EMS: Encoded Multipath Streaming for Real-time Live Streaming Applications

Alix L.H. Chow[1], Hao Yang[2], Cathy H. Xia[3], Minkyong Kim[2], Zhen Liu[4], Hui Lei[2]
University of Southern California, Los Angeles, CA 90089[1]
IBM T. J. Watson Research Center, Hawthorne, NY 10532[2]
Ohio State University, Columbus, OH 43210[3]
Nokia Research Center, Beijing, China[4]
Email: lhchow@usc.edu[1], {haoyang,minkyong,hlei}@us.ibm.com[2], xia.52@osu.edu[3], zhnliu@yahoo.com[4]

***Abstract*—Multipath streaming protocols have recently attracted much attention because they provide an effective means to provide high-quality streaming over the Internet. However, many existing schemes require a long start-up delay and thus are not suitable for interactive applications such as video conferencing and tele-presence. In this paper, we focus on real-time live streaming applications with stringent end-to-end latency requirement, say several hundreds of milliseconds. To address these challenges, we take a joint multipath and FEC approach that intelligently splits the FEC-encoded stream among multiple available paths. We develop an analytical model and use asymptotic analysis to derive closed-form, optimal load splitting solutions, which are surprisingly simple yet insightful. To our best knowledge, this is the first work that provides such closed-form optimal solutions. Based on the analytical insights, we have designed and implemented a novel Encoded Multipath Streaming (EMS) scheme for real-time live streaming. EMS strives to continuously satisfy the application's QoS requirements by dynamically adjusting the load splitting decisions and the FEC settings. Our simulation results have shown that EMS can not only outperform the existing multipath streaming schemes, but also adapt to the dynamic loss and delay characteristics of the network with minimal overhead.**

I. Introduction

Live streaming applications have become increasingly popular nowadays, driven by the widespread adoption of broadband networks. Such applications typically have stringent quality of service requirements, and the transport protocol must ensure the streaming content is delivered in time, with minimal losses. Multipath streaming (e.g., [1], [2], [3], [4], [5], [6]) is one promising approach that leverages the availability of multiple paths between endhosts and exploits such path diversity to improve the streaming quality. Most of these multipath streaming schemes (e.g., [1], [2], [3], [4], [6]) also apply Forward Error Correction (FEC) encoding in the stream, so that the receiver can recover from packet losses without retransmission. With the joint benefits of multipath and FEC, these schemes can successfully provide high-quality streaming of pre-stored or live media content. However, in order to handle bursty losses, the existing streaming systems typically require a long startup delay, several seconds or even longer (e.g., [5], [7]), at the receiver.

This work was done partly while Alix L.H. Chow was working as a summer intern at the IBM T.J. Watson Research Center.

In this paper, we focus on *real-time live streaming* applications that have stringent end-to-end delay requirement, say a few hundreds of milliseconds. Examples of such applications include video conferencing, remote surgery, tele-presence and virtual reality. The interactive nature of these applications precludes the use of a large startup buffer, and the stream content must be delivered within a very short time window after it is generated. According to the ITU-T/G.114 recommendation [8], the end-to-end delay should be no more than 150 ms for highly interactive tasks. Moreover, these live streaming applications also require high-fidelity stream delivery for pleasant user experience. For example, with MPEG-4 encoding, there will be noticeable video quality degradation when the loss rate is above 1%.

We tackle the above challenges in real-time live streaming through both theoretical and practical treatments. We first study the central issue of how to split the stream across multiple paths so as to minimize the information loss. We develop an analytical model and provide asymptotic analysis that allows us to derive closed-form solutions for optimal load splitting. Our analysis shows that, in a general setting of heterogeneous paths, it is asymptotically optimal to split the traffic such that the load on each path is inversely proportional to its loss rate. These surprisingly simple results can offer valuable insights for the protocol design. To the best of our knowledge, *this is the first work that provides closed-form optimal solutions for multipath load splitting*, as the previous work (e.g., [2], [3], [4], [6]) all involve heavy combinatorics computation.

Inspired by these analytical results, we have designed a novel Encoded Multipath Streaming (EMS) scheme for real-time live streaming. EMS addresses practical issues, such as network dynamics and the coupling effects of multipath and FEC, through a set of adaptation mechanisms. Specifically, EMS uses an adaptive search process to dynamically adjust the load splitting decision around the asymptotic optimal one. It also dynamically adjusts the FEC settings, including both FEC group size and FEC redundancy level, based on the current network conditions and the application requirements. For real-time streaming, these FEC settings present multiple tradeoffs between the error correction capability and the delay

performance. Through simple adaptation mechanisms, EMS strives to find a good balance where it satisfies the application requirements with minimal overhead.

We have evaluated the performance of EMS using extensive simulations, and compared it to two existing multipath streaming schemes, namely the TCP-based DMP scheme [5] and another Gilbert Model-based UDP scheme. Our results show that EMS consistently outperforms these existing schemes under different network settings, with orders of magnitude lower information loss rate. The results also confirm the effectiveness of the load splitting and FEC setting adaptation mechanisms in EMS.

In summary, the main contributions of this work are:

- the first asymptotic analysis and closed-form solutions for optimal load splitting in multipath streaming with FEC;
- the design of a novel real-time live streaming scheme with adaptive load splitting and FEC settings;
- an extensive simulation study on the comparison of different approaches and the impact of network dynamics.

II. Modeling and Asymptotic Analysis

In this section, we present an analytical model and asymptotic analysis for the multipath streaming problem using FEC. Most previous work (e.g., [1], [3], [9]) focused on exact distribution analysis, which involves heavy combinatorics computation and is hard to give insight. Here we advocate an asymptotic approximation-based approach from which we can derive closed-form solutions and provide useful engineering insights.

A. Model

Consider a streaming application that generates CBR stream data at rate λ and needs to transmit the data in real-time from sender s to receiver d. The streaming application will use FEC with group size K, redundancy r for the transmission. We assume a simple FEC scheme as follows: For every group of K data packets, we generate $N = K(1+r)$ packets. We refer to these N packets as a FEC group. The encoding scheme is such that, if the number of losses within a FEC group is less than or equal to $N-K$, then we can reconstruct the original K data packets with that FEC group.

There are I potential overlay paths between nodes s and d. As in [1], we assume the network loss rate on each path is dominated by some on and off background traffic on its bottleneck link, and use a Gilbert model to model the loss process. Such a model captures the dependence in consecutive packet losses, and is known to be more accurate than the independent loss model since packet losses tend to occur in a burst due to buffer overflow. Specifically, the packet loss process along path i is modeled as a two-state continuous time Markov process (known as the Gilbert model) $\{S_i(t)\}$, with $S_i(t) \in \{0,1\}$. A packet transmitted at time t is considered lost if the state of path i is $S_i(t) = 1$, and otherwise considered successfully delivered. The infinitesimal generator for this Gilbert model of path i is $\mathbf{Q}_i = \begin{bmatrix} -\mu_0^{(i)} & \mu_0^{(i)} \\ \mu_1^{(i)} & -\mu_1^{(i)} \end{bmatrix}$. Let $\mu_\Sigma^{(i)} = \mu_0^{(i)} + \mu_1^{(i)}$. For simplicity, assume $\mu_\Sigma^{(i)} \equiv \mu_\Sigma$. In steady-state, path i is in the lossy state with probability $\pi_i = \mu_0^{(i)}/\mu_\Sigma$.

Consider a simple multipath routing strategy under load splitting vector $\mathbf{x} = [x_i]$, $\sum_i x_i = 1$ as follows. For every N packets, exactly Nx_i packets[1] will use path i and they are evenly spaced throughout the time window K/λ with interarrival times $\tau_i = \frac{1}{\lambda(1+r)x_i}$. This can be achieved by a proportional round robin scheme based on the weight vector $\mathbf{x}$.

Let $X_k^{(i)} = 0$ (resp. $= 1$) if the kth transmitted packet on path i is lost (resp. successfully delivered). Under the Gilbert model, the packet delivery process $\{X_k^{(i)}\}$ on path i forms a discrete time Markov chain, with transition matrix $\mathbf{P}_i = \begin{bmatrix} 1-\alpha_i & \alpha_i \\ \beta_i & 1-\beta_i \end{bmatrix}$, where $\alpha_i = (1-\pi_i)e^{-\mu_\Sigma \tau_i}$ and $\beta_i = \pi_i e^{-\mu_\Sigma \tau_i}$. For small τ_i, they can be expressed approximately:

$$\alpha_i \approx \mu_1^{(i)} \tau_i, \qquad \beta_i \approx \mu_0^{(i)} \tau_i. \tag{1}$$

Denote $Y_n^{(i)} := \sum_{k=1}^n X_k^{(i)}$ the number of successes in n transmissions on path i. Then the total number of successes in n transmissions over all paths under load splitting vector $\mathbf{x}$ is $Z_n(\mathbf{x}) = \sum_{i=1}^I Y_{nx_i}^{(i)}$. For a FEC group of N packets, if $Z_N(\mathbf{x})$ is smaller than K, then it is not possible to recover all K original data packets within the FEC group. We refer to, $L_K(\mathbf{x}) := P(Z_{K(1+r)}(\mathbf{x}) < K)$, as the *group loss rate*. The percentage of data that cannot be recovered within a FEC group is typically called the *information loss rate*.

As presented in [1], the exact distribution of $Z_n(\mathbf{x})$ can be derived using a complex two-dimensional recursive procedure with heavy combinatorics computation. We next present an asymptotic analysis on Z_n which is more mathematically tractable and helps providing useful guidelines in searching for optimal design that achieves the least information loss.

B. Asymptotic Analysis

Using the well-known central limit argument, one can show that $Y_n^{(i)}$ is asymptotically normal (see [10][p. 138]) with

$$\begin{aligned} E[Y_n^{(i)}] &\sim n\frac{\alpha_i}{\alpha_i+\beta_i} =: n \cdot m_i, \\ Var[Y_n] &\sim n\frac{\alpha_i\beta_i(2-\alpha_i-\beta_i)}{(\alpha_i+\beta_i)^3} =: n \cdot v_i(x_i). \end{aligned}$$

Based on (1), we have $m_i = 1-\pi_i$ and $v_i(x_i) = \frac{b_i}{a}(\rho x_i - a)$, with $b_i = \pi_i(1-\pi_i)$, $\rho = \frac{2\lambda}{\mu_\Sigma}$, and $a = \frac{1}{1+r}$.

Consequently, $Z_n(\mathbf{x}) = \sum_i Y_{nx_i}$ is also asymptotically normal with mean $n \cdot m(\mathbf{x})$ and variance $n \cdot v(\mathbf{x})$, where $m(\mathbf{x}) := \sum_i x_i m_i$, and $v(\mathbf{x}) := \sum_i x_i v_i(x_i)$.

For large K, we can approximate L_K as follows.

$$L_K(\mathbf{x}) \approx \Phi\left(\frac{K - \frac{K}{a}\cdot m(\mathbf{x})}{\sqrt{\frac{K}{a}\cdot v(\mathbf{x})}}\right) = \Phi\left(\frac{a - m(\mathbf{x})}{\sqrt{\hat{v}(\mathbf{x})/K}}\right),$$

[1]To be technically correct, it should be $\lfloor Nx_i \rfloor$ packets instead since we can only take integer values. This, however, will not matter much since we are interested in asymptotic analysis for large N.

where $\Phi(\cdot)$ denotes the cumulative distribution function of a standard normal, and $\hat{v}(\mathbf{x}) = a \cdot v(\mathbf{x})$. Let $\hat{v}_i(x_i) = b_i(\rho x_i - a)$, then $\hat{v}(\mathbf{x}) = \sum_i x_i \hat{v}_i(x_i)$.

Similar (but more complex) asymptotic approximation can be derived for the information loss rate. Our simulation results have shown that the optimal decision that minimizes the information loss rate is quite similar to that minimizes the group loss rate. It is thus sufficient to focus on minimizing the group loss rate L_K.

C. Optimal Load Splitting

In order to find the optimal load splitting scheme $\mathbf{x}$ that minimizes the group loss rate $L_K(\mathbf{x})$, it suffices to solve

$$\max_{\mathbf{x}:\sum_i x_i=1} \frac{m(\mathbf{x}) - a}{\sqrt{\hat{v}(\mathbf{x})/K}}. \tag{2}$$

It can be shown that the optimal solution to (2) is unique and can be obtained by solving the equations from Karush-Kuhn-Tucker (KKT) conditions.

Observe that there are interesting tradeoffs between single path versus multipath load splitting: On one hand, sending more traffic to the less lossy (higher m_i) path increases $m(\mathbf{x})$, encouraging the usage of the single (best) path; on the other hand, the smaller x_i, the smaller the variance $v_i(x_i)$, encouraging the usage of multipath to disperse the burstiness of traffic and reduce variance.

Remark: Note that if one assumes that the packet delivery process on each path is i.i.d. (such as [11]), then $v_i(x_i) \equiv b_i$ and the benefit of dispersing the bursty traffic over multipath is not captured. In this case, one can show that using the single best path always achieves the least information loss.

1) Homogeneous Case: Suppose all paths are homogeneous with $\pi_i \equiv \pi$. Then $m(\mathbf{x}) - a = 1 - \pi - a$ does not depend on $\mathbf{x}$. In this case, $\hat{v}(\mathbf{x})$ becomes Schur convex [12] since it is convex and symmetric. We then claim the following theorem:

Theorem 1. *Suppose the paths are homogeneous.*

i) *If $a < 1-\pi$, then the optimal solution to* (2) *is multipath with equal splitting;*
ii) *If $a > 1 - \pi$, then the optimal solution to* (2) *is single path with no splitting.*

2) Heterogeneous Case: Without loss of generality, assume the I paths are ordered such that $\pi_1 \le \pi_2 \le ... \le \pi_I (< \frac{1}{2})$. To make the problem interesting, assume [2] $m_1 > a$. Therefore, the optimal solution $\mathbf{x}^*$ should satisfy: $m(\mathbf{x}^*) - a > 0$.

Using a simple interchange argument, we have the following monotonicity result on the optimal load splitting.

Theorem 2. *Suppose $m_1 > a$. The optimal solution to* (2) *must satisfy*

$$x_1^* \ge x_2^* \ge \ldots \ge x_I^*.$$

That is, the optimal load splitting tends to assign more load to less lossy paths.

[2]If instead, all $m_i \le a$, then $m(\mathbf{x}) - a < 0$ under all load splitting policy $\mathbf{x}$. Since a is a control parameter, one should obviously set $m_1 > a$, which will achieve a lower group loss rate.

Proof: Suppose on the contrary, the optimal solution $\mathbf{x}^*$ to (2), has $x_i^* < x_j^*$ for some $i < j$, where $\pi_i < \pi_j$.

Let $\tilde{x}_i = x_i^* + \epsilon$, $\tilde{x}_j = x_j^* - \epsilon$, and $\tilde{x}_k = x_k^*$ for $k \ne i, j$, for some small $\epsilon > 0$. Then $m(\tilde{\mathbf{x}}) - m(\mathbf{x}^*) = \epsilon(m_i - m_j) > 0$, and $\hat{v}(\tilde{\mathbf{x}}) - \hat{v}(\mathbf{x}^*) = 2\epsilon\rho[b_i(x_i^* - \frac{a}{2\rho}) - b_j(x_j^* - \frac{a}{2\rho})] + \epsilon^2\rho(b_i + b_j) < 0$, for ϵ sufficiently small, where the last inequality holds because $x_i^* < x_j^*$ and $b_i < b_j$.

Since $m(\mathbf{x}^*) - a > 0$, we then have $\frac{m(\tilde{\mathbf{x}})-a}{\sqrt{\hat{v}(\tilde{\mathbf{x}})}} \ge \frac{m(\mathbf{x}^*)-a}{\sqrt{\hat{v}(\mathbf{x}^*)}}$, which contradicts the assumption that $\mathbf{x}^*$ is optimal to (2). ■

The next theorem establishes the asymptotic behavior for the high redundancy case.

Theorem 3. *As the redundancy r becomes large, the optimal load splitting approaches*

$$x_i^* = \frac{c}{\pi_i}, \quad i = 1, \ldots, I,$$

where $c^{-1} = \sum_{i=1}^{I} \frac{1}{\pi_i}$. That is, as the FEC redundancy gets large, the optimal load x_i on path i becomes inversely proportional to the path loss rate π_i.

Proof: We solve (2) by dropping K (since it does affect the optimal solution) and using the Lagrangian: $J(\mathbf{x}) = \frac{m(\mathbf{x})-a}{\sqrt{\hat{v}(\mathbf{x})}} + \theta \sum_{i=1}^{I} x_i$. Setting $\frac{\partial J}{\partial x_i} = 0$, we have

$$-m_i + \frac{m(\mathbf{x}) - a}{\hat{v}(\mathbf{x})} b_i(\rho x_i - a/2) = \theta\sqrt{\hat{v}(\mathbf{x})}. \tag{3}$$

Having a weighted sum of (3) with weights x_i, we then have: $\theta = \frac{-a + \frac{m(\mathbf{x})-a}{\hat{v}(\mathbf{x})}(\sum_i b_i x_i)\frac{a}{2}}{\sqrt{\hat{v}(\mathbf{x})}}$. As redundancy r gets larger (i.e.$a \to 0$), we have $\theta \to 0$, and equation (3) becomes: $-m_i + \frac{m(\mathbf{x})}{\hat{v}(\mathbf{x})} b_i \rho x_i = 0$. Thus

$$x_i = \frac{m_i}{b_i \rho} \frac{\hat{v}(\mathbf{x})}{m(\mathbf{x})} = \frac{1}{\pi_i \rho} \frac{\hat{v}(\mathbf{x})}{m(\mathbf{x})}.$$

It is easily checked that, by setting $x_i = \frac{c}{\pi_i}$, for all i, we have $\frac{\hat{v}(\mathbf{x})}{m(\mathbf{x})} = c\rho$. Since $\sum_i x_i = 1$, we have $c^{-1} = \sum_{i=1}^{I} \frac{1}{\pi_i}$. ■

D. Model Validation

We have validated the above analytical results using extensive simulations, in which we use brute-force search to find the optimal splitting in different scenarios. Due to the lack of space, we only present the results of a 2-path case in Figure 1 presents We defer the detailed simulation settings to Section IV, and it suffices to note that the loss rates on Path 1 and Path 2 are 5.29% and 8.26%, respectively. Thus, based on Theorem 3, we can calculate the asymptotic optimal splitting as <x_1=61%, x_2=39%>.

In these simulations, we fix the FEC group size to 20 and gradually increase the FEC redundancy from 0% to 60%. For each redundancy value, we vary the load on Path 1 (x_1) from 0 to 100%, with an increment of 3%. The load on Path 2 is always $x_2 = 1 - x_1$. The results of such brute-force search are shown in Figure 1, which plots how the information loss rates changes as we vary x_1. We can see that as the FEC redundancy gets large, the optimal splitting vector is indeed approaching the asymptotic optimal one as provided by Theorem 3.

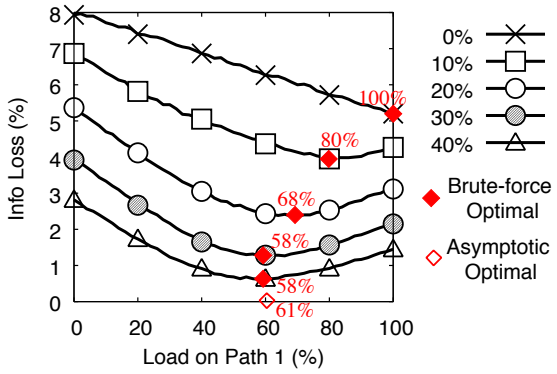

Fig. 1. Optimal splitting (2 paths)

III. Design

In this section, we present the design of our Encoded Multipath Streaming (EMS) protocol for live streaming applications. EMS is inspired by the optimal load splitting scheme and takes an adaptation oriented approach in addressing practical issues such as network dynamics and FEC settings. As a result, EMS can exploit the benefits of both multipath delivery and FEC to ensure high-quality streaming with minimal overhead.

A. EMS Overview

EMS assumes that the sender and the receiver can establish multiple paths between them through external mechanisms. In practice, this can be done by using multiple network interfaces or overlay networks (e.g., [13], [14]), where the traffic is relayed at multiple overlay nodes. However, the establishment and optimization of these overlay paths are out of the scope of EMS.

Figure 2 shows an overview of our proposed EMS protocol. An EMS sender takes a live stream, generated in real time, and encodes it using a FEC encoder. The encoded stream is splitted among multiple available paths and arrives at the receiver with different delay and loss characteristics. The EMS receiver buffers all the received packets and, if it has received enough packets within any FEC group, uses a FEC decoder to recover any lost packets in this group. However, any packet that has missed its deadline, even if it is successfully recovered by FEC, will be discarded.

The above seemingly simple EMS operation poses several research challenges in terms of its effectiveness and efficiency. Specifically, we need to address the following design issues:

- **How to split load among multiple paths under practical and dynamic network settings?** While the analytical results of optimal load splitting offer invaluable insights, they are obtained based on the assumptions of stationary loss processes and a large FEC group size. In practice, however, the path quality is highly dynamic as the background traffic changes, and the FEC group size is typically small. Thus it remains a design challenge for EMS to operate consistently with optimal or near-optimal load splitting vectors.
- **How to properly set the FEC parameters such as redundancy and group size?** Many existing FEC-based streaming protocols (e.g., [1], [3], [4]) have overlooked this issue and simply use static, empirical settings. However, when coupled with the use of multipath, these FEC parameters present multiple dimensions of tradeoff between delay, loss and overhead, thus playing a key role in the overall performance of EMS. For example, we have seen in Section II-D that the FEC redundancy has direct impact on the optimal load splitting. We will elaborate on these design tradeoffs in Sections III-C and III-D.

To address these two issues, we take an adaptation oriented approach as follows. As shown in Figure 2, an EMS receiver continuously monitors the quality of each path in terms of loss and delay, as well as the aggregated information loss rate after FEC recovery. These monitoring results are periodically fed into the EMS Decision Engine, which adapts the load splitting vector as well as the FEC parameters. Subsequently, the sender receives these decisions via feedback from the receiver, and then updates its local settings for the FEC encoder and packet scheduler accordingly. Note that EMS places most of the intelligence at the receiver, because it can readily observe the path quality without any extra signaling mechanism. In the rest of this section, we will describe how EMS adapts the load splitting, FEC redundancy and FEC group size respectively.

B. Online Load Splitting (OLS)

The Online Load Splitting (OLS) scheme in EMS is inspired by the previous asymptotic analysis yet addresses the practical issues of finite FEC group size and network dynamics through an adaptive search process. In order to bootstrap the system, OLS initially splits the load evenly over all the available paths, so that the receiver can measure the loss rate along each path. After collecting T_s seconds of such measurement, the receiver computes the asymptotic optimal solution based on Theorem 3 and passes the resulting splitting vector to the sender, which then changes its packet scheduler accordingly.

Note that the asymptotic optimal solution is only an approximation of the true optimal, which itself is unknown and dynamically changing. Therefore, after applying the asymptotic optimal solution, OLS starts to search for the true optimal solution and adapt to network dynamics in an iterative manner. As illustrated in Algorithm 1 below, OLS sorts the paths in the increasing order of loss rate. It operates in either of two possible states: *PickPath* and *AdaptLoad*. In the *PickPath* state, OLS chooses the first path in the list, i.e., the one with the lowest loss rate among the candidates, and switches to the *AdaptLoad* state. In the *AdaptLoad* state, OLS increases the load on the chosen path by Δ_L, a pre-defined increment term (we set Δ_L as 3% of the stream rate in our implementation). OLS also decreases Δ_L amount of traffic from the other paths collectively, with the load decrement on each path proportional to its loss rate. After applying these load adjustment, OLS measures the information loss rate within a window of T_s seconds and compares it to the previously recorded one. If the loss rate decreases, OLS stays in the *AdaptLoad* state and repeats the load adjustment. Otherwise, it removes the previously chosen path from the list, reverts to the previous

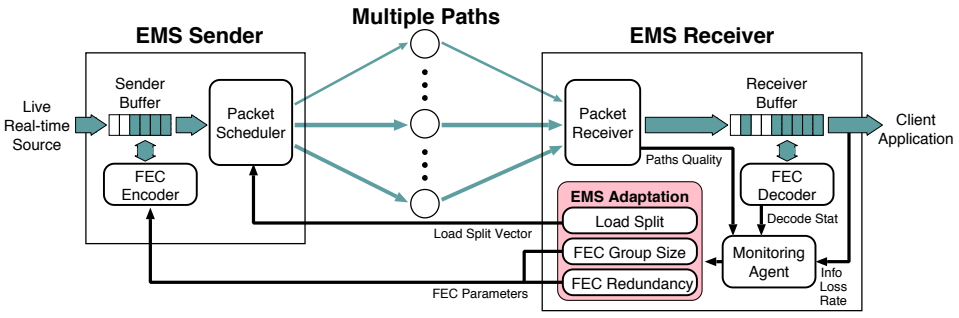

Fig. 2. EMS overview

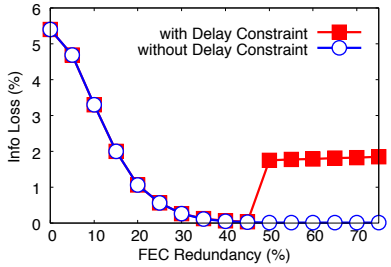

Fig. 3. Info loss vs. FEC redundancy

load splitting, and switches to the *PickPath* state to choose another path. Finally, when all paths have been tried, OLS sorts the paths again and restarts the process.

Pseudo Code 1 Online Load Splitting (OLS)

```
Compute the asymptotic optimal solution (Theorem 3) and split the load accordingly
Sort the paths in the increasing order of loss rate
repeat
    Pick the first path in the list
    repeat
        Increase the load on the chosen path by ΔL
        Decrease the load on each of the remaining paths by a fraction of δ, proportional to their respective loss rates
    until measured information loss rate increases
    Remove the chosen path from the list
    Revert to the previous load splitting
until the path list is empty
goto Step 2
```

Note that the load splitting vector only decides the amount of traffic that each path should carry. The sender can enforce it through different packet scheduling algorithms, resulting in different characteristics of the splitted traffic. The packet scheduler in EMS (Figure 2) seeks to minimize the burstiness of traffic as follows. With a given load splitting vector, it keeps track of the offered load and expected load on each path, both in terms of the number of packets. Whenever a new packet arrives, it is scheduled to the path which has the largest gap between the offered and expected loads. This way, EMS can enforce the splitting vector yet ensure the packets sent on each path are paced out as evenly as possible.

C. FEC Redundancy

There is an inherent tradeoff between the FEC overhead and its error correction power. With more redundant packets, the receiver can recover from more severe losses, at the cost of more traffic offered to the network. One may be attempted to view this tradeoff as one between a user and the network (or other users in the network). That is, if a user increases the FEC redundancy for its own traffic, it can always enjoy the enhanced error correction capability, while the network or other competing users may see unpleasant side-effects due to the increased load. However, our simulation results show that this is not the case once we take into account the latency requirement, and it is to a user's own disadvantage to inject excessive FEC redundancy.

Figure 3 depicts how the information loss rate changes as the FEC redundancy increases. In these simulations (detailed settings in Section IV), there are two paths between the sender and the receiver, with 5.4% losses on each path. The load is evenly split between the two paths as suggested by Theorem 1. As we can see, with the absence of delay constraint, larger FEC redundancy can always improve the information loss rate despite the diminishing returns. However, with a 150 ms delay constraint, there is a sudden increase in the information loss rate as the FEC redundancy goes beyond 50%. This is because excessive redundant packets cause queue buildup at bottleneck links and longer queuing delays for all packets traversing those links. Because a packet arriving after its deadline will be discarded, such delay-induced losses become apparent as the end-to-end latency approaches the bound, and the receiver cannot use FEC to recover these delayed packets either. Similar phenomenon has also been reported in the literature using prototype experiments [3].

To balance between the loss recovery capability and the latency, we have designed a FEC redundancy adaptation scheme, illustrated in Algorithm 2. Our basic idea is to use "just enough" FEC redundancy to meet the application's loss requirement. Suppose L_{req} is the maximum loss rate that the application can tolerate. An EMS receiver keeps track of the percentage of packets that arrived after their deadlines. If such delay-induced loss exceeds αL_{req}, where α is a pre-defined threshold, EMS will decrease the FEC redundancy. Otherwise, the receiver compares the current information loss rate L_c with L_{req}. If L_c is low (i.e., smaller than βL_{req}), EMS will also decrease the FEC redundancy. However, if L_c is relatively high (i.e., larger than γL_{req}), EMS will increase the FEC redundancy. In our implementation, we set $\alpha = 0.5$, $\beta = 0.5$, $\gamma = 0.75$ and the redundancy increment $\Delta_R = 5\%$.

Pseudo Code 2 FEC Redundancy Adaptation

```
backoff = (delay-induced losses > αL_req)? true:false
if (backoff || information loss rate < βL_req) then
    Decrease FEC redundancy by Δ_R
else if (information loss rate > γL_req) then
    Increase FEC redundancy by Δ_R
end if
goto 1
```

It should be noted that EMS adapts the FEC redundancy in a continuous manner, based on both loss and delay measurement. As such, it can react quickly to network dynamics

(e.g., a sudden increase of background traffic) and minimize the FEC overhead subject to the application's loss requirement.

D. FEC Group Size

The FEC group size also affects the tradeoff between loss recoverability and delay in the presence of bursty losses. With a larger FEC group, the impact of bursty losses becomes less significant, thus the receiver has a better chance to recover the lost packets. However, a larger FEC group also results in longer delay for the recovered packets, because recovery only starts after the receiver have received K packets (data or redundant). In the extreme cases, a packet may already have passed its deadline when it is recovered, thus making it effectively useless.

To address this issue, EMS seeks to use the maximal FEC group size that can still ensure recovering the lost packets before their deadlines. Specifically, for every recovered packet, the receiver records its slack time, defined as the difference between its recovery timestamp and its deadline. Note that a packet that has missed its deadline has a negative slack time. The receiver keeps track of the minimum slack time, denoted as T_s, within a sliding window. At the end, it adapts the FEC group size K as follows:

$$K \leftarrow K + \lfloor T_s/\lambda \rfloor - b \tag{4}$$

where λ is the stream source rate in unit of packets per second, and b is a constant (say 3) that prevents the increase in group size from using up the slack time. In essence, we estimate the number of packets that the source can generate during the minimum slack time, and then adjust the FEC group size accordingly. Note that in the case of a negative T_s, the FEC group size will be effectively decreased.

IV. Performance Evaluation

In this section, we evaluate the performance of EMS and compare it to two existing multipath streaming schemes, one based on TCP and the other on UDP. Our results show that EMS consistently outperforms these existing schemes under different settings, and EMS can adapt to network dynamics and provide high-quality streaming with minimal overhead.

A. Methodology

We evaluate different multipath streaming schemes using ns-2 simulations. Figure 4 shows the network topology in use, which consists of multiple virtual paths between a sender and a receiver. In practice, these paths can be established through multi-homing or network overlay. We evaluate the impact of different network characteristics by varying the propagation delay and background traffic on these paths. Specifically, each path consists of one bottleneck link, which has 10 ms propagation delay, 50 Mbps capacity and a router queue length of 50 packets. The total propagation delay along a path is set to 20, 40, or 100 ms, which represents the typical delay found on Internet paths within the same coast, across the coast, and across the continent, respectively.

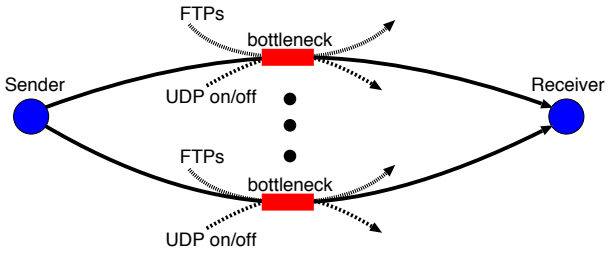

Fig. 4. Network topology used in our simulations

To simulate the bursty loss behavior, each bottleneck link is shared with background traffic that is composed of 10 FTP sessions and one UDP On/Off traffic. The On time and Off time of the UDP traffic each follows an exponential distribution with a given mean value. We vary the two mean values (On and Off time) to introduce different loss characteristics, but fix the sum of them as 0.1 second. When the UDP traffic is active, it sends 1440 B UDP packets at 50 Mbps rate. Except for the results in Section IV-C (to be explained later), we do not directly set the loss rate for a path. Instead, we control it by varying the UDP traffic and report the measured loss rates.

The streaming sender generates CBR traffic at 1900 Kbps, which is a typical 720p HD video streaming rate (e.g., in iTunes Store). The packet size is 500 B. The end-to-end delay requirement is set as 150 ms, based on the ITU-T/G.114 recommendation for highly interactive tasks [8]. Only those packets that arrive before the deadline are considered to be useful. To ensure good video quality, the loss requirement is set as below 1%, which is the typical loss tolerance level with MPEG-4 video. Each simulation run starts with a 30-second warm-up phase (for the background traffic to ramp up), followed by a 3-hour streaming phase. Table I summarizes the default parameter settings in the EMS implementation.

TABLE I
Default EMS Settings

Parameter	Value
Loss Requirement, L_{req}	1%
Delay Constraint	150ms
OLS Step Size, Δ_L	3%
OLS Adapt Window	60s
FEC Adapt Window	30s
FEC Redundancy Adapt Step Size, Δ_R	5%
FEC Redundancy Adapt Threshold, α and β	0.5
FEC Redundancy Adapt Threshold, γ	0.75
FEC Redundancy Init Value	10%
FEC Group Size Init Value	10
FEC Group Size Adapt Threshold, b	3

The primary metric used to compare different schemes is the *information loss rate*, defined as the percentage of streaming packets that fail to arrive before their deadlines (including both lost packets and late-arrival packets).

B. Comparison with TCP-based DMP Streaming

We first compare EMS with the TCP-based DMP streaming scheme [5]. DMP utilizes multiple paths by maintaining a TCP connection on each path. The sender puts data packets in a single sender queue. At any time, only one TCP connection

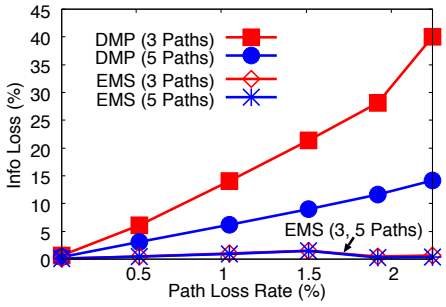

Fig. 5. DMP comparison: Info loss vs path loss (3 or 5 paths)

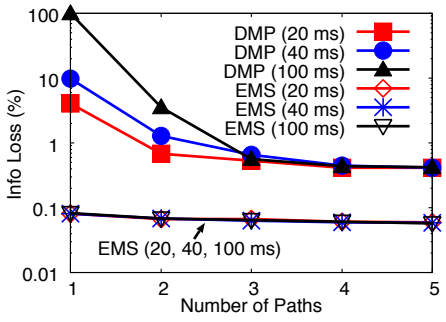

Fig. 6. DMP comparison: Info loss vs # paths (Path loss 0.06%)

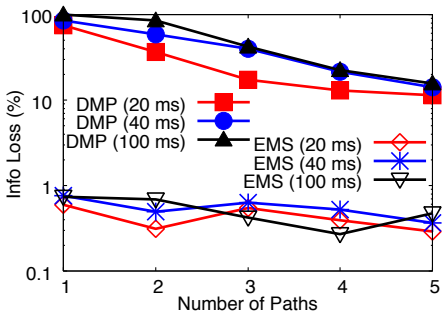

Fig. 7. DMP comparison: Info loss vs # paths (Path loss 2.24%)

can gain the access to the sender queue. The winning TCP connection will keep sending data until the connection is blocked. Another available TCP connection will then gain the access to the sender queue and continue sending data.

Figure 5 plots the information loss rates with EMS and DMP respectively, as the path loss increases from 0.06% to 2.24%. The path propagation delays are fixed as 40 ms. To yield different path loss rates, the on-time of background UDP traffic varies between 0.005 and 0.03 s. From Figure 5 we can make two observations as follows. First, EMS can meet the demanding requirements of high-quality live streaming (delay $<$ 150 ms, loss $<$ 1%) in all scenarios. In contrast, DMP meets the requirements only when the paths are almost loss-free. Secondly, DMP can effectively exploit the existence of more paths to improve the streaming quality, but it is very sensitive to the underlying network losses. On the other hand, EMS can consistently maintain high rate of in-time delivery even though the paths become more lossy, thanks to the use of online load splitting and adaptive FEC mechanisms.

Next we evaluate the impact of the number of available paths and the delay along each path. For illustration purposes, we focus on the two extreme cases, one with very light path losses (0.06%) and the other with heavy path losses (2.24%).

Figure 6 shows the information loss rate when the network is under-utilized and the path loss rate is as low as 0.06%. In such cases, EMS can satisfy the loss requirement with any number of paths, but DMP can only achieve satisfactory performance with 3 or more paths. As expected, as more paths are available, the information loss rate with DMP decreases, because it is less likely that all paths are congested at the same time. However, when there are only one or two paths, DMP is sensitive to the path delays, because the TCP throughput drops as the path delay increases.

When the paths become more congested and lossy, as shown in Figure 7, EMS can still satisfy the loss requirement in all scenarios. In contrast, even with 5 paths, DMP has an information loss rate as high as 10%, which is not acceptable for multimedia streaming. The reason is because the TCP congestion window is throttled by heavy losses, and many packets miss their deadlines as they are queued either at the sender, waiting for an available TCP connection, or at the receiver, waiting for an earlier packet to be retransmitted.

While EMS uses FEC to proactively protect its traffic from network losses, one important question is how much overhead it incurs. To address this concern, Table II lists the FEC settings that EMS uses under different settings. We can see that the FEC overhead in EMS is moderate, ranging from 8.6% to 32%. More importantly, EMS can effectively exploit the path diversity and reduce the FEC redundancy as more paths are available. For example, with 3 paths, the FEC overhead is at most 12%. EMS also adapts the FEC group size to the path delay, which further reduces the FEC overhead when the paths have short delays. A more detailed study of such FEC adaptation mechanisms will be presented later in this section.

TABLE II
FEC SETTINGS USED BY EMS:(K,N) AND REDUNDANCY

	Number of Paths				
	1	2	3	4	5
20ms	(53,63) 18.9%	(54,62) 14.8%	(53,58) 8.6%	(53,58) 8.6%	(53,58) 8.6%
40ms	(44,52) 18.2%	(44,50) 13.6%	(43,47) 9.3%	(43,47) 9.3%	(43,47) 9.3%
100ms	(25,33) 32.0%	(25,30) 20.0%	(25,28) 12.0%	(25,28) 12.0%	(25,28) 12.0%

Finally, we compare the average packet delay with EMS and DMP under different settings, which is plotted in Figure 8. Here the path loss rate is again 2.24%. Not surprisingly, the average packet delay with DMP is much higher than that in EMS. Such a large delay is mostly caused by the TCP reliable and in-order delivery mechanisms. While DMP can utilize multiple paths to significantly reduce the packet delay, it still cannot satisfy applications that demand very short end-to-end delay, say a few hundreds of milliseconds.

In summary, the above results indicate that EMS can provide live streaming with stringent delay and loss requirements, while DMP works well only in restricted settings where multiple nearly loss-free paths are available. However, we would like to stress that DMP was designed for video-on-demand type of streaming applications, which can tolerate several seconds of playback delay [5]. It is indeed well suited for such applications, as it effectively exploits the path diversity to improve the streaming quality, and the use of TCP makes DMP friendly to other competing TCP traffic. Nevertheless, one should take caution in applying such TCP-based schemes in live streaming applications, such as video conferencing, in which both ends need to be tightly synchronized.

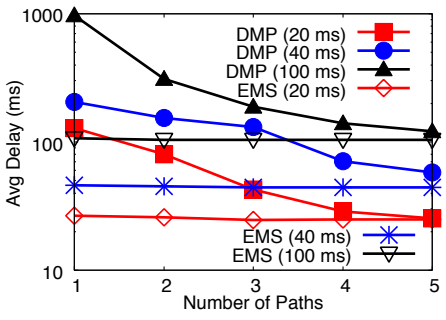

Fig. 8. DMP comparison: Average delay vs # paths (Path loss 2.24%)

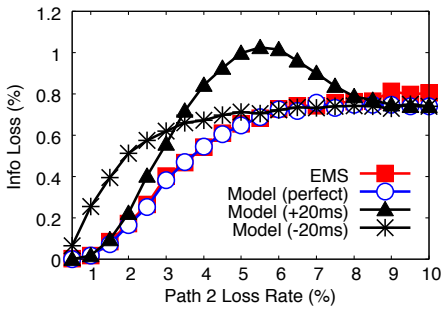

Fig. 9. Model-based scheme comparison: Info loss vs Path loss (10% FEC redundancy)

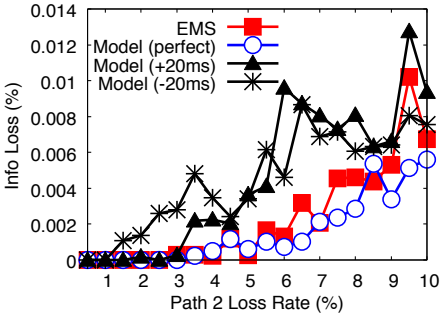

Fig. 10. Model-based scheme comparison: Info loss vs Path loss (40% FEC redundancy)

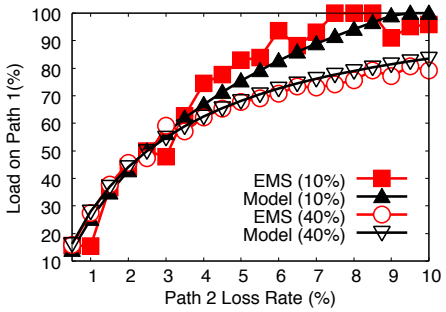

Fig. 11. Model-based scheme comparison: Load splitting vs Path loss (10% or 40% FEC redundancy)

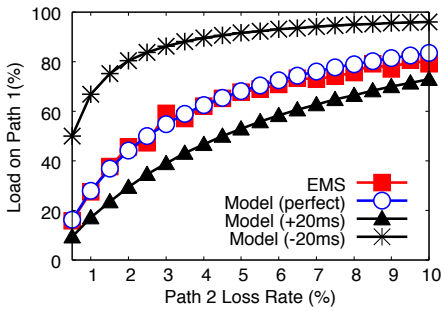

Fig. 12. Model-based scheme comparison: Load splitting vs. Path loss (40% FEC redundancy)

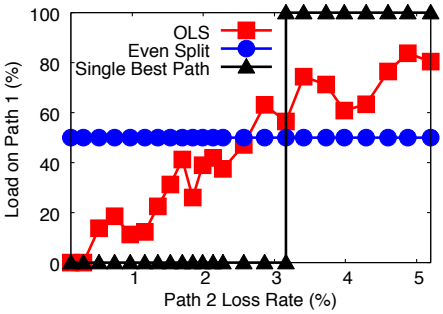

Fig. 13. OLS: Load splitting under different schemes

C. Comparison with Model-based UDP Multipath Streaming

In this subsection, we compare EMS with an existing UDP-based multipath streaming scheme similar to [4], which seeks to optimally allocate load over multiple available paths using a model-driven approach. It assumes the packet loss process on each path follows the Gilbert Model, and the streaming application can accurately estimate the Gilbert Model parameters. With such parameters in place, it finds the optimal load allocation vector numerically. In what follows, we refer to this scheme as the *model-based* scheme. In contrast, our EMS scheme does not assume or make use of any specific packet loss processes; instead, it takes an adaptive approach to dynamically adjust the load on each path.

The model-based scheme suffers from practical limitations that the loss processes may not follow Gilbert Model, and it is hard to accurately estimate the model parameters in real time. Nevertheless, its theoretical optimality provides an important comparison base to gauge how well EMS performs. For this purpose, we carry out this comparison using simulation settings that are ideal for the model-based scheme. Specifically, we remove all background traffic and emulate packet losses using the Gilbert Model. We consider both the "perfect" case, where the model parameters are precisely known, and the "imperfect" case, where the model parameters are estimated with slight errors.

For simplicity, we only consider two paths in this set of experiments. In the Gilbert Model, the sum of average good time and average bad time is fixed as 0.1 sec. For Path 1, the model parameters are fixed to achieve a resultant path loss rate of 2.5%. We vary the model parameter for Path 2 such that the resultant path loss rate varies from 0.5% to 10%. Because the model-based scheme also uses FEC parameters in the computation, we disabled the FEC adaptation mechanism in EMS and experimented with static FEC settings (40-packets group, either 10% or 40% redundancy). The path propagation delays are fixed as 40 ms.

Figure 9 and Figure 10 plot the information loss rate of EMS, perfect model and imperfect models, with 10% and 40% FEC redundancy respectively. With imperfect models, there is a ± 20 ms error in the estimated Gilbert Model parameter (average bad time) for Path 1. We can make two important observations from these two figures.

First, the information loss rates yielded by EMS and the perfect model are very close in all cases. In fact, with 10% FEC redundancy, the performance of EMS and the perfect model is almost identical. This shows that EMS can make near-optimal load splitting decisions, even though it does not utilize any knowledge of the loss processes.

Secondly, the model-based scheme is sensitive to the accuracy of the model parameters. With only 20ms error in the estimation, the information loss rate may increase from 0.6% to 1%. Note that in our simulations, the loss processes indeed follow the Gilbert Model and the parameters are assumed to be known. In practice, the application has to frequently send probe packets along each path and fit the probing results with the Gilbert Model. This process is inevitably error-prone, which casts doubts on the performance of such model-based schemes in practical scenarios.

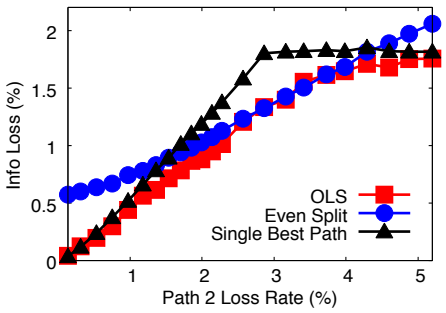

Fig. 14. OLS: Info loss under different schemes

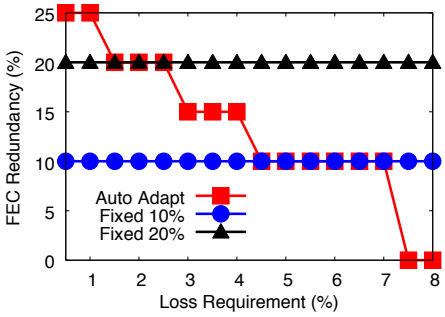

Fig. 15. FEC Redundancy Adaption: Amount of FEC Redundancy

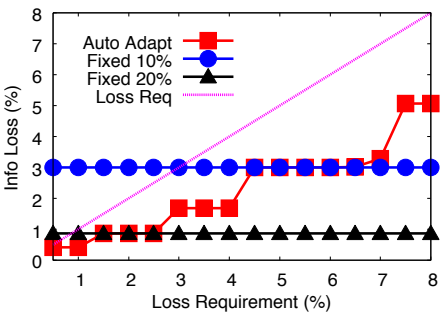

Fig. 16. FEC Redundancy Adaption: Info Loss

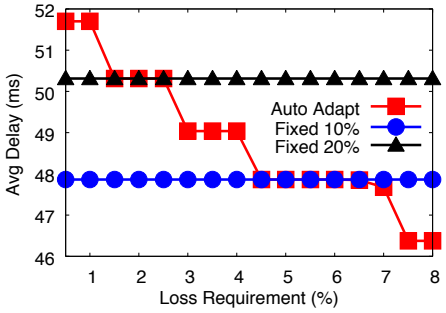

Fig. 17. FEC Redundancy Adaption: Average Delay

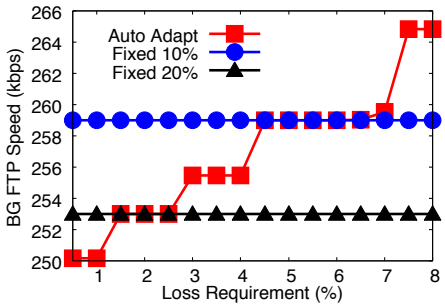

Fig. 18. FEC Redundancy Adaption: Background FTP Speed

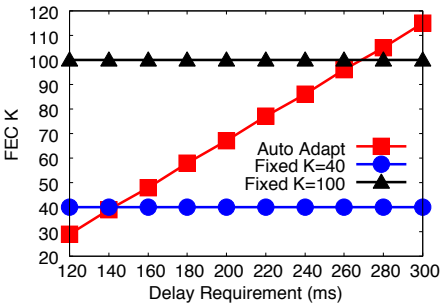

Fig. 19. FEC Group Size Adaption: Info Loss

To validate these results, we also plot the load splitting vectors decided by EMS and the model-based scheme in Figure 11 and Figure 12. We can see from Figure 11 that the OLS scheme in EMS indeed converges to the optimal load splitting solution, as computed by the model-based scheme with accurate parameters. When the loss rate of Path 2 increases, both schemes assign more traffic on Path 1. Furthermore, as the FEC redundancy increases from 10% to 40%, the load split becomes relatively more even. These observations are consistent with the analysis in Section II. On the other hand, Figure 12 shows the inaccuracy in the model parameters can lead to distorted load splitting decisions, which account for the performance degradation as we discussed earlier.

In summary, EMS can achieve near-optimal traffic allocation among available paths, yet avoid the practical hurdle of accurately modeling the packet loss processes. We also note that the existing UDP-based schemes all use static FEC settings, while EMS can dynamically adjust the FEC parameters based on application requirements and network conditions.

D. Effectiveness of OLS

Now we evaluate how well the Online Load Splitting (OLS) scheme in EMS works. Again, we consider two paths, where Path 1 has a loss rate of 2.9% (with background UDP on-time as 40 ms) and Path 2 has a loss rate ranging from 0.1% to 5.2% (with background UDP on-time ranging from 5 ms to 80 ms). To stay focused, we disabled the FEC adaption in EMS and used a static FEC setting with 40-packet groups and 10% redundancy. The delays of both paths are 40 ms. We compare OLS to two alternative schemes, namely *Even-Split* and *Single-Best-Path*. *Even-Split* assigns equal amount of traffic on each path; *Single-Best-Path* detects the path with the lowest loss rate and assigns all traffic on that single path.

Figure 13 plots the amount of load on Path 1 with different schemes, and shows that OLS splits the load according to the loss characteristic of different paths. Figure 14 shows the corresponding information loss rates. The result shows that OLS consistently gives the lowest information loss rate. It is better than *Even-Split*, especially when paths are more heterogeneous, and it also performs better than *Single-Best-Path*, especially when the paths have similar path loss rates.

In summary, OLS can adjust the load splitting based on the path loss characteristics, thus achieving low information loss rate under all settings.

E. Effectiveness of FEC Redundancy Adaptation

Next we evaluate the performance of FEC redundancy adaptation in EMS, using two homogeneous paths with a path loss rate of 5.1%. We disabled OLS and FEC group size adaption in this experiment, and used *Even-Split* and 40-packet FEC groups. Figure 15 plots the amount of FEC redundancy decided by our adaptation scheme under different loss requirements. Figures 16, 17, and 18 show the corresponding information loss rate, average packet delay, and average throughput of the background FTP sessions, respectively.

We can see from Figure 15 that EMS can adjust the amount of FEC redundancy according to the loss requirement, by using higher (or lower) redundancy for more stringent (or more relaxed) loss requirement. Figure 16 shows that, with EMS adaptation, the information loss rate is always kept

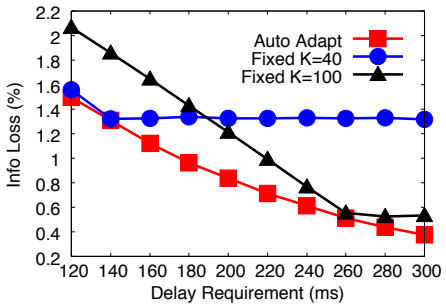

Fig. 20. FEC Group Size Adaption: Info loss

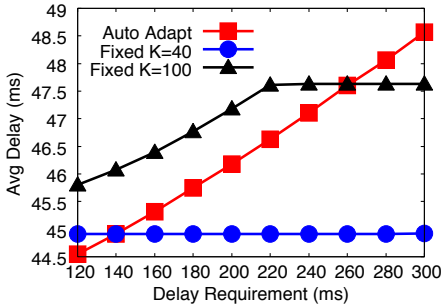

Fig. 21. FEC Group Size Adaption: Average delay

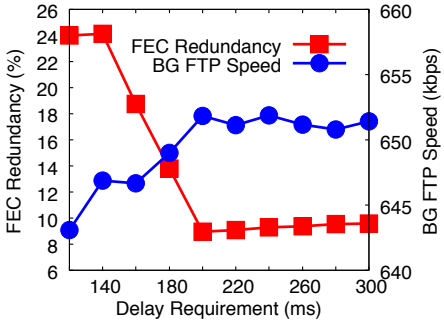

Fig. 22. FEC Group Size Adaption: Joint FEC adaptation

below the requirement. If we use a fixed FEC redundancy amount, e.g., 10% or 20%, it can either result in violating the loss requirement or incur unnecessary traffic overhead. For example, if we use 10% FEC redundancy, it cannot satisfy any information loss requirement below 2.9%. If we use 20% FEC redundancy, it can satisfy the loss requirement ranging from 0.5 to 8%. However, it uses more overhead than EMS adaptation scheme, while both can satisfy the information loss requirement.

From Figure 17, we can see the delay reduced along with more relaxed loss requirement. This is due to i) shorter queuing delay in the network as less traffic is injected, and ii) less FEC recovered packets (which normally has long recovery delay). From Figure 18, we can see that the background FTP traffic have a faster speed with more relaxed loss requirement because less FEC redundant packets are sent on the network.

In summary, by adaptively changing the amount of FEC redundancy, EMS can satisfy the loss requirement with minimum overhead.

F. Effectiveness of FEC Group Size Adaptation

We also study the effectiveness of FEC group size adaptation in EMS using two homogeneous paths (loss rate fixed as 2.9%). We disabled OLS and FEC redundancy adaption in this experiment, and used *Even-Split* and 10% FEC redundancy. Figure 19 shows the FEC group size (K) decided by the EMS adaptation scheme along with different application delay constraint. Figures 20 and 21 show the corresponding information loss rate and average packet delay, respectively.

As shown in Figure 19, EMS can adjust the FEC group size according to the delay constraint. A larger group size is used for longer delay constraint, which results in lower information loss rate. We notice from Figure 20 that, compared to a fixed FEC group size, our adaptation scheme can result in the lowest information loss rate for all different delay constraint. Generally, the larger the FEC group is, the more tolerant it is to the high path loss rate. However, the FEC recovery time also becomes longer, resulting in larger delays. Thus, the applications with stringent delay constraint will experience high information loss rate. Figure 21 shows that the packet delay increases along with the delay constraint for our adaptation scheme. It is because a larger FEC group size is used in our adaptation scheme when the delay constraint is more relaxed.

Finally, we turned on the FEC redundancy adaption and repeated the above experiments. The joint effects of both adaptation mechanisms are illustrated in Figure 22, which plots the FEC redundancy level chosen by EMS as well as the speed of the background FTP traffic. Interestingly, when the delay requirement is relaxed, EMS also reduces the FEC redundancy. This is because in such cases, EMS will choose a larger FEC group size (as explained in Figure 19), which provides stronger recovery strength. As such, EMS can ensure the same loss requirement with a lower FEC redundancy level, which in turn reduces the network overhead and leads to the throughput gains in the competing background FTP traffic.

In summary, the FEC group size adaptation in EMS can efficiently exploit the delay/loss tradeoff and ensure both in-time FEC recovery and maximum recovery strength.

V. Related Work

Multipath streaming has recently attracted much attention for improving the quality of streaming applications, and a broad overview of this research area can be found in [15]. Many existing multipath streaming schemes (e.g., [1], [2], [3], [4], [6]) also use FEC to guard against packet losses. Our EMS scheme differs from these existing works in several aspects. First, EMS targets on live streaming from a single source, while most existing schemes assume that the stream is pre-stored at the senders. Secondly, EMS strives to provide real-time streaming with stringent delay requirement, whereas the existing schemes have largely ignored the delay performance as the receiver can maintain a large playback buffer. Lastly, existing schemes all use static FEC settings. However, as shown in Section III, the FEC settings play a key role in the end-to-end delay, thus EMS uses adaptive schemes to dynamically adjust the FEC redundancy and group size.

While several recent works, such as DMP [5] and MPLOT [11], have proposed multipath data delivery schemes between a single source and a receiver, they cannot be directly applied in real-time live streaming. As shown in Section IV-B and [5], the startup delay with DMP is in the order of seconds, which is not suitable for real-time applications. MPLOT is a transport protocol which focuses on maximizing the application throughput, yet it does not address tight delay constraints.

While MPLOT can greatly benefit applications with bulk data transfer, a live streaming source cannot effectively leverage such throughput benefits because its streaming rate is typically fixed or bounded, e.g., by the encoding schemes.

Optimal load splitting in multipath streaming with FEC has been studied in, e.g., [2], [3], [4], [6]. However, their analysis involve heavy combinatorics computation and cannot directly offer any engineering insight. Their optimal solutions also require fairly accurate knowledge of the path loss model, which is difficult to obtain in practice. In contrast, we derive closed-form optimal solutions via asymptotic approximation, and further design a practical load splitting scheme using adaptive searching. As shown in Section IV-C, EMS can achieve near-optimal load splitting without any assumption of the path loss model.

In the context of multimedia streaming, splitting workload on multipath with an objective to minimize the media distortion have been proposed in, e.g., [16], [17]. Under single path streaming, adaptive FEC has been studied in [18] for VoIP applications. These works are closely tied with the media compression scheme and are based on the knowledge of path loss model. In contrast, EMS doe not assume any specific type of streaming data or specific loss model. Also, the FEC adaptation schemes in EMS address the joint effects of FEC and multipath.

Kurant [19] proposes to utilize the delay differences between multiple paths by assigning more workload (or sending packets with larger spacing) on shorter delay paths. Similar to other model-based approaches, [19] requires the knowledge of path loss model, which is not required in EMS. Furthermore, EMS exploits the delay in multipath streaming by adapting the FEC settings.

Finally, many P2P streaming systems such as PPLive [20] and CoolStreaming [21] have been widely used on the Internet. Their goal is to broadcast video streams to a large number of users, with the users exchanging data among themselves to reduce the source workload. Measurement studies (e.g., [7], [22]) have shown that such systems incur large end-to-end latency, ranging from several seconds to a few minutes, thus are not suitable for real-time live streaming.

VI. Conclusions

In this paper, we study multipath streaming for the emerging real-time live streaming applications with tight delay and loss requirements. Through modeling and asymptotic analysis, we have developed closed-form solutions for optimal load splitting in multipath streaming with FEC. We have also presented the design and evaluation of EMS, a novel multipath streaming protocol with built-in load splitting and FEC adaptation mechanisms. The effectiveness of EMS is confirmed by simulation results under various settings. As part of our ongoing research, we plan to perform extensive evaluation experiments in the Internet, e.g., using PlanetLab. We also plan to pursue the analysis of optimal FEC settings and use it to further enhance the EMS protocol.

Acknowledgments

The authors would like to thank Dr. Bing Wang for providing the ns-2 codes of DMP Streaming and the useful discussions. We also thank the anonymous reviewers for their insightful comments. Alix L.H. Chow's research was funded partly by the USC Annenberg Graduate Fellowship and the NSF 0540420 grant.

References

[1] L. Golubchik, J. C. Lui, T. F. Tung, A. L. Chow, W.-J. Lee, G. Franceschinis, and C. Anglano, "Multi-path continuous media streaming: What are the benefits?" *Performance Evaluation*, vol. 49, no. 1-4, pp. 429–449, Sep 2002.

[2] B. Abdouni, W. C. Cheng, A. L. Chow, L. Golubchik, W.-J. Lee, and J. C. Lui, "Multi-path streaming: Optimization and performance evaluation," in *Proceedings of SPIE MMCN*, Jan 2005.

[3] A. L. Chow, L. Golubchik, J. C. Lui, and A. W.-J. Lee, "Multi-path streaming: Optimization of load distribution," *Performance Evaluation*, vol. 62, no. 1-4, pp. 417–438, Oct 2005.

[4] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 315–326, Apr 2004.

[5] B. Wang, W. Wei, Z. Guo, and D. Towsley, "Multipath live streaming via TCP: Scheme, performance and benefits," in *Proceedings of ACM CoNEXT*, Oct 2007.

[6] Y. Li, Y. Zhang, L. Qiu, and S. Lam, "SmartTunnel: Achieving reliability in the internet," in *Proceedings of IEEE INFOCOM*, May 2007.

[7] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale P2P IPTV system," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1672–1687, Dec 2007.

[8] "ITU-T recommendation G.114," International Telecommunication Union, Tech. Rep., 1993.

[9] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of packet loss processes in high-speed networks," *IEEE Transactions on Information Theory*, vol. 39, no. 1, pp. 98–108, Jan 1993.

[10] D. Cox and H. Miller, *The Theory of Stochastic Processes*. Chapman and Hall, 1965.

[11] V. Sharma, S. Kalyanaraman, K. Kar, K. Ramakrishnan, and V. Subramanian, "MPLOT: A transport protocol exploiting multipath diversity using erasure codes," in *Proceedings of IEEE INFOCOM*, Apr 2008.

[12] A. Marshall and I. Olkin, *Inequalities: Theory of Majorization and Its Applications*. New York: Academic Press, 1979.

[13] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of ACM SOSP*, Oct 2001.

[14] S.-J. Lee, S. Banerjee, P. Sharma, P. Yalagandula, and S. Basu, "Bandwidth-aware routing in overlay networks," in *Proceedings of IEEE INFOCOM*, Apr 2008.

[15] J. Apostolopoulos and M. Trott, "Path diversity for enhanced media streaming," *IEEE Communications Magazine*, vol. 42, pp. 80–87, Aug 2004.

[16] V. R. Gandikota, B. R. Tamma, and C. S. R. Murthy, "Adaptive FEC-based packet loss resilience scheme for supporting voice communication over ad hoc wireless networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 10, pp. 1184–1199, Oct 2008.

[17] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 629–641, Apr 2007.

[18] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for internet telephony," in *Proceedings of IEEE INFOCOM*, Mar 1999.

[19] M. Kurant, "Exploiting the path propagation time differences in multipath transmission with FEC," in *Proceedings of IEEE INFOCOM*, Apr 2009.

[20] *PPLive*, http://www.pplive.com/.

[21] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "DONet/CoolStreaming: A data-driven overlay network for peer-to-peer live media streaming," in *Proceedings of IEEE INFOCOM*, Mar 2005.

[22] G. Marfia, A. Sentivelli, S. Tewari, M. Gerla, and L. Kleinrock, "Will IPTV ride the peer-to-peer stream?" *IEEE Communications Magazine*, vol. 45, no. 6, pp. 86–92, Jun 2007.