely delivery of the second transmission depends heavily on the channel selection, `optARQ` outperformed `WRR` and `WRR2` by 5.1% and 3.5% for trial 1 and 4.1% and 2.8% for trial 2. This demonstrates that the channel selection effect of striping—the clever selection of delivery channels for transmission and retransmission packets—is important and makes significant difference in realistic scenarios.

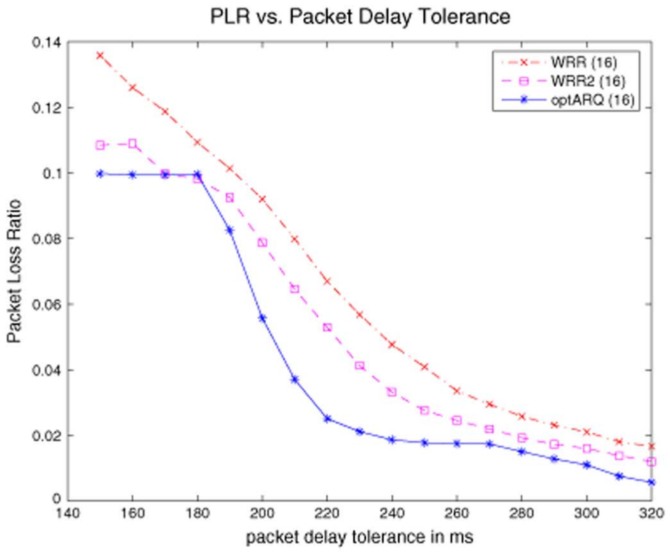
2) FEC-Based Algorithm: We next examine the interleaving effect of striping by investigating the performance of our devised FEC-based algorithm. We limited the feasible FEC set to be the set of $\mathrm{RS}(n,k)$'s, $k < n \leq 5$. First, we compare the performance of our FEC-based algorithm when greedy FEC distribution selection algorithm `local`, discussed in Section V, is used, versus the same algorithm when an exhaustive search algorithm `exhaust` is used to search for good FEC distributions. We conducted two trials for packet spacing of 15 ms and 16 ms. We see in Fig. 9 that `local` performed almost identically to `exhaust` for both packet spacing of 15 ms and 16 ms. This shows that though our `local` selection algorithm may on occasion be sub-optimal, it performs sufficiently well for practical purposes. We note that while the running time of using `exhaust` was about twice the time of using `local`, it was not computationally prohibitive for the small range of $\mathrm{RS}(n,k)$ we searched in the search space.

Using the linear regression method described in Section VII-B to find the appropriate λ for given volume of packets in queues, we traced the performance of our FEC-based algorithm `optFEC` of (24) and plotted in Fig. 10(a) against packet delay tolerance in ms. Input packet spacing was 15 ms. For comparison, we plotted two other FEC schemes. The `uniFEC` finds the currently best performing channel coding $\mathrm{RS}(n,k)$ and transmits the data and parity packets over the *single* channel with the highest delivery success probability given current queue lengths and network conditions. The `fixFEC` performs fixed RS(4, 3) but stripes over three channels using greedy algorithm `local`. Both `uniFEC` and `fixFEC` will elect to send simple packet transmission if simple transmission has better delivery success probability due to delays introduced by FEC.

Several observations can be made in Fig. 10(a). First, performance benefits due to FEC for `fixFEC` kicked in earlier than `uniFEC`. This is because striping across channels typically has the benefit of reducing end-to-end FEC decoding delay. Due

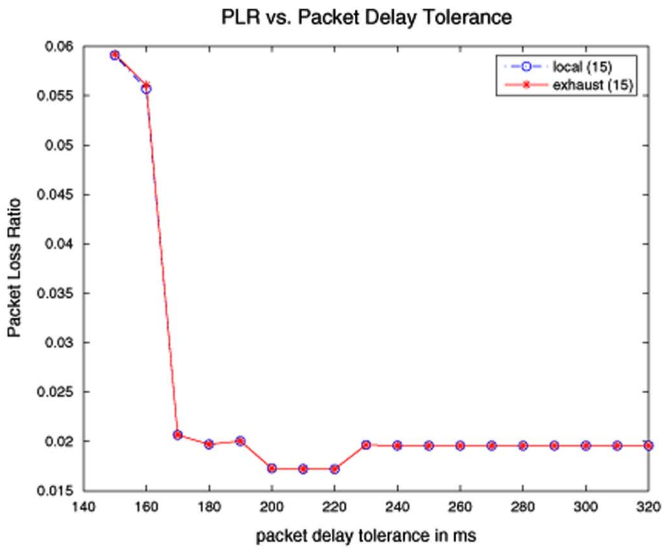


(a)

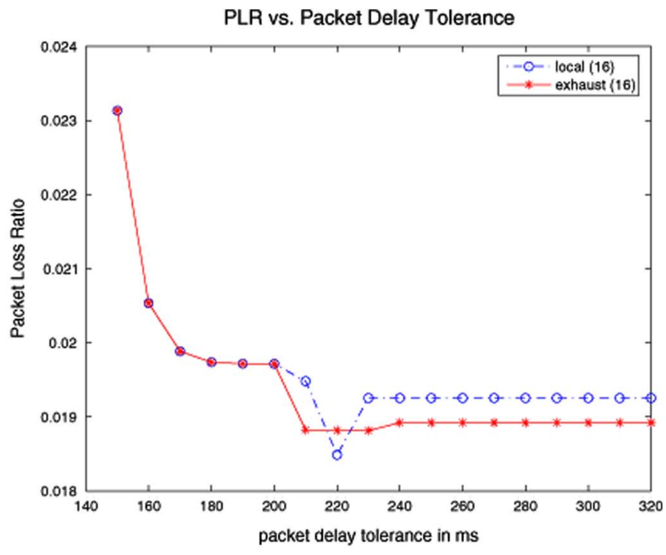


(b)

Fig. 8. Performance comparisons for ARQ schemes. (a) PLR comparison for input packet $\text{spacing} = 15$ ms. (b) PLR comparison for input packet $\text{spacing} = 16$ ms.



(a)



(b)

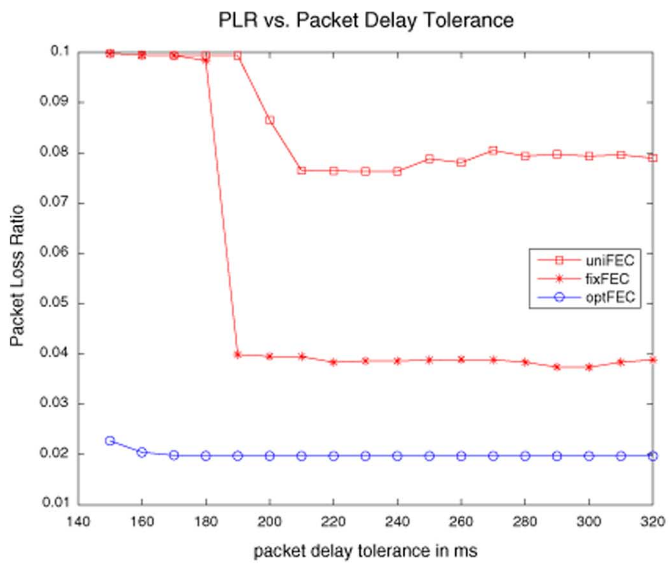
Fig. 9. Comparing greedy and exhaustive FEC distribution selection schemes. (a) PLR comparison for input packet $\text{spacing} = 15$ ms. (b) PLR comparison for input packet $\text{spacing} = 16$ ms.

to this early "kick-in" effect of FEC striping, `optFEC` outperformed `uniFEC` and `fix-FEC` by up to to 8.0% at low delay tolerance.

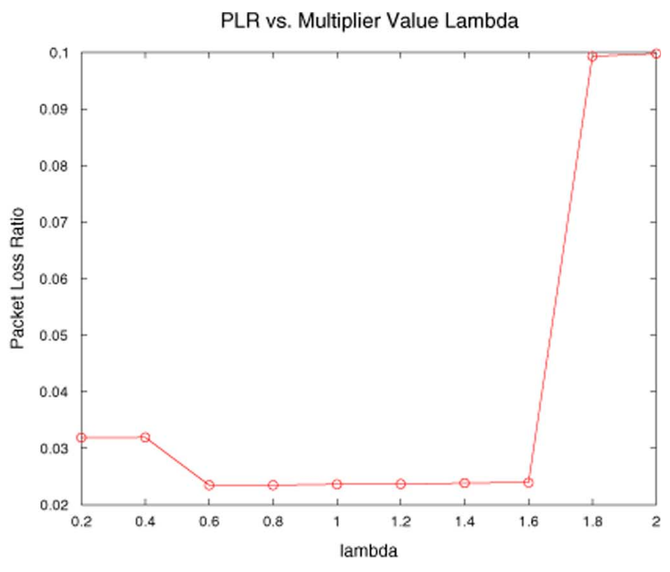
Second, even after FEC benefits of `uniFEC` kicked in, PLR of `fixFEC` was still smaller than `uniFEC`. This is because a single burst in a single channel corrupts entire FEC block for `uniFEC`, while it only corrupts a portion of FEC block for `fixFEC`. `optFEC`, in addition to the interleaving effect, has the flexibility to find the appropriate FEC given the current queuing delays. Due to these advantages, `optFEC` outperformed `uniFEC` and `fix-FEC` by up to to 5.9% and 1.9%. respectively, at high delay tolerance.

3) Hybrid FEC/ARQ Algorithm: We next investigate the performance of the hybrid FEC/ARQ algorithm `Hybrid` in (28) and (29). Recall in Section VII that the performance of both FEC-based and Hybrid FEC/ARQ algorithm depends on the selection of the Lagrange multiplier λ, which determines the weight of the penalty function $\lambda((n-k)/(k))$. To stress this point, we constructed Fig. 10(b), which shows the performance of our hybrid FEC/ARQ algorithm (`Hybrid`) in PLR as a function of λ, where for each data point λ was held constant for the experimental run. Input packet spacing is 16 ms, and end-to-end packet delay tolerance is 150 ms. We see that an inappropriate λ value results in a worse PLR by 7.6%, demonstrating the importance of a cleverly selected λ.

To validate the performance of `Hybrid`, we compare the following. For input packet spacing of 15 ms and 16 ms, performance in PLR is again plotted against packet delay tolerance.

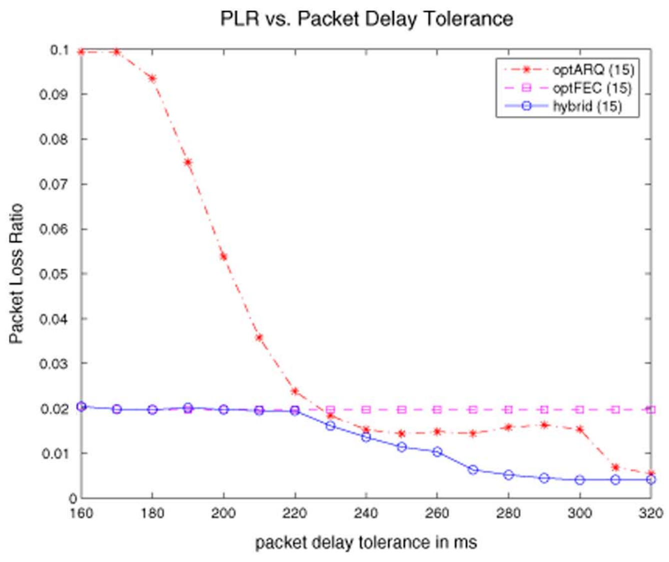


(a)

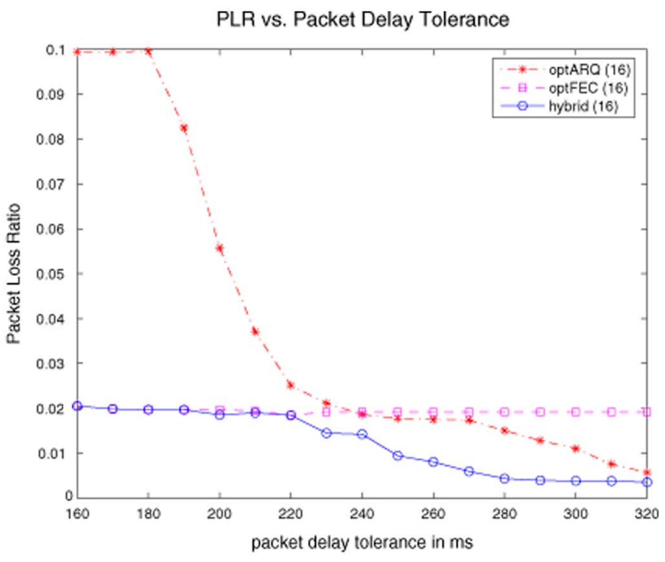


(b)

Fig. 10. Performance comparison for FEC schemes and for multiplier variation. (a) PLR comparison for input packet $\texttt{spacing} = 15$ ms. (b) PLR as function of multiplier value λ.



(a)



(b)

Fig. 11. Comparing `optFEC`, `optARQ` and `Hybrid` schemes.

For comparison, the performance of `optFEC` and `optARQ` are also plotted in Fig. 11. We see that the performance of `hybrid` was at least as good as both `optFEC` and `optARQ` for all range of packet delay tolerance, and at some high delay tolerance, `hybrid` outperformed both `optFEC` and `optARQ`.

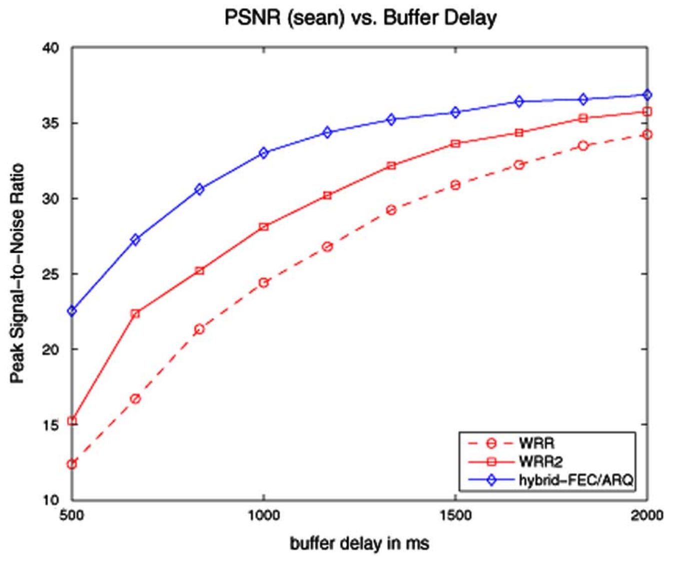
B. *H.264 Video Streaming*

Finally, we show the applicability and performance of our developed packet striping system to streaming video applications. Using a H.264 video encoder, we encoded two 100-frame QCIF (176×144) MPEG test video sequences named `sean` and `foreman` of 10 frames per second at 28 kbps and 64 kbps in IPPP format (one I-frame followed by P-frames). We use a streaming server to send frames to a multi-homed wireless streaming client simply according to their presentation times, via our striping system. Each compressed video frame is broken into one or more packets of no more than 1500 bytes, which is assumed to be the Maximum Transport Unit (MTU). Parameters of the network model for this part of the experiment is shown in Table III.

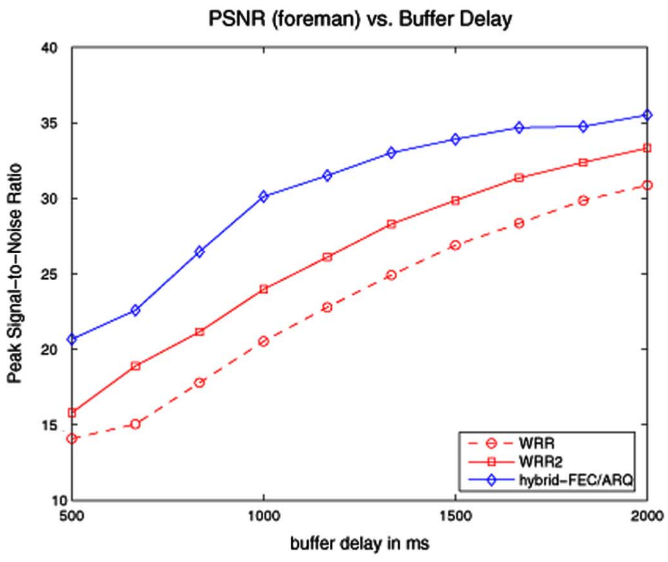
A frame at the client is decoded on time if: a) it is delivered by its playback deadline; and, b) its reference frame was decoded on time. If a frame i is timely decoded, PSNR is calculated using the reconstructed frame i and the original frame i. If a frame i cannot be decoded on time, the most recently timely decoded frame j is used as its replacement, and PSNR is calculated using reconstructed frame j and original frame i.

TABLE III
MODEL PARAMETERS FOR VIDEO STREAMING EXPERIMENTS

channel	p	q	μ (sean)	μ (foreman)	α	λ	κ
1	0.05	0.45	$208ms/pkt$	$202ms/pkt$	3	0.1	50
2	0.03	0.27	$208ms/pkt$	$202ms/pkt$	3	0.1	50
3	0.05	0.4	$194ms/pkt$	$191ms/pkt$	3	0.16	50



(a)



(b)

Fig. 12. Performance comparison among WRR, WRR2 and hybrid-FEC/ARQ schemes: PSNR vs. client buffer delay. (a) PSNR comparison for `Sean`. (b) PSNR comparison for `Foreman`.

We compare the performance of our hybrid FEC/ARQ scheme (`hybrid-FEC/ARQ`) to two competing schemes: weighted round robin (`WRR`), and biased weighted round robin (`WRR2`). `WRR` and `WRR2` are as described in Section IX-A1. In Fig. 12, we see the performance of the three schemes in PSNR as function of the initial playback buffer delay at the client. We see that for small playback buffer delay, `hybrid-FEC/ARQ` outperformed `WRR` by up to 10.5 dB and `WRR2` by up to 7.3 dB for the `sean` sequence, and outperformed `WRR` by up to 9.6 dB, `WRR2` by up to 6.1 dB for the `foreman` sequence. This shows that though all three schemes enjoy the benefit of aggregated bandwidth of three bandwidth-limited channels, a good striping algorithm—one that benefits from both the interleaving effect of striping FEC and the channel selection effect of striping ARQ—can intelligently stripe packets across channels to further improve performance drastically for streaming video, particularly for low-delay applications.

X. CONCLUSION

Use of striping or inverse-multiplexing for sharing and aggregation of the limited bandwidth of WWAN connections in a collaborative community of multi-homed wireless devices, each having both a WWAN interface to connect to the Internet and a WLAN interface to connect to its neighbors, has potential to provide ubiquitous highspeed Internet access. Striping traffic over bundled WWAN connections enables streaming of high quality media to devices without highspeed Internet access. Furthermore, smart striping FEC and ARQ packets across multiple channels can improve the timely delivery of delay-sensitive traffic due to following two effects: i) *interleaving effect*, where by striping FEC packets across channels one can avoid FEC decoding failure due to a single burst loss, at the same time avoid the long interleaving delay of a single-channel interleaver; and, ii) *channel selection effect*, where one can judiciously select one among many available channels that maximizes a packet's survival chances given its delivery deadline and the channels' delay and loss characteristics. We have developed dynamic programming based algorithm for smart striping of streaming media along with error correction over multiple burst-loss channels. Our simulation-based performance evalution shows that our striping algorithm finds an operating region to balance conflicting channel characteristics such as loss, latency and bandwidth to outperform naïve algorithms such as weighted round-robin. We have also presented techniques to aid the real-time implementation of the proposed striping algorithm. Since our striping scheme operates on a per-packet basis and not per-flow, we can easily extend our developed techniques to multiple flows sharing multiple channels. In future we plan to study the performance impact of variations in the channel properties due to fading and interference.

REFERENCES

[1] H. Holma and A. Toskala, *HSDPA / HSUPA for UMTS*. Hoboken, NJ: Wiley, 2006.

[2] P. Sharma, S.-J. Lee, J. Brassil, and K. Shin, "Distributed communication paradigm for wireless community networks," in *IEEE Int. Conf. Communications*, Seoul, Korea, May 2005.

[3] S. Dogan *et al.*, "Error-resilient video transcoding for robust internetwork communications using GPRS," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 453–464, Jun. 2002.

[4] A. C. Snoeren, "Adaptive inverse multiplexing for wide area wireless networks," in *Proc. IEEE GLOBECOM'99*, 1999.

[5] H.-Y. Hsieh and R. Sivakumar, "A transport layer approach for achieving aggregate bandwidth on mutli-homed mobile hosts,," *Wireless Netw.*, vol. 11, no. 1, pp. 99–114, Jan. 2005.

[6] K. Chebrolu and R. Rao, "Communication using multiple wireless interfaces," in *Proc. IEEE WCNC*, Orlando, FL, Mar. 2002.

[7] K. Chebrolu and R. Rao, "MAR: A commuter router infrastructure for the mobile internet," in *Proc. IEEE WCNC*, Orlando, FL, Mar. 2002.

[8] G. T. Nguyen, R. H. Katz, B. Noble, and M. Satyanarayanan, "A trace-based approach for modeling wireless channel behavior," in *Proc. WSC'96*, Coronado, CA, Dec. 1996, pp. 597–604.

[9] J. Padhye, V. Firoiu, D. Towsley, and J. KurosE, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM'98)*, Vancouver, Canada, Sept. 1998, pp. 303–314.

[10] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, no. 3, pp. 336–350, Jul. 1997.

[11] A. Abouzeid, S. Roy, and M. Azizoglu, "Stochastic modeling of TCP over lossy links," in *Proc. IEEE Conf. Computer Communications (INFOCOM)*, Tel Aviv, Israel, Mar. 2000, pp. 1724–1733.

[12] H. M. Chaskar, T. V. Lakshman, and U. Madhow, "TCP over wireless with link level error control: Analysis and design methodology," *IEEE/ACM Trans. Netw.*, vol. 7, no. 5, pp. 605–615, Oct. 1999.

[13] R. Mukhtar, M. Zukerman, and F. Cameron, "Packet latency for type-II hybrid ARQ transmissions over a correlated error channel," in *Eur. Wireless Conf.*, Florence, Italy, Feb. 2002.

[14] L. Pu, M. W. Marcellin, I. Djordjevic, B. Vasic, and A. Bilgin, "Joint source-channel rate allocation in parallel channels," *Proc. SPIE*, vol. 6508, p. 65081A, 2007.

[15] S. Johansen and A. Perkis*, Unequal Error Protection for Embedded Codes Over Parallel Packet Erasure Channels*, 2005.

[16] P. Frossard and O. Verscheure, "Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery," *IEEE Trans. Image Process.*, vol. 10, no. 12, pp. 1815–1825, Dec. 2001.

[17] P. Liu, R. Berry, and M. Honig, "Delay-sensitive packet scheduling in wireless networks," in *Proc. IEEE WCNC 2003*, New Orleans, LA, Mar. 2003.

[18] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel," *Wireless Netw.*, vol. 8, no. 1, pp. 13–26, Jan. 2002.

[19] S. H. Kang and A. Zakhor, "Packet scheduling algorithm for wireless video streaming," in *Proc. Packet Video 2002*, Pittsburgh, PA, Apr. 2002.

[20] R. S. Tupelly, J. Zhang, and E. K. P. Cheng, "Opportunistic scheduling for streaming video in wireless networks," in *Proc. 2003 Conf. Information Science and Systems*, Baltimore, MD, Mar. 2003.

[21] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

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

[23] M. Zorzi, "Performance of FEC and ARQ error control in bursty channels under delay constraints.," in *Proc. IEEE VTC'98*, Ottawa, ON, Canada, May 1998.

[24] S. Mao, S. Panwar, and Y. Hou, "On optimal traffic partitioning for multipath transport," in *IEEE INFOCOM 2005*, Miami, FL, Mar. 2005.

[25] G. Cheung, P. Sharma, and S. J. Lee, "Striping delay-sensitive packets over multiple bursty wireless channels," in *IEEE Int. Conf. Multimedia and Expo*, Amsterdam, The Netherlands, July 2005.

[26] G. Cheung, P. Sharma, and S. J. Lee, "Striping delay-sensitive packets over multiple burst-loss channels with random delays," in *IEEE Int. Symp. Multimedia*, Irvine, CA, Dec. 2005.

[27] G. Cheung, P. Sharma, and S. J. Lee, "Implementation and evoluation of packet striping for media streaming over multiple burst-loss channels," in *IEEE Int. Conf. Multimedia and Expo*, Toronto, ON, Canada, July 2006.

[28] E. O. Elliott, "A model of the switched telephone network for data communications," *Bell Syst. Tech. J.*, vol. 44, pp. 89–109, Jan. 1965.

[29] P. A. Chou, S. Mehrotra, and A. Wang, "Multiple description decoding of overcomplete expansions using projections onto convex sets," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Mar. 1999, pp. 72–81.

[30] S. B. Wicker and V. K. Bhargava, Eds.*, Reed-Solomon Codes and Their Applications*. New York: Wiley, 1999.

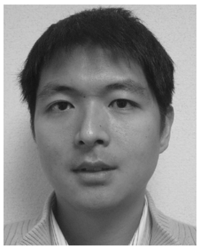
[31] F. Zhai, Y. Eisenberg, T. N. Pappas, R. Berry, and A. K. Katsaggelos, "Rate-distortion optimized product code forward error correction for video transmission over IP-based wireless networks," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP 2004)*, Montreal, QC, Canada, May 2004.

[32] U. Sorger, "A new reed-solomon code decoding algorithm," *IEEE Trans. Inform. Theory*, vol. 39, no. 2, Mar. 1993.

[33] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Image Commun.*, vol. 15, no. 1–2, Sept. 1999.

[34] A. Leon-Garcia*, Probability and Random Processes for Electrical Engineering*. Reading, MA: Addison Wesley, 1994.

[35] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, Eds.*, Numerical Recipes in C++*. Cambridge, U.K.: Cambridge Univ. Press, 2002.



Gene Cheung (M'00–SM'07) received the B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1998 and 2000, respectively.

In August 2000, he joined Hewlett-Packard Laboratories Japan, Tokyo, where he is currently a Senior Researcher in the multimedia systems architecture group. His research interests include multimedia processing and networking, wireless networks and combinatorial optimization.

Puneet Sharma (M'97) received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Delhi, and the Ph.D. degree in computer science from the University of Southern California, Los Angeles, in 1998.

Currently, he is a Senior Research Scientist at Hewlett-Packard Laboratories, Palo Alto, CA, where he conducts research in wireless and mobile networking, overlay network services, network measurement and monitioring.

Sung-Ju Lee (M'00–SM'06) received the Ph.D. degree in computer science from the University of California, Los Angeles.

He is a Research Scientist at the Mobile & Media Systems Lab (MMSL) of HP Labs, Palo Alto, CA. He has published nearly 60 papers in the field of computer networks. His research interests include computer networks, mobile networking and computing, wireless mesh and ad hoc networks, wireless LANs, overlay networks, media over networks, and large-scale service infrastructure networks.

Dr. Lee currently serves on the editorial board of Elsevier Science's *Ad Hoc Networks Journal* and also serves as a technical program committee member of various prestigious networking related conferences. He is on a steering committee of IEEE SECON and ACM WMASH. He is a member of ACM.