Overlay Multi-hop FEC Scheme for Point-to-Point Video Streaming over Peer-to-Peer Networks

Yufeng Shan[1], Ivan V. Bajić[2], and Shivkumar Kalyanaraman[1]

1/ ECSE Department, Rensselaer Polytechnic Institute, Troy, NY 12180

2/ ECE Department, University of Miami, Coral Gables, FL 33146

Abstract

In this paper, we focus on the problem of providing lightweight support at *selected* intermediate overlay forwarding nodes to achieve increased error resilience on a *single* peer-based overlay path for point-to-point video streaming. We propose a novel "Overlay Multi-hop FEC" (OM-FEC) scheme that is *flexible* and considers the error characteristics of each overlay hop. We partition the end-to-end overlay path into *segments* and provide error resilience over those segments. Architecturally, this flexible design lies between the end-to-end and hop-by-hop paradigms, and we argue that it is well suited to peer-based overlay networks. We evaluate our work by both simulations and real-world implementation using a real video streaming application. These evaluations show that OM-FEC outperforms a pure end-to-end strategy by 10-15 dB in terms of video PSNR, and can be much more efficient than a heavyweight hop-by-hop resilience strategy.

Index terms

Peer-to-peer networks, overlay networks, video streaming, forward error correction.

I. INTRODUCTION

Providing high-quality video streaming over the current best-effort Internet is a challenging problem due to the characteristics of video data such as high bit rate requirement, delay and loss sensitivity. Streaming media distribution has been an intensively studied research topic in the past several years. Most recently, peer-to-peer (P2P) architectures and overlay networks are gaining attention. Padmanabhan et al. [1] discussed the problem of distributing streaming media content, both live and on demand, to a large number of receivers in a scalable way. They propose a solution called CoopNet, an approach for content distribution that combines aspects of infrastructure-based and peer-to-peer based content distribution, where clients cooperate to distribute content, thereby alleviating the load on the server. CoopNet builds multiple distribution trees spanning the source and all the receivers for its multiple description coded media content. Yeo et al. in [2] propose an application level multicast overlay using peering technology and a lightweight gossip mechanism to monitor prevailing network conditions and to improve the tree robustness. Client can dynamically switch to other parents if they experience a poor QoS. In [3], Chu et al. explore the possibility of video conferencing applications using an overlay multicast architecture. The constructed overlay spanning tree is optimized according to the measurement of available bandwidth and latency among users, and can be modified by the addition of good links and the dropping of poor links. The main goal of RON [4] is to enable a group of nodes to communicate with each other in the face of problems with the underlying Internet paths connecting them. RON detects problems by aggressively probing and monitoring the paths connecting its nodes. If the underlying Internet path is the best one, that path is used and no other RON node is involved in the forwarding path. If the Internet path is not the best one, the RON will forward the packet by way of other RON nodes.

Performance characteristics of a peer-based overlay network are likely to be very different and highly variable compared to the traditional Internet or even traditional managed overlay networks because packets may cross the Internet several times. However, the massive diversity, i.e. multiple peer-based overlay paths harnessed could compensate for the performance variability of any one path. In addition, lightweight support at intermediate nodes can improve the single path performance. In this paper, we focus on the latter problem and propose a novel "Overlay Multi-hop FEC" scheme for video streaming over peer-based overlay networks. The OM-FEC scheme partitions the end-to-end overlay path into segments according to the error characteristic of the overlay path, and provides error resilience over those segments. In this paper, we do not focus on overlay path construction and routing problems. We assume a fixed constructed peer-based overlay path and focus on how to efficiently utilize it. We will henceforth use the term "overlay path" to mean the constructed path over a peer-to-peer network.

A. Scope and Assumptions

Most of the prior work discussed above was focused on massive video data distribution or video conferencing using application layer *multicast* based on an overlay or peer-to-peer network. In contrast, our objective is to revisit the problem of efficiently utilizing the resources of a *single* overlay path. Our approach operates at small time-scales in the data-plane, and can be combined with overlay routing and topology management approaches that operate in the control-plane and in larger time-scales [4]. In this sense, our OM-FEC scheme is *complementary* to the prior work where resilience is provided using overlay routing methods. We assume that we can always construct an overlay path with higher bandwidth than default Internet route by using P2P techniques such as Chord [7] or Pastry [12], to obtain a set of intermediate forwarding nodes as shown in Figure 1. In the figure, the dashed lines represent the virtual link between overlay

nodes and the solid line represents the default Internet path. Quantities B_i, P_i, and RTT_i represent, respectively, the bandwidth, loss rate, and round trip time of the *i*-th *virtual* link.

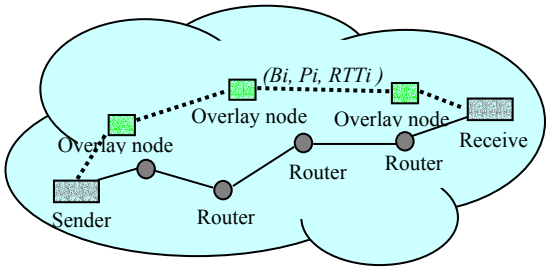

Figure 1: Streaming video using overlay network

B. Motivation

The advantages of application-layer overlay networks arise from two fundamental properties: (1) overlay nodes have capabilities of computation and storage power, which are far beyond basic store and forward operation, and (2) the overlay topology can be constructed and manipulated to suit one's purposes. Based on these considerations, we argue that applying forward error correction (FEC) purely end-to-end in an overlay network is a sub-optimal strategy. For example, in Table 1 (a), we list a set of possible bandwidth and loss rates in a 6-hop overlay path, where B_i and P_i are, respectively, the bandwidth and loss rate of the *i*-th virtual link. In this paper, FEC is provided by the use of Reed-Solomon (RS) erasure-correcting codes.

Hop	1	2	3	4	5	6
B_i	300K	380K	550K	400K	600K	800K
P_i	0%	4%	3%	3.5%	1.5%	2%

FEC Method	End-to-End	OM-FEC
Good-put	258K	300K
Path loss rate	14%	14%

Table 1: (a) Possible bandwidth and loss rate of an overlay path; (b) Good-put: OM-FEC vs End-to-End

In order to fully recover lost packets, the end-to-end based FEC scheme would have to design its FEC based on the end-to-end available bandwidth 300Kbps and the end-end loss rate, which in this case is approximately 14%. Thus, the good-put is reduced to 0.86×300 = 258Kbps. On the other hand, if a heavyweight hop-by-hop based FEC scheme is used, the end-to-end good-put can

be engineered to be 300Kbps, because hop 1 whose bandwidth of 300Kbps is the bottleneck in this overlay path, has 0% loss. Obviously, the hop-by-hop FEC scheme may induce more per-hop delay and use more computation power of the overlay nodes than necessary. To balance the delay and bandwidth efficiency with architectural complexity considerations, we propose a flexible and adaptive error resilience protocol called Overlay Multi-hop FEC (OM-FEC) for video streaming over peer-based overlay path. Our proposed scheme aims to maximize the video good-put over the overlay path and to minimize the overall computation complexity in the intermediate nodes. Specifically, our OM-FEC scheme does not require FEC encoding/decoding at each hop. Instead, OM-FEC scheme *optimally partitions* the whole overlay path into sub-paths and performs FEC over these sub-paths.

The rest of the paper is organized as follows. In Section II we describe our protocol, rate allocation scheme, and algorithms for our proposed novel OM-FEC strategy. Next, we describe the simulation, real Internet experiments and discuss the results in Section III. Finally, in Section IV we conclude our work and provide some ideas for possible extensions.

II. PROTOCOL OVERVIEW

Our proposed protocol operates in two modes: (1) pure end-to-end mode; and (2) OM-FEC mode. The default mode is the end-to-end mode. In this mode, FEC is designed based on the end-to-end network characteristics, and the overlay nodes simply receive and forward data and FEC packets to the destination. If the network experiences congestion such that the end-to-end FEC scheme cannot recover the lost packets, the protocol transitions to the OM-FEC mode.

The basic building blocks for the OM-FEC mode include an algorithm to determine optimal partitioning of the overlay path, a rate allocation algorithm for allocating appropriate FEC rate for different virtual links of an overlay path, and the actual deployment of the FEC on the path.

In the OM-FEC mode, video server sends out an active probe packet every Δt time units. Each overlay node measures the loss rate and round trip time (*RTT*) of its related virtual link in the overlay path using the probe packet. Thus obtained per-hop *RTT* and loss rate estimates are used to infer the TCP-friendly available bandwidth of each virtual link. Now, with this available bandwidth and loss rate, the optimal FEC rate for each hop can be calculated. However, FEC coding need not be implemented at each hop. The server runs a partitioning algorithm to calculate the optimal path partitioning consistent with the above FEC rate estimates, so that the overall computational complexity at intermediate hops is minimized without sacrificing the FEC-based resilience gains. Partitioning splits the overlay path into sub-paths, and FEC coding is employed over these sub-paths. Hence, only the boundary nodes between sub-paths are involved in FEC encoding/decoding. Overall, the final deployment of FEC in each time interval Δt would maximize the end-to-end realized good-put and minimize the computation complexity at intermediate peers. If the path partition algorithm produces a single sub-path (equivalent to the entire end-to-end overlay path), the system transitions back to the end-to-end mode. The decision made by the server is conveyed to every node by a command packet from server, so each node knows what it should do after it receives the command packet. The following sections will outline the details of the rate-allocation and path segmentation strategies in the OM-FEC scheme.

A. Rate Allocation Strategy in OM-FEC

The available bandwidth of the overlay path is allocated to both video data and FEC parity data. An adaptive end-to-end rate allocation scheme is described in our prior work [8]. In the OM-FEC mode, the problem of allocating optimal rate for FEC and video data to each virtual link can be stated as follows: find the FEC scheme of each hop that maximizes the video good-put

$$B_{data} = \min_{1 \le i \le N} \{B_{data}(i,t)\}$$

$$\text{subject to} \quad \begin{cases} 0 \le B_{FEC}(i,t) \le B_{req}(i,t), \\ B_{data}(i,t) \le B(i,t) - B_{FEC}(i,t), \end{cases}$$

where N is the total number of virtual links, $B_{FEC}(i, t)$ is the actual FEC bandwidth at i-th virtual link over a time interval $(t, t + \Delta t)$. $B_{data}(i, t)$ denotes the bandwidth assigned to video data on the i-th virtual link. $B_{req}(i, t)$ is the required FEC bandwidth for i-th virtual link. The required FEC bandwidth is the bandwidth with which the lost packets at receiver can be fully recovered by FEC. $B(i, t)$ is the estimated *TCP-friendly bandwidth* of the i-th virtual link and can be calculated using an equation from [5] as follows:

$$B(i,t) = \frac{S}{T_{RTT-i}\sqrt{\frac{2p(i,t)}{3}} + T_{rto-i}(3\sqrt{\frac{3p(i,t)}{8}})p(i,t)(1+32p(i,t)^2)}. \quad (1)$$

In equation (1), S is the packet size in bytes. $T_{RTT\text{-}i}$ is the estimated RTT of the i-th hop in seconds. $T_{rto\text{-}i}$ is the TCP timeout of i-th link, $p(i, t)$ is the estimated loss rate of the i-th link. The end-to-end bandwidth $B(t)$ from source to receiver is limited by the minimal per-hop TCP-friendly available bandwidth of the overlay path, i.e.,

$$B(t) = \min_{1 \le i \le N} \{B(i,t))\}. \quad (2)$$

The goal of the rate allocation scheme is to find the suitable rate allocation $B_{FEC}(i, t)$ for the constructed overlay path to maximize the good-put of the video data in case of network congestion. Since we do not assume any special structure of the video bit stream or any special packet scheduling scheme, the distortion of video quality is minimized when the amount of video data delivered is maximized. Our strategy proceeds as follows: for i-th virtual link, the algorithm assigns a portion of the available bandwidth $B(i, t)$ for video data, and the remaining bandwidth is assigned to FEC until either the desired FEC rate is met or the rest of the available bandwidth

budget is exhausted. In an extreme case, if $B(i, t) \leq B_{data}(i, t)$, all the available bandwidth is assigned to the video data.

B. Forward error correction

Forward error correction (FEC) codes are usually used for channel coding to protect the data from channel errors (e.g. packet losses, bit errors). In this paper, we use systematic Reed-Solomon erasure correcting codes as FEC. The RS (n, k) encoder takes k data packets and generates $n - k$ parity packets. Given the position of the lost packets, the RS decoder can reconstruct up to $n - k$ lost packets out of a total of n packets. Hence, larger ratio of n/k leads to higher level of protection for the original data. In a video steaming system, k cannot be chosen arbitrarily, since video data is time sensitive. Larger values of k imply longer delays in the receiver side. The value of k is related to the bit rate of the encoded video bit stream, packet size and buffering time at the receiver side, as illustrated next. Let the encoded bit rate be β bps, the packet size be η bytes, and the receiver buffer size be λ seconds. If the receiver buffer is fully buffered with video data, the total amount of bits in the buffer is $\lambda\beta$. The amount of bits in a network packet is 8η, so the total amount of packets in buffer is $\lambda\beta/8\eta$. If FEC is deployed over k packets, the receiver needs to wait for at least k packets to arrive prior to RS (n, k) decoding. If $k > \lambda\beta/8\eta$, the buffer is empty before FEC decode. Therefore, we need $k \leq \lambda\beta/8\eta$.

Using a systematic code, the encoder picks groups of k source data blocks to generate $n - k$ parity blocks. Every source data block is used $n - k$ times, so we can expect the encoding time to be a linear or approximately linear function of $n - k$. Since our system relies on real-time FEC encoding/decoding, it is necessary to evaluate the performance of the RS codec. We test our implementation of the RS codec (based on Phil Karn's RS codec [10]) on a Dell PC with Pentium 4 CPU at 2.0 GHz, with 256 MB RAM, running Linux RedHat 8.2, with $n = 255$, and k

variable. The time needed to encode k packets is shown in Table 2, for various values of $n - k$ and packet size.

$n - k$	5	10	15	20	25	30	35	40
256 Bytes/packet	1.1ms	1.9ms	2.2ms	3.3ms	3.9ms	4.3ms	4.9ms	5.4ms
512 Bytes/packet	2.0ms	3.7ms	4.3ms	6.5ms	7.8ms	8.6ms	9.8ms	10.8ms
1024 Bytes/packet	4.1ms	7.3ms	8.6ms	13.0ms	15.6ms	17.2ms	19.5ms	21.6ms

Table 2: The RS encoding time as a function of $n - k$ and packet size.

From Table 2, we observe that very high FEC encoding rates can be achieved even on commodity PCs (which are going to be the peers of a peer-to-peer overlay network). For example, the encoding bit rate of RS (255,245) code can be up to 274 Mbps at packet size 1024 bytes, and this code can recover the lost packets at random loss rate up to 3.92%. Erasure codes tested in [11] gave similar results. Since the decoding process is much faster than encoding, we do not list our decoding test results here.

C. Overlay Multi-hop FEC (OM-FEC)

Based on the estimated parameters of the constructed overlay path, the server runs OM-FEC algorithm to decide what kind of FEC scheme should be added to protect the video data, and also which overlay node should perform the FEC encoding/decoding and how much FEC should be added at the chosen nodes. The criteria for choosing FEC scheme for the overlay network are: (1) maximize the good-put of the constructed overlay path, and (2) minimize the overlay nodes computation complexity. In order to maximize the good-put of the constructed overlay network, we use the rate allocation algorithm described in Section II.A. To minimize the computation burden of the overlay nodes, the OM-FEC scheme should use as few nodes as possible for the FEC encoding/decoding. Fewer nodes involved in the FEC encoding/decoding results in smaller jitter and transmission delay at receiver side. Our OM-FEC algorithm works as follows.

(a) The server first calculates the parameters for the end-to-end RS (n, k) code based on a given target loss probability. According to the results presented in [9], the viewing quality of MPEG-4 encoded video is acceptable at a loss rate of 10^{-5}, and good at a loss rate of 10^{-6}. In this paper, we choose the target loss probability $P_{target} \leq 10^{-6}$;

$$P_{target} = \sum_{i=n-k+1}^{n} \binom{n}{i} P^i (1-P)^{n-i}. \quad (3)$$

The end-to-end loss rate is approximately estimated as $P \approx 1 - \prod_{i=0}^{i=N} (1-P_i)$. Given a target loss rate P_{target} and the loss rate of P, if k is fixed, n can be calculated, and vice versa. The total bandwidth B_{total} needed for transmitting both data and parity packets is determined as

$$B_{total} = B_{data} + B_{FEC} = B_{data} + \frac{(n-k)}{k} B_{data} = \frac{n}{k} B_{data}. \quad (4)$$

(b) If $B_{total} \leq B$, this means the bandwidth needed for original video data and FEC is smaller than the available end-to-end bandwidth B of the overlay path, then FEC is added end-to-end. No intermediate overlay node is involved into the FEC encoding/decoding. The operating mode is therefore the end-to-end mode.

(c) If $B_{total} > B$, the available end-to-end bandwidth B is not large enough for both the original video data and end-to-end FEC overhead. If the current mode is end-to-end mode, then the protocol transitions to the OM-FEC mode. In order to reduce the computation burden, the OM-FEC scheme partitions the overlay path into sub-paths according to the characteristics of each virtual link as shown in Figure 2. For example, the OM-FEC algorithm partitions the overlay path into X sub-paths, which are the first J nodes as sub-path1, the next L nodes as sub-path2 and the last M nodes as sub-path$X$, respectively. FEC scheme is deployed over each sub-path. Thus, the overall computational burden is reduced compared to the hop-by-hop FEC computation.

Parameters *J, L, ..., M* are dynamically determined by the OM-FEC algorithm as explained below.

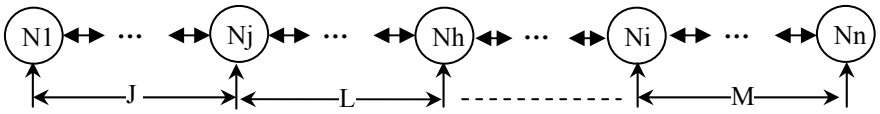

Figure 2: Overlay path is partition into segments (sub-paths)

The algorithm of partitioning the overlay path into sub-paths is described as follows.

Start = 0*;* *//begin calculation from the server*

For (i = 1*; i <= N; i++){*

// calculate the FEC bandwidth which should be allocated to the path from start node to i-th node

Calculate $B_{FEC,(start—i)}$*);*

// calculate the FEC bandwidth should be allocated to the path from start node to (*i*+1)*-th node*

Calculate $B_{FEC,(start—(i+1))}$*;*

// find the boundary node to partition the overlay into sub-paths

If $((\min\{B_{start}, \ldots, B_i\} \geq (B_{data} + B_{FEC,(start—i)}))$ && $(\min\{B_{start}, \ldots, B_{(i+1)}\} < (B_{data} + B_{FEC,(start—(i+1))})))${

From start node to i-th node is partitioned as one sub-path;

FEC is deployed over this sub-path, the FEC bandwidth allocated to this path is $B_{FEC,(start—i)}$*;*

Start = i ; // start from the i-th node to partition the rest of the path, the i-th node is boundary node

}

}

In the above pseudo code, B_i is the estimated available bandwidth of *i*-th virtual link; B_{data} is the required bandwidth for video data. *N* is the total number of links. $B_{FEC(start—i)}$ is calculated based on Section II.A and equations (1)–(4), but considering the segment from "*start*" node to *i*-th node as one virtual calculation link and using the cumulated loss rate and available bandwidth of this virtual calculation link. The server runs the above algorithm to partition the overlay path into sub-paths and deploys different FEC over different sub-paths. The decision is conveyed to all

intermediate nodes by a command packet. For each boundary node, the command packet contains a 3-byte field specifying the node ID, and the *n* and *k* parameters of the RS (*n*, *k*) codes. The nodes whose ID is not listed in the command packet will simply forward all the packets they receive, without FEC coding/decoding. Thus, each node knows what it should do after it receives the command packet. The boundary nodes of these sub-paths are the only ones involved in the FEC encoding and decoding. Based on the OM-FEC strategy, the largest sub-path could include all the nodes of the overlay path (same as the end-to-end scheme), and the smallest sub-path could be one hop (i.e. hop-by-hop). In other words, OM-FEC is an adaptive strategy that tunes the architectural complexity between the extremes of end-to-end and hop-by-hop operation.

D. Probe packet

An active probing method is used to calculate the round trip time (*RTT*) and loss rate of each virtual link. In order to synchronize overlay parameters calculation and reduce the bandwidth overhead, the sender uses a small active probing packet to synchronize the estimation procedure. The probe packet is sent from server every time interval Δt. Each overlay node processes the probe packet, and calculates the loss rate and the round trip time.

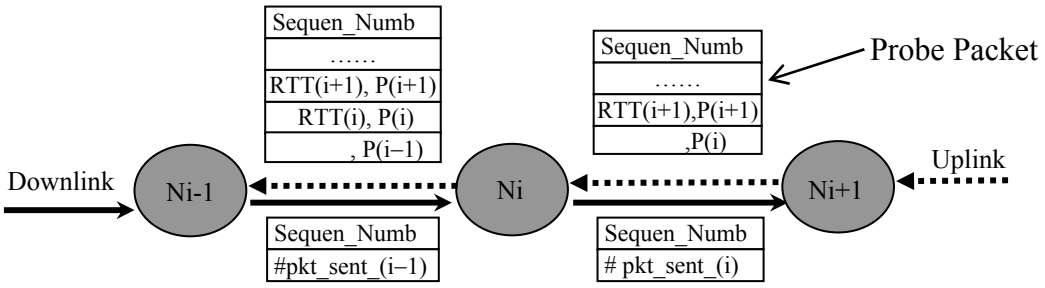

Figure 3: Probe packet process and the information

The structure of the probe packet is shown in Figure 3. *Sequen_Numb* is the sequence number of the probe packet, *#pkt_sent*(*i*) denotes the number of packets the *i*-th node sent. Parameters {*RTT, P*} denote round trip time and loss rate, respectively. The probe packet goes through all

nodes along the constructed overlay path. For downlink path parameter estimation, each node caches the probe packet from its up node, replaces item *"#pkt_sent_*(i–1)*"* with its own *"#pkt_sen_*(i)*"* and then, forwards the probe packet to the next node. Each node records the time T_{send} - the instant when it sends the probe packet to its down node. This parameter is later used for round trip time calculation. With the received item *"#pkt_sent_*(i–1)*"* and the measured received data packets *"#pkt_recvd_from_*(i–1),*"* the i-th node can calculate the loss rate of the (i–1)-th link as $P_{i-1} = \frac{\#\, pkt\_sent(i-1)}{\#\, pkt\_recvd\_from(i-1)}$. The probe packet is fed back to the server after it reaches the receiver. The feedback packet collects all information from these overlay nodes while going back to server. As for round trip time estimation, as soon as a probe packet arrives at the i-th node from the (i+*1*)-th node, the i-th node obtains the arrival time of this packet T_{arrive}, and then calculates the round trip time of i-th link as follows:

$$RTT_i = T_{arrive} - T_{send} - \sum_{j=i}^{j=n} RTT_j \tag{5}$$

The i-th node attaches the calculated loss rate of (i+*1*)-th link and the round trip time of the i-th link to the probe packet and then forwards the packet to (i–1)-th node, until it arrives the server. The server analyzes the information brought back by the probe packet and calculates the available bandwidth for each virtual link.

III. RESULTS

We now demonstrate the effectiveness of our OM-FEC scheme by comparing it with the end-to-end based FEC scheme in both simulations and real Internet experiments. Based on our algorithm we expect that, as the number of virtual links increases and the variation of the loss

rate becomes larger, our approach would outperform the end-to-end FEC scheme. This expectation is confirmed by our simulations and experiments. In this section, all the curves are the averages of at least ten runs of simulations or experiments.

A. Matlab simulations

In the Matlab simulations described in this section we compare our OM-FEC scheme and the end-to-end scheme in terms of the video good-put they provide. We assume the task is to transmit a video encoded at 512 kbps (which also represents the highest possible video good-put) through a sequence of overlay nodes. The simulation configuration is shown in Figure 4. The topology includes one sender, one receiver and three intermediate overlay nodes. L1 through L4 represent four overlay virtual links. Similarly to [13] and [14], we use a 2-state Markov model (Gilbert model) to simulate each virtual link. The sender sends out video packets through the three overlay nodes to the receiver, and the feedback information is sent back using the same nodes but in reverse direction. Probing packet is sent from the server once every second.

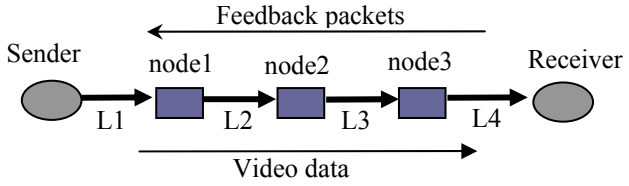

Figure 4: Simulation configuration

We begin our simulation by starting the network in a state of slight congestion. The simulation parameters are shown in Table 3. The average burst lengths were in the range of 2-3 packets.

Test	Basic Test	Test A	Test B
L1	lossrate = [1% to 2%],RTT=10ms	lossrate = [1% to 2%]	Lossrate = [2% to 3%]
L2	lossrate = [1% to 4%],RTT=30ms	lossrate = [3% to 5%]	Lossrate = [3% to 6%]
L3	lossrate = [3% to 5%],RTT=10ms	lossrate = [3% to 5%]	Lossrate = [3% to 6%]
L4	lossrate = [2% to 4%],RTT=20ms	lossrate = [2% to 4%]	Lossrate = [3% to 4%]
RS(n, k)	k = 80, n is variable		
Network	conditions change every second		
Video	Encoded video bitrate = 512 kbps		

Table 3: Simulation parameters

We set the round trip time and range of packet loss rates for each virtual link. The network condition is changed every second. At time t = 0s, the sender begins to send out video data to receiver. The gathered network information by probe packet from each hop is fed back to the sender. For the basic test, the sender calculates the available bandwidth of each link as shown in Figure 6, according to the measured loss rate (Figure 5) and round trip time of each virtual link. The sender determines what kind of FEC scheme should be deployed for the current network condition. Either end-to-end scheme or OM-FEC scheme is chosen. The end-to-end scheme deploys FEC according to the available bandwidth and end-to-end loss rate. Our scheme fine grains the overlay path into sub-paths and deploys the FEC according to the characteristics of the overlay path and network conditions. Thus, OM-FEC scheme can use bandwidth more efficiently in case of network congestion as shown in Figure 7. In Figure 7, the video good-put is defined as bandwidth occupied by the video data. To test our approach in a heavier congestion condition, we increase the loss rate of several links in the simulation setup shown in Table 3, Test A and Test B. In Figure 8, we can see that our scheme outperforms the end-to-end scheme by a larger margin at severe congestion. For the end-to-end scheme, the increased end-to-end loss rate results in more FEC overhead over the entire overlay path. On the other hand, our OM-FEC scheme only considers the related sub-paths where loss rate increases. Thus, OM-FEC scheme has better performance than the end-to-end scheme at severe congestion.

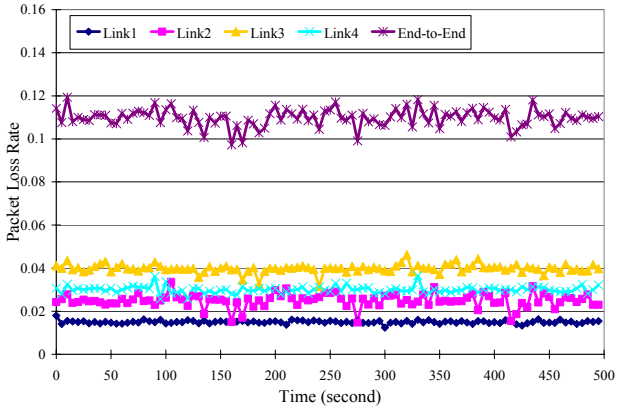

Figure 5: The packet loss rate of the overlay path

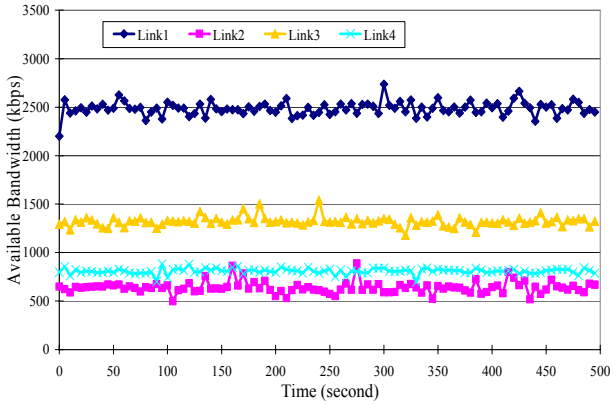

Figure 6: The available bandwidth of each virtual link

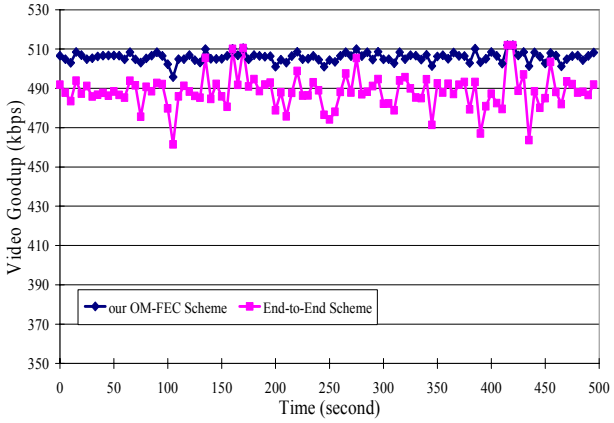

Figure 7: The video good put of our scheme vs. end to end scheme

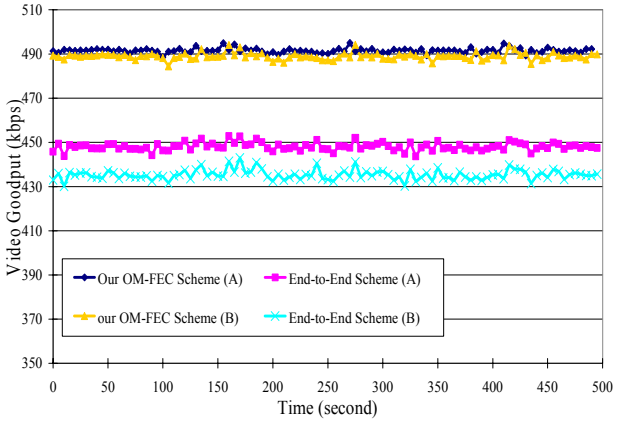

Figure 8: The video good put of our scheme vs. end to end scheme in test condition A and B

B. Real-world Internet experiments

We also implemented our protocol over the real Internet using the Planet-Lab infrastructure [6]. The implementation includes an overlay agent and the protocol itself. Our overlay agent can run at any Linux Planet-Lab node. The agent forwards video packet to the next node until it arrives at the destination. The experimental topology is the same as Figure 3 and the Planet-Lab nodes involved are listed in Table 4.

Server	nima.eecs.berkeley.edu
Node1	planetlab1.flux.utah.edu
Node2	planetlab-1.cmcl.cs.cmu.edu
Node3	planetlab1.cs.cornell.edu
Receiver	video.testbed.ecse.rpi.edu

Table 4: Nodes involved in our experiments

In the experiments described in this section we measure the objective video quality at the receiver in terms of the Peak Signal-to-Noise Ratio (PSNR). The video sequence used in the experiments is the *Foreman* sequence, QCIF resolution, at 30 frames per second. The video bit-stream is encoded using H.263+ encoder with error-resilient option at 512 kbps, Intra frame refresh at every second. Since there is virtually no congestion from UC Berkeley to RPI, packets are artificially dropped to simulate the congestion effect. The packet loss rate from Utah to CMU is set to 5%, other links are set to 1%. The available bandwidth from Utah to CMU is also upper bounded to 550 kbps. Under these conditions, the end-to-end scheme designs a FEC based on the 550 kbps bandwidth and total loss rate 8%. Our OM-FEC scheme identifies the bottleneck and partitions the overlay into three sub-paths as follows: sub_path1 from Server to Node 1, sub_path2 from Node 1 to Node 2, and sub_path3 from Node2 to the receiver. Two nodes are involved in FEC encoding/decoding. The FEC is deployed within each sub-path. The OM-FEC designs FEC at the bottleneck for a bandwidth of 550 kbps and 5% loss rate. It can recover more packet loss than the end-to-end scheme, so its video quality is much higher than that provided by the end-end scheme, as shown in Figure 9. The PSNR gains are on the order of 13 dB.

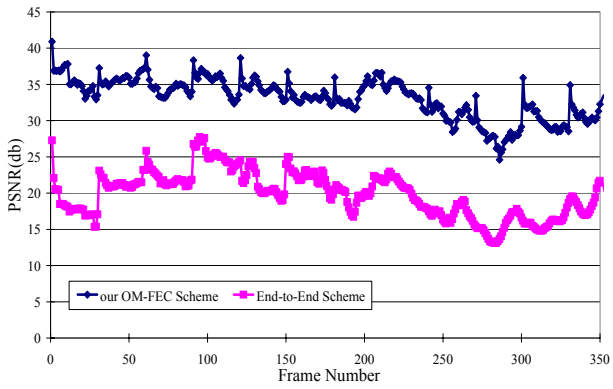

Figure 9: Video PSNR of our OM-FEC scheme vs End-to-End Scheme (4 virtual links)

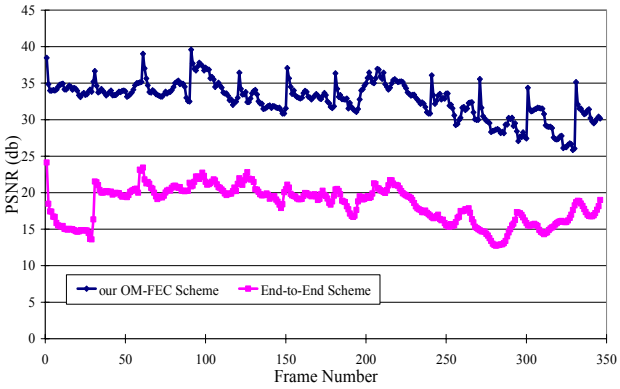

Figure 10: Video PSNR of our OM-FEC scheme vs End-to-End Scheme (5 virtual links)

In the second set of experiments we add one overlay node (Node 4: planet1.ecse.rpi.edu) to the path at last hop with 1% loss rate. In this case, for the end-to-end scheme, the FEC is designed based on the bandwidth of 550 kbps and loss rate 9%. Our OM-FEC scheme still partitions the overlay path into sub-paths as before, and the FEC at the bottleneck is still designed for the bandwidth of 550 kbps and 5% loss rate. Still, two nodes are involved in FEC encoding/decoding. The PSNR results are shown in Figure 10, which shows that the advantage of our OM-FEC scheme over the end-to-end scheme is increased compared to Figure 9. Here, the PSNR gains are on the order of 14 dB. As the number of nodes involved in the transmission increases, our OM-FEC scheme performs dramatically better than the end-to-end scheme. For visual comparison, we show a few decoded frames in Figures 11 and 12.

Figure 11: Video streaming over 4 virtual links: OM-FEC (left) vs. end-to-end scheme (right).

Figure 12: Video streaming over 5 virtual links: OM-FEC (left) vs end-to-end scheme (right)

IV. Conclusions and Future Work

In this paper, we have proposed a novel Overlay Multi-hop FEC scheme for streaming video over peer-to-peer networks, which automatically adapts its architectural complexity between the extremes of pure end-to-end or pure hop-by-hop operation. The proposed OM-FEC scheme improves the good-put of the constructed peer-based overlay transmission path by dividing the overlay path into segments based on link characteristics, and applying the appropriate amount of FEC over each segment. We have shown that video streaming using our approach outperforms the end-to-end FEC approach without incurring high per-hop complexity. In future work, we will incorporate ARQ and multi-path routing, as well as techniques for dealing with node failures [4], in an effort to build up an overall network service abstraction for video streaming and conferencing over peer-to-peer networks.

References

[1] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking," Microsoft Technical Report MSR-TR-202-37, April 2002

[2] K. Yeo, B. S. Lee, and M. H. Er, "A Peering Architecture for Ubiquitous IP Multicast Streaming," *ACM SIGOPS Operating Systems Review*, vol. 36, no. 3, pp 82-95, July 2002.

[3] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture," *Proc. SIGCOMM'01*, San Diego, CA, August 2001.

[4] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," *Proc. ACM SOSP'01*, pp. 131-145, Banff, Canada, October 2001.
[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," ICSI Technical Report TR-00-03, March 2000.
[6] http://www.planet-lab.org/
[7] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications,". *Proc. SIGCOMM'01,* pp 149-160, San Diego, CA, August 2001.
[8] Y. Shan and A. Zakhor "Cross Layer Techniques for Adaptive Video Streaming Over Wireless Networks," *Proc. IEEE ICME'02*, pp. 277-280. Lausanne, Switzerland, August 2002.
[9] S. Gringeri, R. Egorov, K. Shuaib, A. Lewis, and B. Basch, "Robust compression and transmission of MPEG-4 video," *ACM MM 2000 Electronic Proceedings*, June 2000.
[10] Phil Karn, "General-purpose Reed-Solomon encoder/decoder in C," Version 4.0, available at http://www.ka9q.net/code/fec
[11] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Computer Communication Review*, vol. 27, pp.24-36, April 1997.
[12] A.Rowstron and P.Druschel "Pastry: Scalable, distributed object location and routing for large-scale peer to peer system," In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelburg, Germany, 2001.
[13] J. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive fec-based error control for internet telephony," in *Proceedings of IEEE INFOCOM*, 1999.
[14] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1012-1032, June 2000.