Striping Delay-sensitive Packets over Multiple Burst-loss Channels with Random Delays

Gene Cheung, Puneet Sharma and Sung-Ju Lee
Mobile & Media Systems Lab, Hewlett-Packard Laboratories

Abstract

Multi-homed mobile devices have multiple wireless communication interfaces, each connecting to the Internet via a long range but low speed and bursty WAN link such as a cellular link. We propose a packet striping system for such multi-homed devices — a mapping of delay-sensitive packets by an intermediate gateway to multiple channels, such that the overall performance is enhanced. In particular, we model and analyze the striping of delay-sensitive packets over multiple burst-loss channels with random delays. We first derive the expected packet loss ratio when forward error correction (FEC) is applied for error protection over multiple channels. We next model and analyze the case when the channels are bandwidth-limited with shifted-Gamma-distributed transmission delays. We develop a dynamic programming-based algorithm that solves the optimal striping problem for the ARQ, the FEC, and the hybrid FEC/ARQ case.

1 Introduction

Many modern wireless devices are multi-homed — having multiple wireless communication interfaces, each connecting to the Internet via a wireless wide area network (WWAN) interface such as a cellular link. Though this type of interface provides long range services, the bandwidth is limited, and packet losses are frequent and bursty. To enhance performance in this setting, an assistant gateway can "aggregate" device's low speed WAN channels — a mapping of incoming packets to its multiple channels together with the use of error protection schemes such as forward error correction (FEC) and retransmissions (ARQ) — to optimize end-to-end packet delivery. Clearly, such *striping* engine can potentially improve delivery of delay-sensitive media streaming data greatly: like a typical single channel packet interleaver, by spreading FEC packets across channels, one avoids decoding failure due to a single burst loss, yet unlike the interleaver, one also avoids excessive transmission delay of long interleaving.

Indeed, this striping or *inverse multiplexing* problem has recently received great interest in mobile wireless networking domain [1, 2]. Yet previous work has mainly focused on designing wireless inverse multiplexing systems to improve TCP throughput in such environments. Unlike previous work that focuses on bulk transfer, we focus our attention on delay-sensitive packet delivery such as media streaming. The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 provides basic definitions and the modeling of bursty error channels. Section 4 derives the effective packet loss ratio when FEC is applied to a single bursty channel. Section 5 derives the effective loss ratio when Reed-Solomon (n, k) code is striped over a set of m bursty channels under a particular mapping. Striping on bandwidth limited, bursty channels is analyzed and optimization algorithms are designed in Section 6 for the ARQ-based algorithm and Section 7 for the FEC-based and hybrid FEC/ARQ algorithms. Finally, results and conclusion are presented in Section 8 and Section 9, respectively.

This work is a generalization of our previous work [3] on striping delay-sensitive packets over multiple burst-loss wireless channels. In [3], the transmission delay of each channel is modeled as a constant. In contrast, the transmission delay in this work has been generalized to a shifted-Gamma-distributed random variable, which is found to be accurate for common network load [4]. Given the generalization of network model, we detail the corresponding striping optimization based on earlier developed algorithms [3].

2 Background

As shown in Figure 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 onto the channels for optimized performance (i.e., high throughput and bounded delay) is also critical. The receiving end of the striping system must re-synchronize out-of-order delivery packets. In this paper we assume the existence of reassembly mechanisms that handle reordering of packets. Applications such as media streaming use receiver buffers that can also be used for packet reordering.

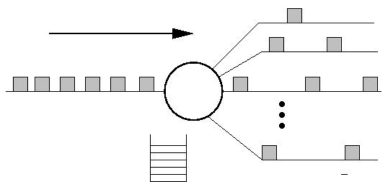


Figure 1. Packet Striping Engine

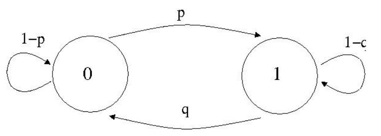


Figure 2. Gilbert loss model

The inverse multiplexing network model we consider is composed of wireless channels with bursty losses. We also apply forward error correction (FEC) technique and analyze the striping performance in bandwidth limited channels. We are particularly interested in streaming applications that are delay-sensitive. There is a significant amount related work, although only a few of these consider striping in wireless channels. We overview the earlier research in this area.

Modeling the wireless channel behavior has been an active research area. Wireless channel is modeled using the traces in [5]. 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.

Streaming over lossy channels creates another challenge as packets are delay-sensitive. Streaming packet scheduling over wireless channels has been investigated in [6, 7]. Rate-distortion optimized packet scheduling is thoroughly analyzed in [8], and scheduling of layered streaming video is presented in [9]. FEC and ARQ performances in continuous streams over bursty channels are compared in [10].

In our model, we assume the packet size and the transmission rate are constant. The wireless channels are always available, although they will sometimes have errors. In other words, the disappearance of the channels due to mobility of the end hosts is not considered.

3 Channel Model Basics

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

We begin with definitions similar to those introduced in [11]. 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. $p(i) = \Pr(0^i 1|1)$. 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:

$$p(i) = \begin{cases} 1-q & \text{if } i=0 \\ q(1-p)^{i-1}p & \text{o.w.} \end{cases} \quad (1)$$

$$P(i) = \begin{cases} 1 & \text{if } i=0 \\ q(1-p)^{i-1} & \text{o.w.} \end{cases} \quad (2)$$

$$q(i) = \begin{cases} 1-p & \text{if } i=0 \\ p(1-q)^{i-1}q & \text{o.w.} \end{cases} \quad (3)$$

$$Q(i) = \begin{cases} 1 & \text{if } i=0 \\ p(1-q)^{i-1} & \text{o.w.} \end{cases} \quad (4)$$

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

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

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:

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

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.

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

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.

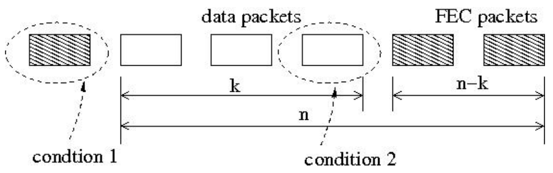


Figure 3. FEC encoding of data packets.

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

$s(m,n)$ and $\bar{s}(m,n)$ are defined counterparts to $r(m,n)$ and $\bar{r}(m,n)$.

4 FEC for One Burst-loss Channel

We derive the expected packet loss ratio (PLR) of FEC code — α_{RS} of (n,k) Reed-Solomon code in particular — on a burst-loss channel. Reed-Solomon code is commonly used in practice for FEC packet-level recovery systems with delay constraints [12, 13, 14]. Figure 3 shows an example of a RS $(5,3)$ code.

Recall 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), giving us two conditional probabilities, $\alpha_{RS|1}$ and $\alpha_{RS|0}$, respectively. α_{RS} can then be expressed as:

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

where $\pi = \frac{p}{p+q}$ is the raw PLR of the channel.

To find $\alpha_{RS|1}$, we consider the k data packet block and the $n-k$ parity packet block separately. We condition on the status of the last (k-th) data packet; given the k-th data packet is lost or received, we use $R(.,.)$ or $S(.,.)$ for probability calculation of the trailing $n-k$ parity packet block.

Conditioning on the event when the k-th data packet is lost, 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 $\frac{i+1}{k}$, assuming there are $\geq n-k+1$ total loss packets including the $n-k$ parity packets. Similar analysis conditioning on the event when the k-th data packet is successfully received completes the derivation for $\alpha_{RS|1}$:

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

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

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

5 Striping FEC for Multiple Burst-loss Channels

Data and parity packets of a given RS(n,k) can be striped over a set of m channels in multiple ways. We call the mapping of k data and $n-k$ parity packets to m bursty channels an *FEC distribution*. We denote such mapping function as 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. In addition, we define $w_i = u_i + v_i$ as the total number of packets assigned to channel i.

Let random variable X be the number of unrecoverable data packets at the receiver in k data packets in a RS(n,k) code. Let Y, Z and Θ be the number of correctly transmitted data packets, parity packets and total packets, respectively. X, Y and Z are related as follows:

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

When given probability mass functions (pmfs) of Y, Z and $\Theta = Y + Z$, we can find the expectation of X as follows:

$$\begin{aligned} E[X] &= E[k-Y|Y+Z \leq k-1] P(Y+Z \leq k-1) \\ &= (k - E[Y|\Theta \leq k-1]) P(\Theta \leq k-1) \end{aligned} \tag{13}$$

To find $P(\Theta \leq k-1)$, we first define random variables $Y_i \leq u_i$, $Z_i \leq v_i$ and $\Theta_i \leq w_i$ as the number of correctly transmitted data packets, parity packets and total packets in channel i, respectively. We can then write:

$$Y = \sum_{i=1}^{m} Y_i, \quad Z = \sum_{i=1}^{m} Z_i, \quad \Theta = \sum_{i=1}^{m} \Theta_i \tag{14}$$

For each channel i, pmf of $\Theta_i = Y_i + Z_i$ can be written as:

$$P(\Theta_i = j) = \pi_i\, R(w_i - j, w_i) + (1-\pi_i) S(j, w_i) \tag{15}$$

Table 1. Average PLR for FEC distribution search algorithms

Algorithm	greedy1	greedy2	greedy3	greedy4	even	optimal
Avg PLR	0.0128	0.0127	0.0130	0.0129	0.0172	0.0124

where $j = 0, \dots, w_i$. Since Θ, as well as Y and Z, are all sums of random variables, we derive pmf of Θ using probability generating function (pgf) $G_\Theta(\xi)$:

$$\begin{aligned} G_\Theta(\xi) &= E[\xi^\Theta] = \sum_j P(\Theta = j)\xi^j = E[\xi^{\Theta_1 + \dots + \Theta_m}] \\ &= E[\xi^{\Theta_1}] \cdots E[\xi^{\Theta_m}] = G_{\Theta_1}(\xi) \cdots G_{\Theta_m}(\xi) \end{aligned}$$

Hence pgf $G_\Theta(\xi)$ is simply a product of pgfs $G_{\Theta_i}(\xi)$'s. We recover pmf of Θ from pgf $G_\Theta(\xi)$ as follows (p.148 of [15]):

$$P(\Theta = j) = \frac{1}{j!}\frac{d^j}{d\xi^j} G_\Theta(\xi)\bigg|_{\xi=0} \tag{16}$$

We can now find $P(\Theta \le k-1)$ by summing $P(\Theta = j)$ for $0 \le j \le k-1$.

To find $E[Y|\Theta \le k-1]$, we make the simplifying assumption that Y and Z are independent. We get:

$$\begin{aligned} E[Y|Y + Z \le k-1] &\approx E[Y|Y \le k-1] \\ &= \sum_{j=1}^{k-1} \frac{jP(Y=j)}{P(Y \le k-1)} \end{aligned} \tag{17}$$

pmf of Y is found similar to Θ. We will denote $\pi(\text{g})$ as $E[X]/k$ — PLR given mapping g for RS (n, k) code.

5.1 Fast FEC Distribution Search Algorithms

The number of unique mappings of $n-k$ parity and k data packets to m channels is exponential in m and k. Instead of exhaustive search, we explore practical greedy schemes to select good FEC distributions. A greedy algorithm incrementally grows an FEC distribution one packet at a time. The order in which one grows the FEC distribution — when to insert a data packet or a parity packet — greatly affects the performance. We tried several greedy algorithms and present the four best performers.

The first 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. We also compare them with an even allocation scheme `even` where the same number of data and parity packets are evenly allocated to each channel, $\lfloor \frac{k}{m} \rfloor$ and $\lfloor \frac{n-k}{m} \rfloor$, with leftover packets, $k - m\lfloor \frac{k}{m} \rfloor$ and $(n-k) - m\lfloor \frac{n-k}{m} \rfloor$, allocated to the channel with smallest PLR. For three burst-loss channels of parameters $(0.05, 0.45)$, $(0.03, 0.27)$, $(0.05, 0.4)$, we calculated PLR for these algorithms for RS$(7, x)$ and RS$(8, x)$ where $1 \le x \le n-1$. The resulting average effective PLRs over the possible FEC's are shown in Table 1. We compare their performance with the optimal FEC distribution, found by exhaustive search `optimal`.

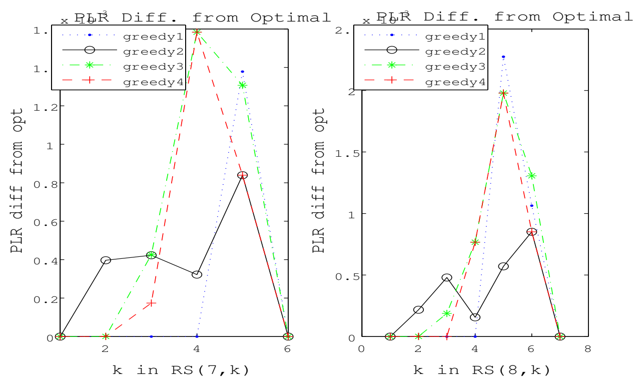


Figure 4. PLR for different FEC distribution search algorithms

`even` is by far the worst performer and `greedy2` is the best overall performer. In fact, when we plot the difference in effective PLR compared with `optimal` in Figure 4, we see that although `greedy2` may not always be the best performer in the group, it has the overall smallest maximum difference. For the above reasons, we use `greedy2` as our heuristics for constructing FEC distribution.

6 Delay-sensitive Traffic over Bandwidth-limited Channels

We expand the Gilbert loss model in Figure 2 to a bandwidth-limited, burst-loss model with random delays as shown in Figure 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. At a given time, the fullness of the queue j is l_j. The time required to transmit a packet through queue j is then: $(l_j + 1)/\mu_j + \gamma_j$. In more details, a

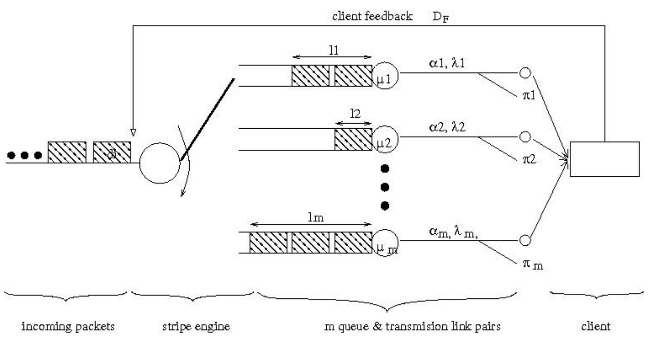


Figure 5. Bandwidth-limited Network Model

Gamma random variable γ with Gamma shape parameter α and scale parameter λ has the following probability density function (pdf) (pg.117 of [15]):

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

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

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

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

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

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

For input into the striping engine, we assume the packets in the incoming queue before the striping engine 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. We assume the packets are ordered in the incoming queue by earliest expiration times. We assume striping engine is activated whenever there is a packet in the incoming queue.

6.1 ARQ-based Algorithm

To achieve low complexity, we choose to optimize one packet at a time with expiration time d. Let $f(d'), d' = d - t$, be the probability that a packet with expiration d is timely delivered to the client, where t is the time of optimization instant at the striping engine. Let $f_{ARQ}(d')$ be the probability that the same packet is timely delivered using (re)transmission (ARQ). Let $f_{ARQ}^{(i)}(d')$ be the probability that the same packet is timely delivered if channel i is first used for ARQ. Given the client can errorlessly inform the engine of the loss event in time D_F, the packet has a chance for retransmission with a tighter deadline. We can write:

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

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 $\gamma > d' - \left(\frac{l_i+1}{\mu_i}\right)$.

6.2 Quantization & Dynamic Programming

As (21) is defined recursively within an integral, it is difficult to solve directly. Instead, our approach is to 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 $g_{\Gamma_s}(\gamma), \gamma \leq d' - \left(\frac{l_i+1}{\mu_i}\right)$, into L evenly spaced regions, where region l has boundaries $[b_{l-1}^{(i)}, b_l^{(i)})$:

$$\begin{aligned} b_{l-1}^{(i)} &= \kappa_i + \frac{l-1}{L}\left(d' - \left(\frac{l_i+1}{\mu_i}\right) - \kappa_i\right) \\ b_l^{(i)} &= \kappa_i + \frac{l}{L}\left(d' - \left(\frac{l_i+1}{\mu_i}\right) - \kappa_i\right) \end{aligned} \qquad (22)$$

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

$$f_{ARQ}^{(i)}(d') \approx \sum_{l=1}^{L} \int_{b_{l-1}^{(i)}}^{b_l^{(i)}} g_{\Gamma_s}^{(i)}(\gamma)d\gamma \left[(1-\pi_i) + \pi_i f\left(d' - D_F - b_l^{(i)}\right)\right] \qquad (23)$$

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 $O(Lmd')$.

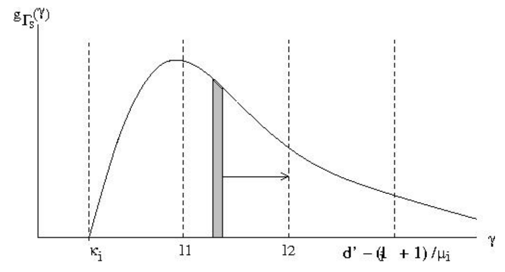
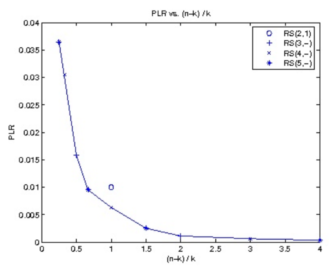
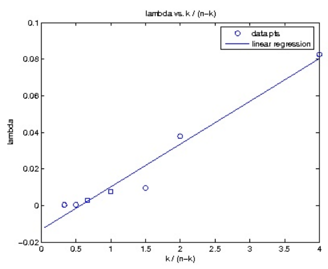


Figure 6. Illustration of Quantization Scheme.



a) Convex Hull of FEC



b) Linear Regression of Lagrange Multiplier

Figure 7. Method of Selecting Lagrange Multiplier Value

7 Deriving FEC Striping Algorithms

In this section, we derive FEC striping algorithms for a set of m bandlimited, burst-loss channels with random delays. We first derive an FEC-based algorithm in Section 7.1. 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 Hybrid FEC/ARQ algorithm in Section 7.3.

7.1 FEC-based Algorithm

We will assume greedy algorithm `greedy2` is always used to find a sub-optimal but good FEC distribution g for a given RS (n, k) to be deployed on a set of m burst-loss channels. In general, we consider RS (n, k) while varying n and k for different channel coding strengths and FEC encoding/decoding delays. Let $f_{FEC}(d'_1), d'_1 = d_1 - t$, be the probability that a packet with expiration d_1 is timely delivered using FEC. To be precise, $f_{FEC}(d'_1)$ affects all k data packets in 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 $f^{\mathrm{g}}_{n,k}(d'_i)$ of each of k packets with expiration time d_i with the FEC decoding success probability of the first packet $f^{\mathrm{g}}_{n,k}(d'_1)$. We can now write $f_{FEC}(d'_1)$ as:

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

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

Notice there is a *penalty* term $\lambda(\frac{n-k}{k})$ in (26). The reason is that using 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 queue overflows. The proper selection of λ is 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 RS (n, k) over m channels, multiplied by the probability $\Phi^{\mathrm{g}}_{n,k}(d'_1)$ that each of the n FEC packets arrives at the receiver in time duration d'_1. $\Phi^{\mathrm{g}}_{n,k}(a)$ is defined as follows:

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

$f^{(i)}_{n,k}(d'_1)$ can now be written as:

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

7.2 Lagrange Multiplier Selection

At a high level, since the goal of the penalty function $\lambda\left(\frac{n-k}{k}\right)$ 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 $\lambda\left(\frac{n-k}{k}\right)$ 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 RS (n^o_i, k^o_i) to RS (n^o_{i+1}, k^o_{i+1}). As as example, we see in Figure 7a that the performance of each FEC RS (n, k), $n \leq 5$, is plotted on a graph of PLR vs. *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 (PLR_{i+1}, r_{i+1}), will induce a slope $\lambda_i = (PLR_{i+1} - PLR_i)/(r_i - r_{i+1})$, which is the value at which (24) will switch from solution (PLR_i, r_i) to (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 Figure 7b, 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 [16], 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:

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

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:

1. Find performance data points (PLR_i, r_i) of PLR vs. parity-to-data ratio for various candidate FECs, RS (n, k).

2. Trace the convex hull of the performance graph.

3. Using convex hull points $(\lambda_i, 1/r_i)$'s, derive appropriate d and h using (27).

7.3 Hybrid FEC/ARQ Algorithm

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:

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

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:

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

where $\phi^{\mathrm{g}}_{n,k}(\gamma) = \frac{d\, \Phi^{\mathrm{g}}_{n,k}(\gamma)}{d\gamma}$ is the probability that RS (n, k) is ready for decoding after exactly γ time duration. As done in Section 6.2 for the ARQ-based algorithm, in order to separate the integral from the recursion in (29), we need to first divide the non-zero area under pdf $\phi^{\mathrm{g}}_{n,k}(\gamma)$ into L quantization regions. We first define the largest queuing delay,

Table 2. Model Parameters for Experiment

chnl	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

$D_{\max}$, experienced by any packet in RS (n, k) given FEC distribution g:

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

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 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:

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

8 Results

To test the developed striping algorithms, we implemented a network simulator in `C` running on `linux`. We assumed input packet rate of $62.5pkt/s$ and network model parameters as shown in Table 2. For each data point of PLR, $300,000$ packets were inputed for an averaging effect.

8.1 ARQ-based Algorithm

We first compare the performance our optimal ARQ scheme `opt-ARQ` 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. Figure 8 shows the resulting PLR of the two schemes as a function of packet end-to-end delay tolerances in ms. In the experiment, we used quantization $L = 3$ to solve (23). We see that `opt-ARQ` outperformed `WRR` for the entire range of packet tolerance delay, with maximum difference being 4.2%.

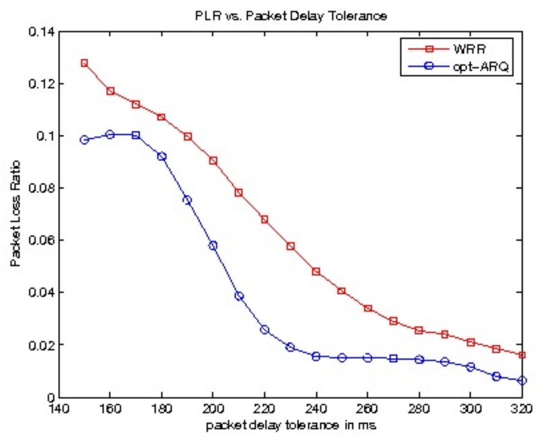


Figure 8. Performance Comparison for ARQ Schemes: PLR vs. Packet Delay Tolerance

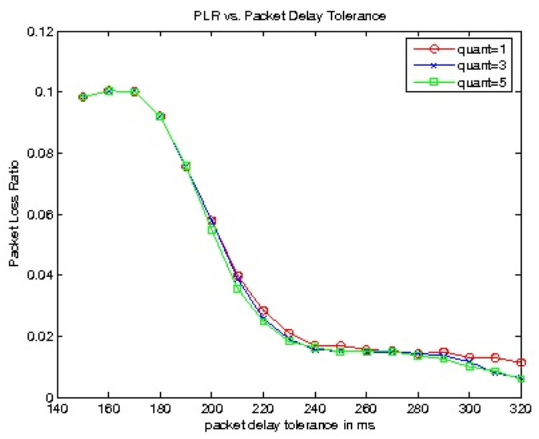
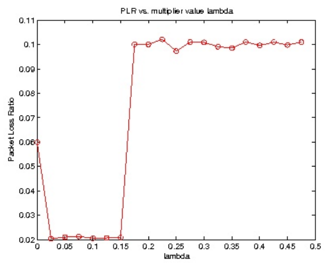
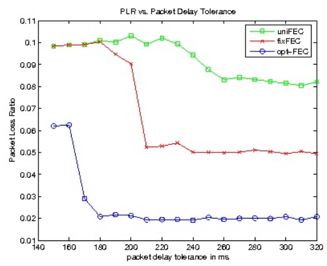


Figure 9. Performance Comparison for ARQ Schemes using different Quantization



a) PLR vs. Multiplier Value λ



b) PLR vs. Packet Delay Tolerance

Figure 10. Performance Comparison for FEC Schemes

Recall the performance of `opt-ARQ` depends on the quantization level L. To investigate the relative performance impact for changing L, we reran `opt-ARQ` for $L = 1$ and $L = 5$ as well. The results are shown in Figure 9. We first notice that the performance for all three quantization levels were quite similar. We next notice that the PLR improvement as L increases is diminishing: the maximum difference between $L = 1$ and $L = 3$ is 0.51%, while the maximum difference between $L = 3$ and $L = 5$ is 0.32%. Recalling that the complexity of (23) is proportional to L, we decided that a good performance-complexity tradeoff point is $L = 3$. This quantization level will be used throughout the rest of the experiments. Support for retransmission will be turned off for FEC-based algorithms.

8.2 FEC-based Algorithm

We next investigate the performance of FEC-based algorithms. We limited the feasible FEC set to be the set of RS(n, k)'s, $k < n \leq 5$. Recall in FEC-based algorithm (24) that the performance depends heavily on the selection of the Lagrange multiplier λ, which determines the weight of the penalty function $\lambda\left(\frac{n-k}{k}\right)$. To stress this point, we constructed Figure 10a, which shows the performance in PLR as a function of λ, which for each data point was held constant for the experimental run. We see that an inappropriate λ value can result in a poorer PLR by 8%, demonstrating the importance of a cleverly selected λ.

Using the linear regression method described in Section 7.2 to find the appropriate λ for given volume of packets in queues, we traced the performance of the optimal `opt-FEC` of (24) and plotted in Figure 10b. For comparison, we plotted two other FEC schemes in addition. `uniFEC` performs fixed channel coding RS$(4, 3)$ on the data packets and transmits over the *single* channel with the highest delivery success probability given current queue lengths and network conditions. `fixFEC`, like `uniFEC`, performs fixed RS$(4, 3)$ but strips over three channels using greedy algorithm `greedy2`. Both `uniFEC` and `fixFEC` will elect to send regular transmission if regular transmission has better delivery success probability due to delays introduced by FEC.

Several observations can be made in Figure 10b. 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. 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`. Finally, we see that `opt-FEC` outperformed both `uniFEC` and `fixFEC`,

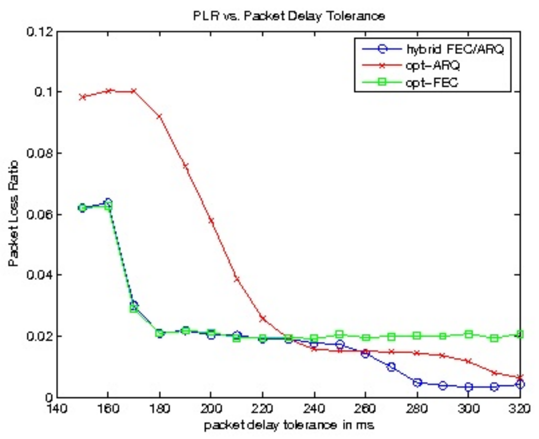


Figure 11. Performance Comparison among Hybrid FEC/ARQ, ARQ-based and FEC-based: PLR vs. Packet Delay Tolerance

demonstrating the value of dynamically controlling λ as the volume of packets in queue varies.

8.3 Hybrid FEC/ARQ Algorithm

Finally, we investigate the performance of the hybrid FEC/ARQ algorithm `hybrid FEC/ARQ` in (28) and (29). For comparison, the performance of `opt-FEC` and `opt-ARQ` are also plotted in Figure 11. We see that the performance of `hybrid FEC/ARQ` was at least as good as both `opt-FEC` and `opt-ARQ` for all range of packet delay tolerance.

9 Conclusion

With the advent of multi-homed devices with multiple WWAN links, aggregation of multiple channels can effectively address limited bandwidth issues. Packet striping in such environment is difficult because of bursty losses and random delays. In this paper we investigated and developed near-optimal striping algorithms for delay-sensitive packets over multiple burst-loss channels with random delays.

We first derived the packet loss ratio of an (n, k) Reed-Solomon code on a burst-loss channel. We then analyzed the performance of FEC over a set of bursty channels under a particular mapping. Given the number of mappings to a set of m channels is exponential, we derived a greedy algorithm that constructs near-optimal FEC mappings in linear time. We then extended our network loss model to include random transmission delays and bandwidth constraints. We developed dynamic programming-based algorithms that solve the striping problem for the ARQ-only case, the FEC-only case, and the hybrid FEC/ARQ case. Our simulation results show that our algorithm outperforms naïve algorithms such as weighted round-robin. Note that our striping scheme operates on a per-packet basis and not per-flow, and hence we can easily apply our developed techniques to multiple flows sharing multiple channels.

References

[1] H.-Y. Hsieh and R. Sivakumar, "A transport layer approach for achieving aggregate bandwidth on mutli-homed mobile hosts," in *Proceedings of the ACM MobiCom 2002*, Atlanta, GA, Sept. 2002, pp. 83–94.

[2] A. C. Snoeren, "Adaptive inverse multiplexing for wide area wireless networks," in *Proceedings of the IEEE GLOBECOM'99*, 1999.

[3] G. Cheung, P. Sharma, and S. J. Lee, "Striping delay-sensitive packets over multiple bursty wireless channels," in *IEEE International Conference on Multimedia and Expo*, Amsterdam, the Netherlands, July 2005.

[4] A. Mukherjee, "On the dynamics and significance of low frequency components of internet load," in *Internetworking: Res. Experience*, December 1994, vol. 5, pp. 163–205.

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

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

[7] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel," in *Proceedings of the ACM WoWMoM'99*, Seattle, WA, Aug. 1999, pp. 35–42.

[8] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," Tech. Rep. MSR-TR-2001-35, Microsoft Research, February 2001.

[9] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Asilomar Conference on Signals, Systems, and Computers*, November 2000.

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

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

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

[13] Stephen B. Wicker and Vijay K. Bhargava, Eds., *Reed-Solomon Codes and Their Applications*, John Wiley & Sons, 1999.

[14] 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 *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004)*, Montreal, Canada, May 2004.

[15] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison Wesley, 1994.

[16] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, Eds., *Numerical Recipes in C++*, Cambridge University Press, 2002.