A Report to ooVoo LLC Evaluation of loss recovery schemes for layered video in video conferencing applications: retransmission and FEC

Yang Xu, Chenguang Yu, Yong Liu and Yao Wang
Polytechnic Institute of New York University
2012-09-20

I. Introduction

In this document, we introduce the results of our simulation to compare the recovery effect of using retransmission and using Forward Error Correction (FEC).

II. Simulator Description

In the simulator, we don't transmit packets in the real Internet environment. Instead, simulation is based on theoretical numerical calculation. Like Fig. 1 shows, our simulator consists of three parts: sender side, network side and receiver side.

In sender side, whenever video packets belonging to the intended layers have been generated, we will insert those packets to a FIFO queue with infinite size – the sending buffer queue in Fig. 1. In network side, we use a FIFO queue with finite size to simulate the congestion queue in Internet. To simulate propagation delay and packet loss, after popping out of network queue, if loss happens, that packet wouldn't be delivered to receiver side. Otherwise, after one-way propagation delay time, that packet will be received by receiver side. In receiver side, received packet will be put into receiver buffer and one ACK indicating having received such packet will be received by sender after one-way propagation delay time.

At any given time, the total packets running in the network couldn't exceed the number that sending window size W specifies. Sending window size $W = \frac{R*RTT}{S}$, where RTT is the round-trip-time including congestion delay and propagation delay, R means the network bandwidth, S means the packet size.

To simulate the decoding process, we employs an identical constant decoding deadline gap value for each frame. It means that in receiver side, receiver would try to decode the frame after that frame has been produced in sender side for a constant decoding deadline gap time. If at that time, all packets in that frame have been received, that frame could be decoded. Otherwise, that frame couldn't be decoded.

III. Simulator Settings

We use SVC Reference Software (JSVM) to generate packet trace. hierarchical P architecture is employed. Four frames form a GOP and the decoding dependency in one GOP is shown in Fig. 2. We only simulate temporal scalability: in one GOP, first frame belongs to layer 0, third frame belongs to layer 1, second and fourth frames form layer 2. In the simulation, one I frame will emerge every 32 frames and the other frames are all P frame. We use JSVM to generate totally 300 frames, which represents 10 second video when FPS is 30. We repeat this trace 60 times to simulate a 10 minute video conference. Table I lists FPS and its corresponding bitrate when different layers are transmitted.

If size of one frame is larger than 1400 bytes, we will divide that frame into multiple packets: sizes of packets except the last one are all 1400 bytes. The last packet only store the remaining content. In Fig. 3,

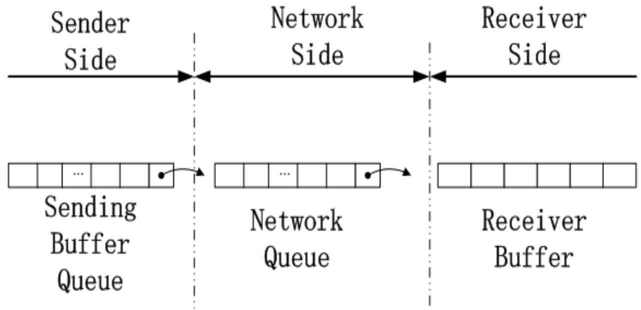


Fig. 1: Main Modules of Simulator

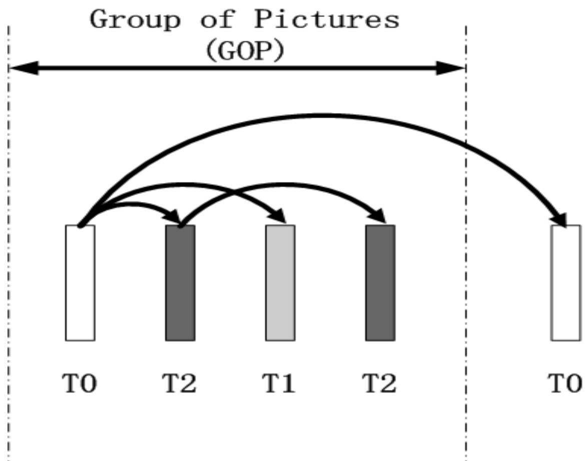


Fig. 2: Decoding Dependency in one GOP

we plot video rate for one GOP (four frames) in the trace that JSVM generates. From that Figure, we could see that video rate is not smooth. In Fig. 4, we plot the packet numbers that each frame would be splitted into. The nine largest peaks represent I frames, which will occupy $11-15$ packets. For P frames in Layer 0, they would occupy $2-5$ packets. For the frames in Layer 1 and Layer 2, mostly they only occupy 1 packet.

We don't add background competition flow in the simulation. Network bandwidth R is set to be 1000 kbps, which is larger than average video bitrate and smaller than video peak rate. Round-trip propagation delay RTT_{prop} is 20 ms. For the congestion delay, we consider a conservative value – the congestion delay that one packet (1400 bytes) is transmitted in that network. Thus, the sending window size is set to be 3. We set network queue size to be 10. Since sending window size is only 3 and no other flow is induced in the simulator, no congestion loss would occur. The decoding deadline is set to be 300 ms. Since encoding delay and decoding delay will cost about another 300 ms, this decoding deadline could

TABLE I: FPS and Bitrate

FPS	Bitrate (kbps)	Layer Indexes involved
7.5	295.05	Layer 0
15.0	383.86	Layer 0, Layer 1
30.0	512.11	Layer 0, Layer 1, Layer 2

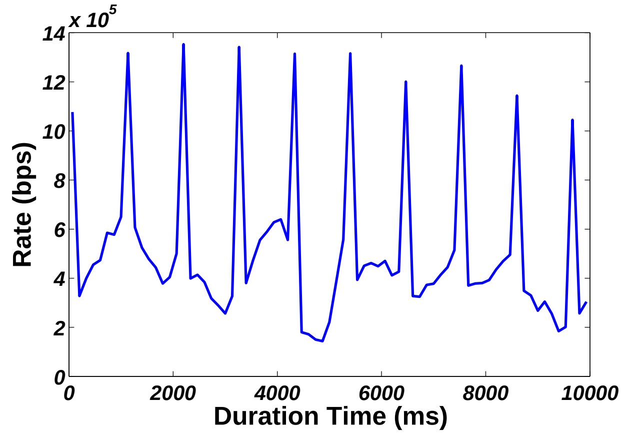


Fig. 3: Video Bitrate

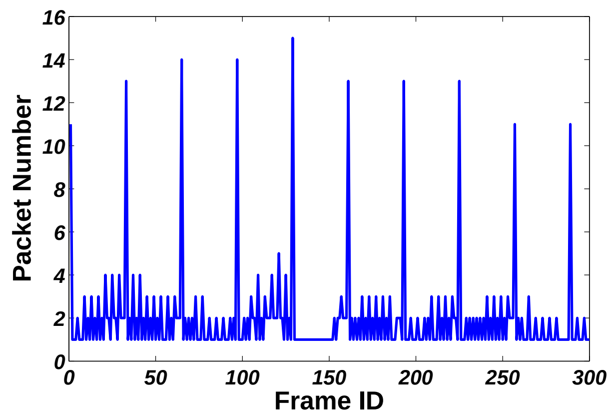


Fig. 4: Packet Number For Each Frame

guarantee that one-way display delay is smaller than 600 ms.

IV. Algorithm Description

We implement two solutions to do recovery: using retransmission and using forward error correction (FEC).

The algorithm for using retransmission is listed like below:

1) *We set a timeout value for each packet, if one packet has been sent out for that value time and the ACK of that packet hasn't been received by sender, timeout event happens. Ideally, timeout value* RTO *should be close to round-trip-time* RTT*. In the simulation, first we set a large* RTO *value (60ms), which is larger than* RTT*. For the* N*-th time we get one new* RTT *value (through ACK),* RTO *could be updated through* $RTO = \frac{RTO*N+RTT}{N+1}$*. Through this way,* RTO *would be closer to* RTT*.*
2) *In addition to timeout mechanism, we also use received ACK order to conjecture packet loss. We will store the order of sending packet ID. If the IDs of received ACK packets are out of order, we can infer that loss has happened before timeout event. Then, we can save some waiting time to do retransmission immediately.*
3) *In transmission, if packets don't belong to layer 0 and we find that those packets couldn't be delivered to receiver before their decoding deadline, sender wouldn't put those packets into network queue.*
4) *When we find loss happens, we only retransmit those loss packets belonging to layer 0. If sliding sending window size is not zero, we will put the loss packet into network queue. Otherwise, we put*

loss packet into the front position of sending buffer queue. Also, like the rule above, we will give up retransmit layer 0 packet when I-frame after this packet has been transmitted.

When simulating the effect of FEC, we don't consider the specific FEC solution. Instead, we just add some redundancy packets to the original packets. We sort frames into three different classes like Table II shows. We only add redundancy to layer 0 frames and the redundancy ratios for I-frame and P-frame are different. We will put original packets and their redundancy packets all into the network. For one specific frame, if received packet number is equal or larger than the original packet number, then that frame could be decoded. For example, if original layer 0 frame contains 5 packets and that frame is an I frame. Then adding 5 redundancy packets, 10 packets will be transmitted. In the receiver side, if 5 or more packets are received, we could decode that frame. And if we know that we couldn't deliver one frame before its decoding deadline, we will give up transmitting that frame like Rule 3 in the above retransmission algorithm shows. Fig. 5 shows transmission rate when using FEC approach in Table II. Compared to Fig. 3, difference between peak rate and lowest rate is much larger and average rate under such condition is 729.13 kbps.

TABLE II: Different Frame Classes and their Redundancy Ratios

Class	Frame Description	Redundancy Ratio
1	Layer 0 and I-frame	1.0
2	Layer 0 and P-frame	0.5
3	Layer 1, Layer 2	0.0

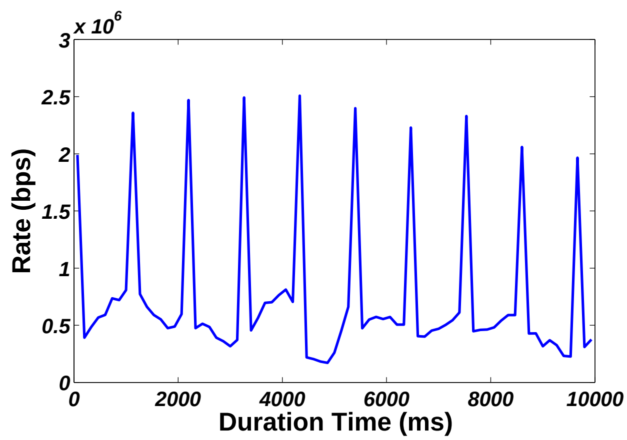


Fig. 5: Transmission Rate under FEC approach

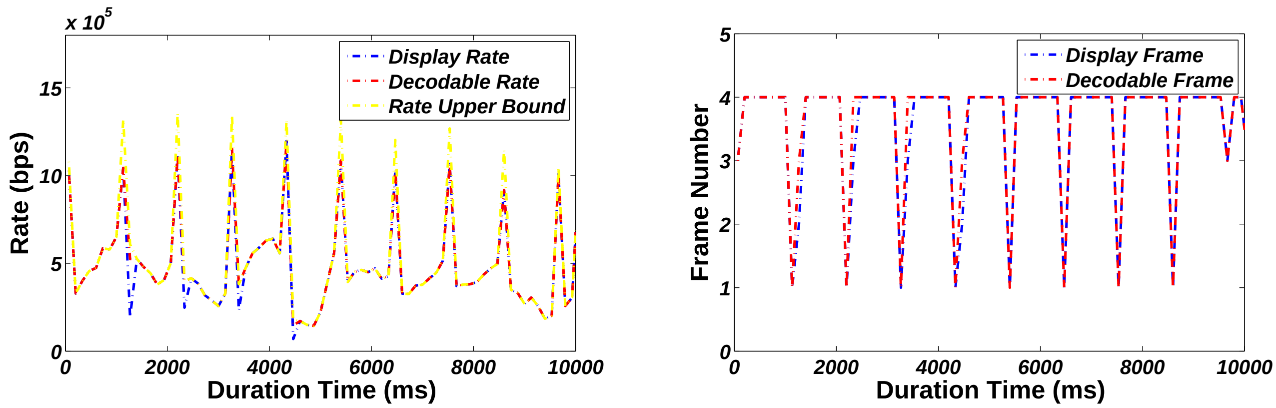
V. Simulation Results

In this section, we will show our simulation results under different network conditions (under loss variation). And we focus on two parameters: FPS and video rate. For each parameter, we will show the result in the form of **a/b**. Here **a** means the value got strictly before the deadline and **b** means the value finally got by receiver regardless of deadline. Although some packets may miss their decoding deadline, they may also help the decoding process. That's the reason why we show the value of **b**. We call the value of **a** as **Display FPS(Rate)** and the value of **b** as **Decodable FPS(Rate)** throughout the following sections.

A. No Loss

Here we consider the case that no packet loss is induced into network. Simulation shows that when using retransmission approach, video rate is 512.11/512.11 kbps and FPS is 30/30, which means that all packets have been transmitted to receiver before their decoding deadline.

Correspondingly, video rate is 477.79/487.25 kbps and FPS is 26.80/27.20 when using FEC approach. Although average transmission rate is still below network bandwidth, since here we have delay deadline constraint and transmission rate is not smooth as Fig. 5 shows, we have to give up transmitting some frames. Fig. 6 shows rate and frame number conditions when using FEC approach under no packet loss. Fig.6(b) shows display/decodable frame number in one GOP and the frame number upperbound under anytime is 4. From Fig. 6(a), we could find that transmitting large frame will cause problem. And large frame also affects the transmission of following frame, like the cases when time is about 1100 ms, 2200 ms, etc. That's the problem of using static FEC approach, introducing too much redundancies and causing rate varying too large when employing different protection priorities for different frames.



(a) Rate Distribution

(b) Frame Number in One GOP Distribution

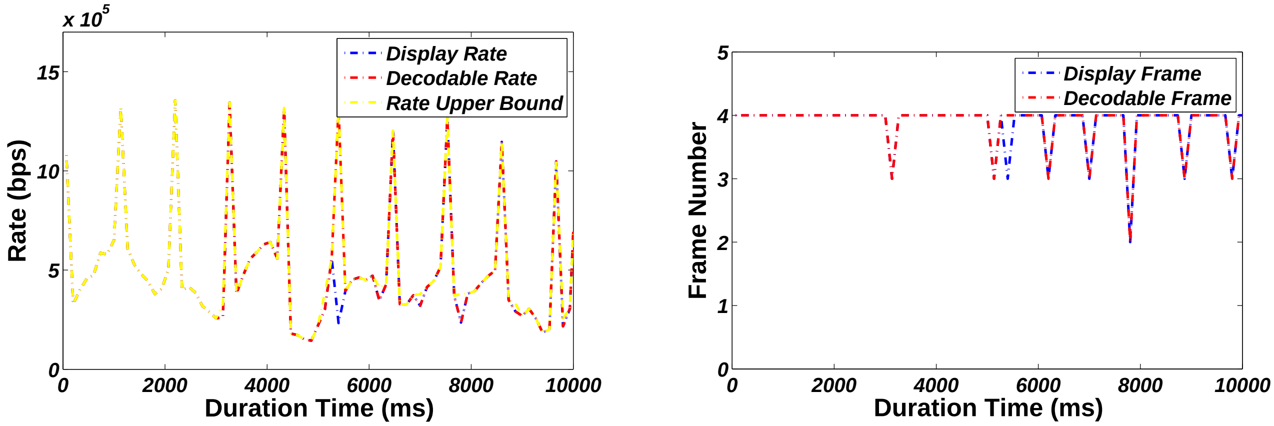
Fig. 6: FEC approach under No Packet Loss

B. Random Loss

We consider two random packet loss conditions: small packet loss and large packet loss.

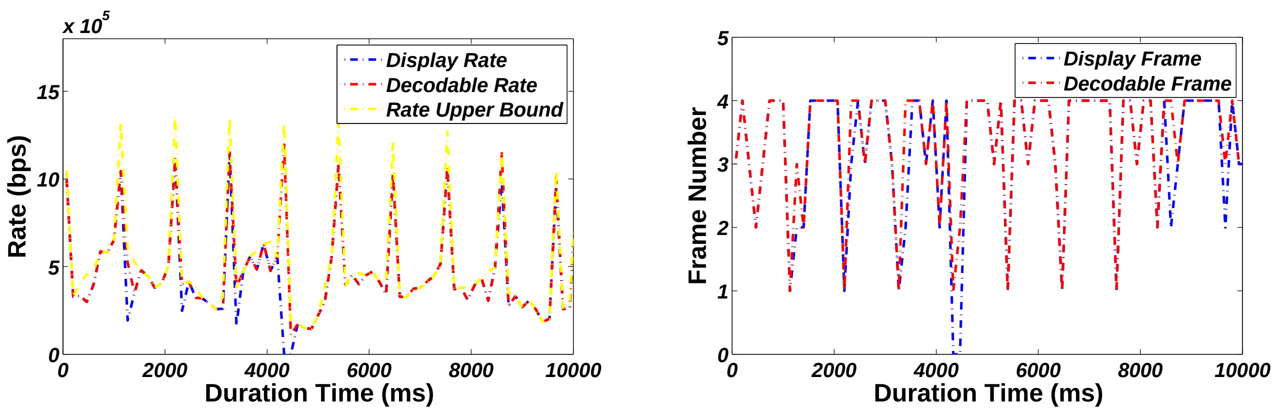
First, we set random packet loss ratio to be 5% to simulate small random packet loss situation. Under such scenario, when using retransmission approach, video rate is 488.83/504.16 kbps and FPS is 29.04/29.15. We could see that retransmission approach could recover most of random packet loss. And we just need to do retransmission once to do recovery. The worst condition happens around 5200 ms in Fig. 7(a). That is the case when loss happens in I-frame, retransmission is hard to recover that frame before its decoding deadline. Because that frame is too large. To the contrary, When using FEC approach, video rate is 433.59/452.80 kbps and FPS is 23.37/24.28, which are all worse than retransmission approach. Because here we only add redundancy to layer 0 frames. Thus, when loss happens to other frames, we couldn't decode those frames at all. So its recovery effect is not good.

Then, we set random packet loss ratio to be 30 % to simulate large random packet loss situation. When using retransmission approach, video rate is 196.48/375.50 kbps and FPS is 14.58/17.38. And when using FEC approach, video rate is 104.60/212.35 kbps and FPS is 4.78/6.71. We show the rate and frame number distributions for these two approaches in Fig. 9 and Fig. 10. It is easy to see that retransmission approach behaves much better than FEC approach. We can get an acceptable video by using retransmission approach. The problem for FEC approach is that it can only protect layer 0 frames. But even for layer 0 frames, we could hardly decode I frames before their decoding deadline because their lengths are too large. Thus, we observe a large difference between its display rate and decodable rate. For retransmission approach, also there is a big difference between these two kind of rates. That's also because of the existence of large I frames.



(a) Rate Distribution (b) Frame Number in One GOP Distribution

Fig. 7: Retransmission approach under 5% Random Packet Loss



(a) Rate Distribution (b) Frame Number in One GOP Distribution

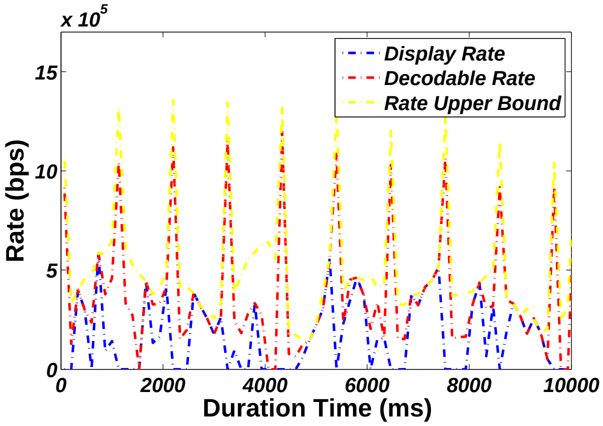
Fig. 8: FEC approach under 5% Random Packet Loss

C. Bursty Loss

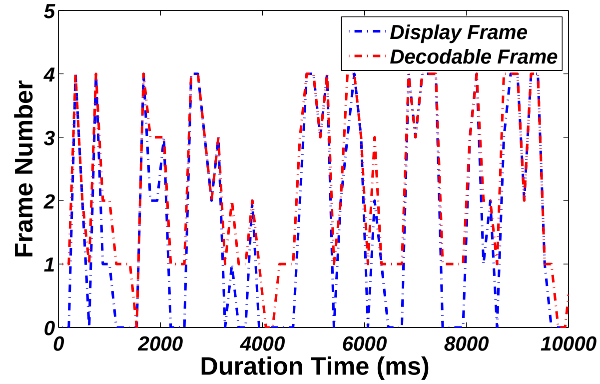
Packet losses on a drop-tail link can be bursty if the traffic it carries is bursty. The Gilbert model is most commonly used to describe bursty losses often experienced by end users in the Internet. As shown in Fig. 11, a Gilbert loss model has two parameters, unconditional and conditional loss probability, denoted as p_u and p_c. The conditional loss probability p_c captures loss burstiness: the larger p_c, the more bursty loss is. p and q are transition probabilities between non-loss and loss state. They can be computed as:

$$q = 1 - p_c,$$
$$p = \frac{p_u(1 - p_c)}{1 - p_u}.$$

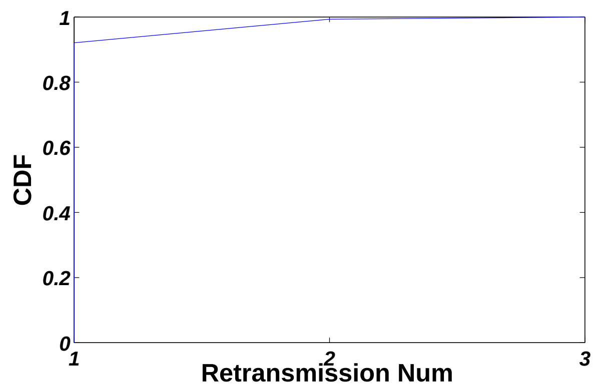
We set $p_u = 0.1, p_c = 0.6$ $(q = 0.4, p = 0.04)$ to simulate bursty loss condition. When using retransmission approach, video rate is 422.25/482.05 kbps and FPS is 26.31/26.93. We could see that retransmission approach is very efficient to recover bursty packet loss. The only problem happens when doing recovery to I frames, like 1000 ms, 2100 ms, 6400ms, etc in Fig. 12(a). To the contrary, FEC approach behaves much worse as Fig. 13 shows. At this time, video rate is 287.94/341.42 kbps and FPS is 14.82/16.17.



(a) Rate Distribution

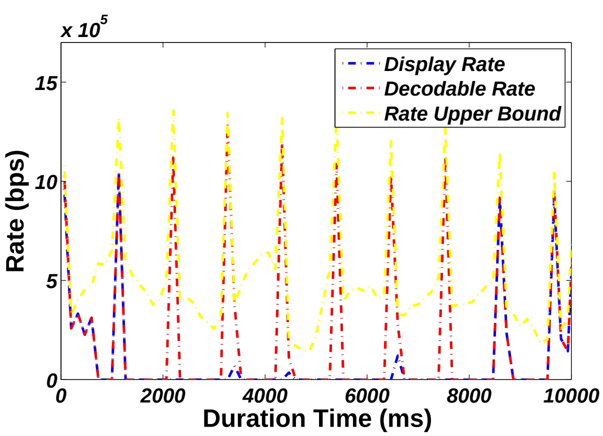


(b) Frame Number in One GOP Distribution

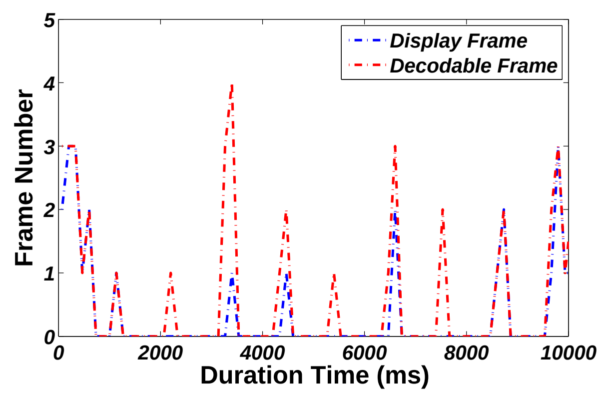


(c) Retransmission Number Distribution

Fig. 9: Retransmission approach under 30% Random Packet Loss



(a) Rate Distribution



(b) Frame Number in One GOP Distribution

Fig. 10: FEC approach under 30% Random Packet Loss

VI. Conclusion

In this document, we compare our retransmission approach with a FEC approach under three conditions: No Loss, Random Loss, Bursty Loss. Our FEC approach is kind of simple. However, we could see the weakness of FEC protection from our simple simulation: FEC need to predict loss condition. To protect loss, it will add a lot of redundancy. Since a lot of frames are very small (layer 1, layer 2 frames), we could only add FEC to more important frames (layer 0). Otherwise, we will add even much more

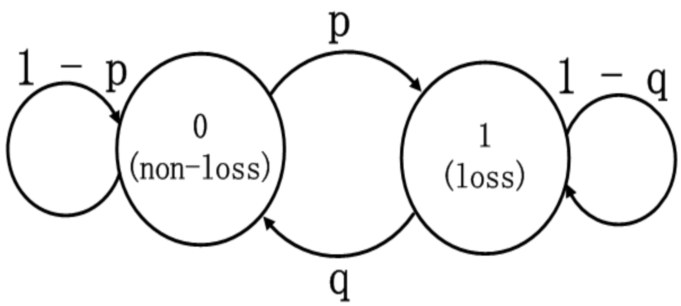
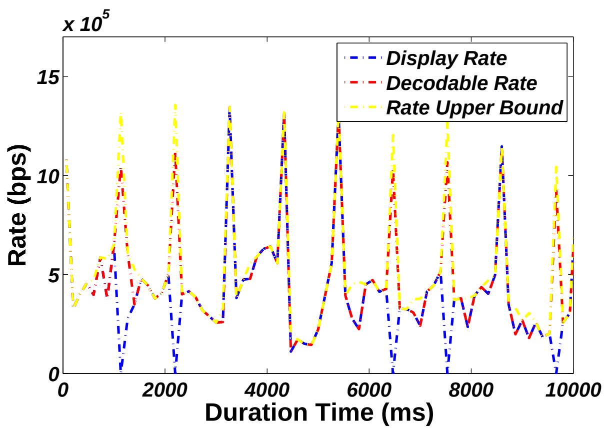
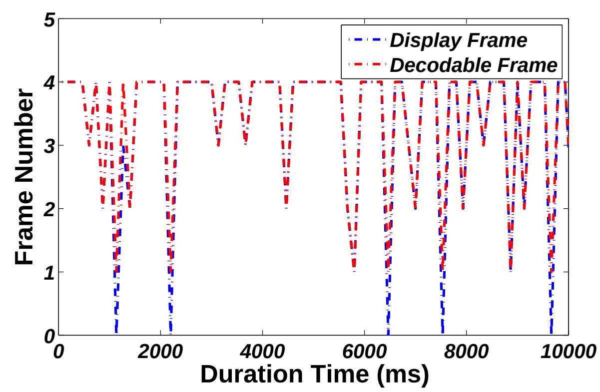


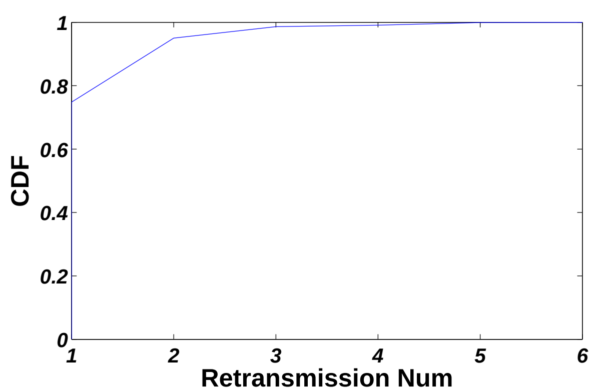
Fig. 11: The Gilbert Model



(a) Rate Distribution



(b) Frame Number in One GOP Distribution

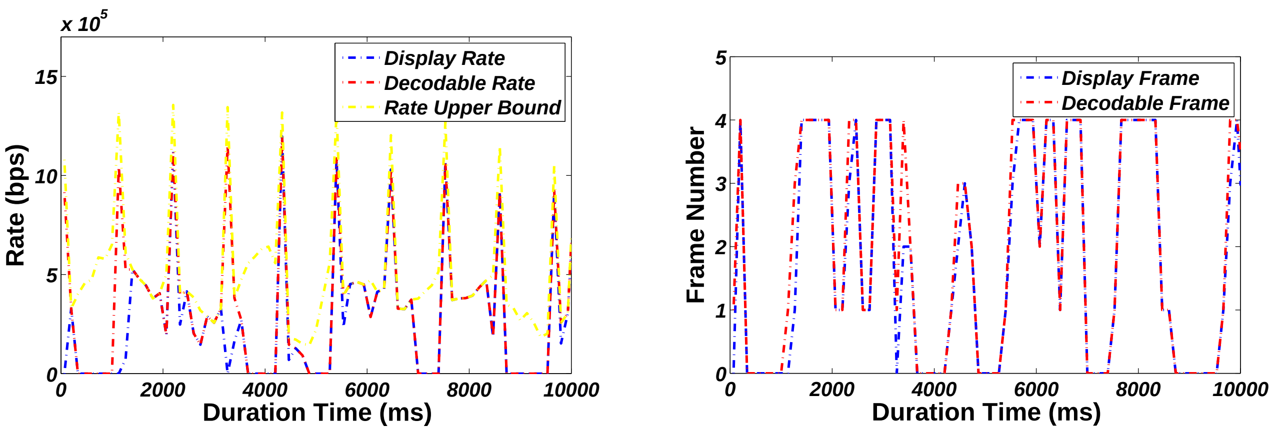


(c) Retransmission Number Distribution

Fig. 12: Retransmission approach under Bursty Loss ($p_u = 0.1, p_c = 0.6$)

redundancy.

Retransmission approach behaves much better than FEC approach when under large random loss or bursty loss. It will just add the prerequisite redundancy (Only retransmit packets when loss happens), which is the benefit. However, when propagation delay is very large, it is hard to use this approach in the Internet. Under such condition, we need further study.



(a) Rate Distribution

(b) Frame Number in One GOP Distribution

Fig. 13: FEC approach under Bursty Loss ($p_u = 0.1, p_c = 0.6$)