STRIPING DELAY-SENSITIVE PACKETS OVER MULTIPLE BURSTY WIRELESS CHANNELS

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 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 packets by agateway to multiple channels, such that the overall performance is enhanced. We model and analyze the striping of delay-sensitive packets over multiple burst-loss channels. We derive the expected packet loss ratio when FEC (Forward Error Correction) and retransmissions are applied for error protection over multiple channels. We next model and analyze the case when the channels are bandwidth-limited. 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 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, 3]. Unlike previous work that focuses on bulk transfer, we focus our attention on delay-sensitive packet delivery such as media streaming.

2. CHANNEL MODEL BASICS

Given the burst loss nature of wireless links, we model each channel using a two-state Markov chain (Gilbert model). 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., $P(i) = \Pr(0^i|1)$. Given Gilbert model parameters p and q, $p(i)$ and $P(i)$ are:

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

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

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

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

$q(i)$ 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 and can be expressed as:

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

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

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

3. FEC FOR BURSTY CHANNELS

We derive the expected packet loss ratio (PLR) of FEC code — (n, k) Reed-Solomon code in particular α_{RS} — on a bursty channel. 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 packet transmitted (loss/success), giving us two conditional probabilities, $\alpha_{RS|1}$ and $\alpha_{RS|0}$, respectively. α_{RS} can then be expressed as:

$$\alpha_{RS} = \pi * \alpha_{RS|1} + (1-\pi) * \alpha_{RS|0} \tag{6}$$

where $\pi = \frac{p}{p+q}$ is the raw packet loss ratio (PLR).

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 $\alpha_{RS|1}$:

$$\begin{aligned}\alpha_{RS|1} = &\sum_{i=0}^{k-1} \left(\frac{i+1}{k}\right) r(i,k-1) \sum_{j=[n-k-i]^+}^{n-k} R(j,n-k) \\ &+\sum_{i=1}^{k-1} \left(\frac{i}{k}\right) \bar{r}(i,k-1) \sum_{j=[n-k+1-i]^+}^{n-k} S(n-k-j,n-k)\end{aligned} \tag{7}$$

where $[x]^+$ is the positive part of x. Similar analysis can be performed to derive $\alpha_{RS|0}$.

4. STRIPING FEC DATA

Data and parity packets of a given RS(n, k) can be striped over a set of 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 $\mathsf{g} : (k, n-k) \rightarrow (\mathbf{u}, \mathbf{v}),\ \mathbf{u}, \mathbf{v} \in \mathcal{I}^m$. 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 data packets dropped 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:

$$X = \begin{cases} k-Y & \text{if } Y+Z \leq k-1 \\ 0 & \text{o.w.} \end{cases} \tag{8}$$

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

$$\begin{aligned} E[X] &= E[k-Y | Y+Z \leq k-1] P(Y+Z \leq k-1) \\ &= (k - E[Y|\Theta \leq k-1])\, P(\Theta \leq k-1) \end{aligned} \tag{9}$$

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:

$$Y = \sum_{i=1}^{m} Y_i, \quad Z = \sum_{i=1}^{m} Z_i, \quad \Theta = \sum_{i=1}^{m} \Theta_i \tag{10}$$

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

$$P(\Theta_i = j) = \pi_i\, R(w_i - j, w_i) + (1-\pi_i) S(j, w_i) \tag{11}$$

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

$$G_\Theta(\xi) = E[\xi^\Theta] = E[\xi^{\Theta_1}] \cdots E[\xi^{\Theta_m}] = G_{\Theta_1}(\xi) \cdots G_{\Theta_m}(\xi)$$

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 [4]):

$$P(\Theta = j) = \frac{1}{j!} \frac{d^j}{d\xi^j} G_\Theta(\xi) \bigg|_{\xi=0} \tag{12}$$

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

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

$$\begin{aligned} E[Y|Y+Z \leq k-1] &\approx E[Y|Y \leq k-1] \\ &= \sum_{j=1}^{k-1} \frac{jP(Y=j)}{P(Y \leq k-1)} \end{aligned} \tag{13}$$

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

4.1. 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

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

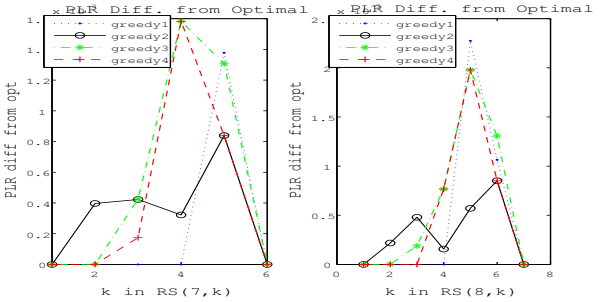


Fig. 1. PLR for different FEC distribution search algorithms

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 left-over 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 bursty 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 $1 \leq x \leq n-1$. 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`.

`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 1, 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.

5. DELAY-SENSITIVE TRAFFIC OVER BANDWIDTH-LIMITED CHANNELS

We add bandwidth limitations to our previous burst loss model as shown in Figure 2. 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 fixed delay Δ_j and Gilbert-modeled bursty 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: $(l_j + 1)/\mu_j + \Delta_j$.

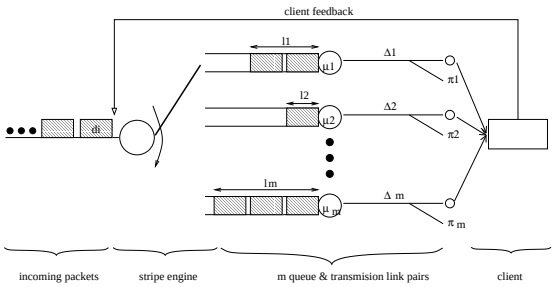


Fig. 2. Model of bandwidth-limited channels.

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 smallest expiration times. We assume striping engine is activated whenever there is a packet in the incoming queue.

5.1. ARQ-based Algorithm

We assume we 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 correctly and 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 correctly and timely delivered using (re)transmission (ARQ). Let $f^{(i)}_{ARQ}(d')$ be the probability that the same packet is correctly delivered on time if channel i is first used for ARQ. Assuming the client can errorlessly inform the striping engine of the packet loss (packet loss ratio π_i), the packet has a chance for retransmission with a tighter deadline.

$$\begin{aligned} f(d') &= \begin{cases} f_{ARQ}(d') & \text{if } d' \geq 0 \\ 0 & \text{o.w.} \end{cases} \\ f_{ARQ}(d') &= \max_{i=1\ldots m} f^{(i)}_{ARQ}(d') \\ f^{(i)}_{ARQ}(d') &= \begin{cases} (1-\pi_i) + \pi_i f(d' - D^{(i)}_T - D_F) & \text{if } d' \geq D^{(i)}_T \\ 0 & \text{o.w.} \end{cases} \end{aligned} \tag{14}$$

where $D^{(i)}_T = \frac{l_i+1}{\mu_i} + \Delta_i$ is the transmission delay for channel i and D_F is the feedback delay for the receiver to inform the striping engine of the loss event, same for all channels.

We can solve (14) using dynamic programming (DP), where each time $f_{ARQ}(d')$ or $f^{(i)}(d')$ is called, the optimal solution is stored in the $[i, d']$ entry of a DP table, so that each repeated subproblem is solved only once. Assuming d' and $D^{(i)}_T + D_F$ are integers, the complexity of (14) is $O(md)$. If they are not integers, we need to round *down* d' and round *up* $D^{(i)}_T + D_F$ for an approximate solution.

5.2. FEC-based Algorithm

As mentioned, we use greedy algorithm `greedy2` to find a sub-optimal but good FEC distribution. We will denote such

Table 2. Model Parameters for Experiment

chnl	p	q	μ	Δ
1	0.05	0.45	$30ms/pkt$	$80ms$
2	0.03	0.27	$30ms/pkt$	$80ms$
3	0.05	0.4	$25ms/pkt$	$100ms$

distribution as g from now on. In addition, we consider only RS $(n, n-1)$ while varying n for different channel coding strength. We bound the delay of using FEC, given mapping function g, as the maximum delay experienced by a packet in the n-packet group:

$$D_T = \max_{i=1,\ldots,m} \left[\frac{l_i + u_i + v_i}{\mu_i} + \Delta_i \right] \tag{15}$$

Let $f_{FEC}(d'_1), d'_1 = d_1 - t$, be probability that a packet with expiration d_1 is correctly and timely delivered using FEC. Because $f_{FEC}(d'_1)$ affects $n-1$ data packets, it is the average success probability of the first $n-1$ packets in the head of the incoming packet queue. Given RS $(n, n-1)$, function g results in PLR $\pi(\mathsf{g})$. $f_{FEC}(d'_1)$ is:

$$\begin{aligned} f_{FEC}(d'_1) &= \max_n \left[\frac{1}{n-1} \sum_{i=1}^{n-1} f_n^{\mathsf{g}}(d'_i) - \lambda \frac{n}{n-1} \right] \\ f_n^{\mathsf{g}}(d'_i) &= \begin{cases} (1-\pi(\mathsf{g})) & \text{if } d'_i \geq D_T \\ 0 & \text{o.w.} \end{cases} \end{aligned} \tag{16}$$

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

Notice there is a *penalty* term $\lambda(\frac{n}{n-1})$ in (16). The reason is that using RS $(n, n-1)$ invariably increases the traffic volume by factor $n/(n-1)$. Hence a penalty term is used to regulate the packet volume so that it does not lead to queue overflow. λ is selected inverse-proportionally to the total amount of traffic currently in the m queues.

5.3. Hybrid Algorithm with FEC and ARQ

We can now combine the ARQ and FEC algorithms into one hybrid algorithm. $f(d'_1)$ is now simply the larger value of the two possible choices — (re)transmission or FEC:

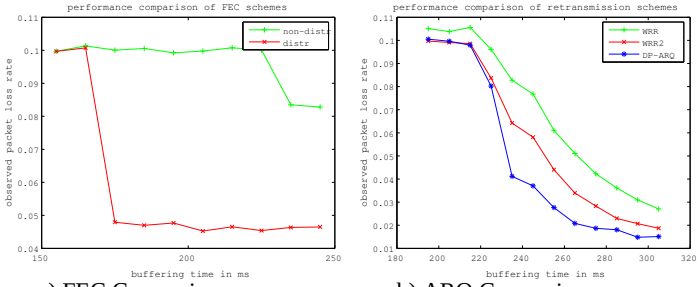
$$f(d'_1) = \begin{cases} \max[f_{ARQ}(d'_1), f_{FEC}(d'_1)] & \text{if } d'_1 \geq 0 \\ 0 & \text{o.w.} \end{cases} \tag{17}$$

where $f_{FEC}(d'_1)$, unlike (16), is now defined recursively to permit retransmission:

$$\begin{aligned} f_{FEC}(d'_1) &= \max_n \left[\frac{1}{n-1} \sum_{i=1}^{n-1} f_n^{\mathsf{g}}(d'_i) - \lambda \frac{n}{n-1} \right] \\ f_n^{\mathsf{g}}(d'_i) &= \begin{cases} (1-\pi(\mathsf{g})) + \pi(\mathsf{g}) f(d'_i - D_T - D_F) & \text{if } d'_i \geq D_T \\ 0 & \text{o.w.} \end{cases} \end{aligned} \tag{18}$$

5.4. Experimental Results

We implemented a simulator in C running on `linux` to test the developed striping engine. In the first experiment, we validate the performance of near-optimal FEC distribution algorithm `greedy2` striping FEC packets across bursty channels. Given input packet rate of $62.5pkt/s$ and FEC RS$(4, 3)$,



a) FEC Comparison b) ARQ Comparison

Fig. 3. Performance Comparison of FEC and ARQ Schemes

greedy stripes packets across three channels with parameters shown in Table 2. Figure 3a shows the resulting PLR of `greedy2` (`distr`) as a function of packet end-to-end delay tolerances in ms, over a non-distributed FEC scheme (`non-distr`), which uses a single channel for each set of FEC packets. Two observations can be made. First, the benefit of FEC kicks in earlier for `distr` than `non-distr`. This is because spreading FEC packets has lower end-to-end delay than non-spreading. Second, even after the benefit of `non-distr` kicks in for large end-to-end delay, `distr` is still better. This is because spreading FEC across channels has the effect of de-correlating the bursty channels, and RS works better in uncorrelated channels than correlated ones.

In the second experiment, we validate the performance of the striping engine in band-limited scenario. Figure 3b shows the performance of optimal ARQ scheme over two weighted round-robin (WRR) schemes. `WRR` randomized only according to bandwidth of channels, while `WRR2` randomized only using channels that the pending packet has a chance to arrive on-time given current buffer occupancies. We observe that after end-to-end delay tolerance is sufficiently large to tolerate at least one retransmission, the optimal `ARQ` outperformed `WRR` and `WRR2`.

6. CONCLUSION

We presented algorithms for optimal striping of delay-sensitive packets over multiple burst loss channels. Our algorithms find an operating region to balance conflicting channel characteristics such as loss, latency and bandwidth to reap the benefit of aggregating multiple channels. Simulation results show that our algorithm outperforms naïve algorithms such as weighted round-robin.

7. REFERENCES

[1] C. Carter and R. Kravets, "User device cooperating to support resource aggregation," in *Proceedings of the IEEE WMCSA 2002*.

[2] 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.

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

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