

# Unequal Growth Codes: Intermediate Performance and Unequal Error Protection for Video Streaming

Alexandros G. Dimakis and Jiajun Wang and Kannan Ramchandran

Department of Electrical Engineering and Computer Science,  
University of California, Berkeley, CA 94720.

{adim, junewang, kannanr}@eecs.berkeley.edu

**Abstract**—We investigate the design of fountain codes with good intermediate performance and built-in unequal error protection for low-delay video multicast. In particular, we design novel short-blocklength fountain codes for media streaming applications to multiple heterogeneous receivers and analyze their performance. Our theoretical contribution is the generalization of the growth code analysis for unequal error protection to suit the characteristics of video data. Simulation results show that the proposed method can effectively increase the number of decodable packets over a very wide range of packet drop rates and provide smooth and graceful video quality degradation for users with various channel conditions. The proposed scheme also enjoys the important benefits of much lower decoder complexity and simpler system architecture compared to traditional MDS erasure coding based solutions.

## I. INTRODUCTION

While today’s popular hybrid video codecs, such as MPEG and H.26x, can compress video efficiently, the compressed bit-stream is fragile to transmission losses. This fragility is a direct consequence of the prediction-based coding framework that underlies MPEG-like video codecs, since in order to decode the current frame correctly, all previous frames need to have been received correctly. Packet losses thus lead to predictor mismatch or “drift” between encoder and decoder. A number of schemes have been proposed to alleviate the problem of drift in a unicast setup, mostly by using Automatic Repeat Request (ARQ), Forward Error Correction codes (FEC) or a combination of both, i.e. Hybrid ARQ. In a multicast setup, in order to avoid repeat request implosion at the encoder, the scheme of layered and staggered Hybrid ARQ/FEC, such as in [1] and [2], can be adopted such that with some additional delay, the receivers can choose to receive the appropriate amount of parity packets according to their own bandwidth constraints and channel conditions. Whether for unicast or multicast, maximum distance separable (MDS) codes, such as Reed-Solomon codes, have been the popular choice of channel codes for their good performance, especially for low blocklengths. Such MDS-based FEC codes can recover all the dropped packets when the total number of received packets is at least equal to the number of data packets. However, due to a combination of delay and bandwidth constraints, receivers may not be able to receive as many parity packets as the number of lost data packets. When this happens, none of the *lost* packets can be recovered. In these cases the number of dropped packets is typically large and the error effect will propagate through the video until the next intra-frame is received. The quality degradation is often sudden and severe and is vividly described as the “cliff” effect. In this work, we aim to design short fountain codes that can provide smooth and graceful quality degradation over a wide range of packet drop rates. Fountain codes will also enjoy minimal encoding and decoding complexity as well as very simple system design.

The idea of using fountain codes with scalable video multicast has been proposed before, such as in [5], [6]. Both works apply fountain codes to different layers of a scalable video bitstream so that decoders

can receive parity packets to recover all the lost packets for each layer (starting from the base layer). Although also using fountain codes in multicast with heterogeneous receivers, this work instead focuses on the design of *short-blocklength, systematic fountain codes* with good intermediate performance and built-in unequal error protection (instead of relying on a scalable video bitstream).

One key aspect of our setup is that we do not use the fountain code to send all the data: The fact that the code is systematic corresponds to a layered approach: the original video bitstream is sent uncoded (e.g. through one multicast group) and a second smaller stream of fountain encoded parity packets is sent separately (e.g. through another multicast group) to recover as many lost packets as possible. The goal is for receivers that cannot receive enough parity packets to recover all the dropped data packets due to delay or bandwidth constraints, to still be able to recover a large portion of them. Out of all the dropped packets, receivers will also have a much higher probability of recovering the packets that are more crucial to video quality. This setup may be used in multimedia multicast over wireless ad-hoc or peer-to-peer networks. The key element is that communication is happening over channels with unknown and varying packet drop rates. The universal character of fountain codes [4] allows us to design erasure codes that perform well for a wide range of channel conditions. The two key innovations in this work are the fountain code design for *good intermediate performance* and *unequal error protection*. Our theoretical contribution is the generalization of the growth code analysis [3] for unequal error protection. We show how the performance of a restricted decoder can be completely characterized probabilistically and use this prediction to design a degree distribution that performs well in practice. Experimentally we show that our design increases the number of decodable packets over a very wide range of channel drop rates and provides smooth video quality degradation for users with various channel conditions.

## II. PROPOSED METHOD

### A. Review of fountain codes

The simplest fountain codes are Luby Transform (LT) Codes [4] that operate as follows: assume we want to communicate  $k$  initial data packets over a packet erasure channel. Let  $\Omega_1, \Omega_2, \dots, \Omega_k$  be a probability distribution on  $\{1 \dots k\}$ ,  $\sum_{i=1}^k \Omega_i = 1$ . An encoded packet is generated as follows:

- Select a random degree  $d$  using the distribution  $\Omega_1, \dots, \Omega_k$
- Choose  $d$  uniformly random input data packets
- Form the encoded packet by performing the bitwise XOR of the  $d$  selected input data packets.

The encoded packet is then transmitted and this process can be repeated a potentially infinite number of times. The property of generating each encoded packet *independently* of all others is the defining property of fountain codes which are also called *rateless*

codes because they have no pre-defined number of encoded packets and hence rate.

The receiver, after receiving a number of encoded packets, executes the decoding algorithm which is a simple form of belief propagation: it searches for one encoded packet of degree one - this is just an input data packet. This packet can then subsequently be XORed with other encoded packets of degree two and possibly generate new encoded packets of degree one. This iterative procedure continues until decoding is complete or no other degree one packets can be found. Therefore, both decoding and encoding are extremely simple and computationally efficient. The key part is the careful design of a good degree distribution  $\Omega$ . It is remarkable that degree distributions exist [4] such that this simple decoder succeeds with high probability after having received  $r = (1 + \epsilon)k$  packets - only slightly more than  $k$  encoded packets.

The traditional approach to communicating over erasure channels is using MDS codes (like Reed-Solomon codes) that generate a fixed number  $n$  of encoded packets from  $k$  initial data packets with the property that *any*  $k$  out of the  $n$  encoded packets suffice to reconstruct the original  $k$ . The advantages of MDS over fountain codes is that their guarantees are deterministic (as opposed to probabilistic) and that there is no overhead, while fountain codes have an overhead which becomes very small only for large  $k$ . The two main disadvantages of MDS codes are higher encoding and decoding complexity and inflexibility that comes with a fixed rate.

1) *Intermediate performance*: The design of fountain codes is usually optimized for a receiver that starts with no data, receives slightly more than  $k$  parity packets and only full recovery of all  $k$  data packets is considered success.

However for video streaming applications, it might be the case that some receivers, due to delay/bandwidth constraints or bursty packet drops, receive fewer than  $k$  packets for a time window. In this case it is of course impossible to fully recover all input packets, but it would be very desirable to still be able to decode a large number of input packets, since partial information has value for video applications. The main disadvantage of MDS codes is apparent for this problem: *while any  $r = k$  received packets will suffice to recover everything,  $r = k - 1$  packets cannot be used to recover any lost packet at all*. We would like to design codes with a smoother performance curve: while they require a slightly larger number of packets to recover everything, smaller  $r$  still have good performance.

It turns out that LT-like conventional rateless codes designed for  $r \approx k$  have very poor intermediate performance even if the number of received packets is very close to  $k$ . On this problem, there has been some related work in the very recent literature: Sanghavi [8] investigated the optimal intermediate performance of fountain codes (when the receiver starts with no initial data). He obtains asymptotic (almost tight) upper and lower bounds on the number of decodable packets, for degree distributions optimized for any specific  $r$ . While these bounds are very useful for large  $k$ , in this paper we are interested in streaming applications with delay constraints that force  $k$  to be small. Also, we are not interested in a specific  $r$  but want to *design a code that behaves well for a range of  $r$ , from very high to very low packet drop rates*.

Similar in spirit, Growth codes [3] were designed for sensor networks in catastrophic or emergency scenarios. They try to maximize the number of decodable packets for every  $r$  by using the following design principle: A received parity packet is *immediately decodable* if  $d - 1$  of the packets used to form this parity packet are already decoded/known. On the other hand, if all  $d$  packets used to form a parity packet are already known, this parity packet is useless. Growth

codes try to maximize the probability that each parity packet is immediately decodable for a receiver that accumulates more and more packets. As the number of packets that the receiver has increases, the degree  $d$  for each new encoding symbol needs to increase, hence the name growth codes. To understand this, consider selecting the degree of a single encoded packet and wanting to maximize the probability that this packet is immediately decodable to a receiver that has access to  $r'$  randomly selected data packets. Clearly if  $r'$  is very small, say the receiver has no packets, the optimal degree will be 1. As  $r'$  increases, the optimal degree grows as a function of  $r'$ . The resulting degree distribution, while not optimal for any particular  $r$ , performs very well for a wide range of packet drop rates.

2) *Unequal Error Protection*: We will be interested in optimizing a performance metric similar to growth codes, but with unequal error protection (UEP). UEP is crucial for MPEG-like streams, since packets corresponding to I-frames are far more important as far as video quality is concerned compared to P-frames. There is a very natural way to introduce unequal error protection in fountain codes introduced by Rahnavard et al [7]: Assign unequal probability to each input data packet, the more important the packet, the higher the probability. During the encoding process, after selecting a degree  $d$  from the degree distribution, choose  $d$  data packets to be XORed according to the probability assigned. Naturally, packets with higher assigned probability will be XORed in more parity packets. Therefore the decoder does not need to be modified and the packets of higher importance will have higher probabilities of being recovered. See [7] for an analysis of asymptotic properties of UEP rateless codes and bounds on the ML decoding error probabilities.

## B. Proposed Code Design

The proposed Unequal Growth codes, protect some input packets more than others and have good performance for a range of  $r$  received packets available at the decoder. For video streaming we will be using *systematic encoding* since it dramatically reduces the complexity when low packet drop rates allow most packets to be received uncoded. In addition to the uncoded packets, some high degree encoded packets will be generated from some degree distribution  $\Omega$  that we want to design. This can be thought of as having one multicast group streaming MPEG-encoded video and another multicast group streaming parity packets. For the purpose of analysis, we will use unequal error protection of two levels (for I-frames and P-frames) but our technique easily generalizes to an arbitrary number of protection levels. We present sketches of proofs due to space limitations.

Let  $N_1, N_2$  denote the number of high importance and low importance packets respectively. Let  $s_1, s_2$  be two constants such that input packets of high importance are selected (without replacement) with probability  $\frac{s_1}{N_1 s_1 + N_2 s_2}$  and packets of low importance with probability  $\frac{s_2}{N_1 s_1 + N_2 s_2}$ . We can now analyze what is the probability that an encoded packet of degree  $d$ , with packets selected using parameters  $s_1, s_2$ , will be immediately useful for a decoder that has already decoded  $r_1$  high importance and  $r_2$  low importance packets.

First consider a packet of degree  $d$  that is the XOR of  $d_1$  high importance packets and  $d_2$  low importance packets ( $d_1 + d_2 = d$ ), and say those packets have degrees  $(d_1, d_2)$ . We can count the number of such useful packets:

*Lemma 1*: The number of encoded packets with degrees  $(d_1, d_2)$  that are immediately useful to a receiver that has decoded  $(r_1, r_2)$  packets already, is:

$$\mathcal{N}_u(d_1, d_2) = (N_1 - r_1) \binom{r_1}{d_1 - 1} \binom{r_2}{d_2} \quad (1)$$

$$+ (N_2 - r_2) \binom{r_2}{d_2 - 1} \binom{r_1}{d_1}. \quad (2)$$

*Proof:* (sketch) Notice that there are  $\binom{N_1+N_2}{d}$  possible encoded symbols total. Out of those, the ones that can be immediately useful in decoding a high importance data packet must be the XOR of one important data packet not in  $r_1$ ,  $d_1 - 1$  high importance data packets out of the  $r_1$  known and  $d_2$  low importance data packets out of the  $r_2$ . This can be done in  $\mathcal{N}_u^1 = \binom{N_1 - r_1}{d_1 - 1} \binom{r_2}{d_2}$  ways. Similarly there are  $\mathcal{N}_u^2 = \binom{N_2 - r_2}{d_2 - 1} \binom{r_1}{d_1}$  ways for decoding a low importance data packet. ■

We can now proceed and compute the probabilities that an encoded packet of degree  $d$  is immediately useful to decode a high and low importance data packet:

*Lemma 2:* The probability that an encoded packet of degree  $d$  can be used to immediately recover a high importance packet for a receiver that has decoded  $r_1, r_2$  packets already is:

$$P_1^d(r_1, r_2) = \sum_{d_1, d_2, d_1+d_2=d} \mathcal{N}_u^1(d_1, d_2) \frac{d_1 s_1 + d_2 s_2}{Z} \quad (3)$$

where  $Z = \sum_{d_1, d_2, d_1+d_2=d} \binom{N_1}{d_1} \binom{N_2}{d_2} (d_1 s_1 + d_2 s_2)$  is the normalizing constant. Similarly the probability of recovering a low importance packet  $P_2^d(r_1, r_2)$  can be found by replacing  $\mathcal{N}_u^1$  with  $\mathcal{N}_u^2$ .

*Proof:* (sketch) The unequal way of selecting the initial data packets imposes a probability measure on each data packet: packets which have more connections to important input packets have higher probability of being created. Therefore, to find the probability that an encoded packet of degree  $d$  is immediately useful we need to add all the possibilities of  $d_1, d_2$  that add up to  $d$  and weight each by the appropriate probability which is proportional to  $d_1 s_1 + d_2 s_2$ . The constant  $Z$  is selected to appropriately normalize the probabilities. ■

Our results generalize the growth code analysis [3] for unequal error protection. Figure 1 illustrates the computed probability  $P_1^d(r_1, r_2)$  as a function of  $r_1$  for  $r_2 = cr_1$  for various degrees, and  $c = 1.75$ . Observe that larger degrees become optimal as the number of decoded packets at the receiver increases. Since we are aiming at a wide range of operation points we need a degree distribution that is a mixture of degrees. In practice the number of parity packets available at the decoder is generating a dynamic process that we want to track. Consider the S decoder [3] that sorts the received packets in non-decreasing order of their degrees and sequentially tries to decode a parity packet immediately or discard it, if it cannot be used. While one would never use this decoder in practice, it is tractable for finite length analysis and provides a lower bound on the performance of the actual fountain decoder [3]. The main benefit of our analysis is that  $P_1^d(r_1, r_2), P_2^d(r_1, r_2)$  can be used to track the time evolution of  $(r_1(t), r_2(t))$  for the S decoder, when  $t$  packets have been received:

*Proposition 1:* The probabilistic evolution of  $(r_1(t), r_2(t))$  under the S decoder can be completely characterized using  $P_1^d(r_1, r_2), P_2^d(r_1, r_2)$  and the degree distribution  $\Omega$ .

*Proof:* (sketch) Assuming a packet drop rate  $p_e$ , the systematic part of the code will succeed in delivering  $r_1(0) = N_1(1 - p_e)$  high importance packets and  $r_2(0) = N_2(1 - p_e)$  low importance packets in expectation. Therefore, the first (lowest degree) encoded parity packet (of degree  $d(1)$ ) will be useful to recover a high importance packet with probability  $P_1^d(r_1(0), r_2(0))$  and a low importance packet with probability  $P_2^d(r_1(0), r_2(0))$ . After one such packet,  $r_1(1) = r_1(0)$  with probability  $1 - P_1^d$  and  $r_1(1) = r_1(0) + 1$  with probability  $P_1^d$ . Similarly for any state  $(r_1(t), r_2(t))$  the next possible states are  $(r_1(t), r_2(t)), (r_1(t) + 1, r_2(t)), (r_1(t), r_2(t) + 1)$ . The degree distribution  $\Omega$  can be used to find the probability that the  $i$ th parity packet has degree  $d$  and by averaging for all possible degrees we obtain the transition probabilities for each  $t$  and  $\Omega$ . ■

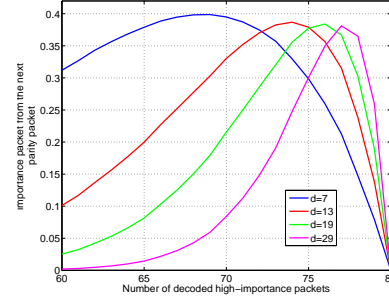


Fig. 1. The probability of recovering a high-importance data packet from the next parity packet, i.e.  $P_1^d(r_1)$ , when it has degree 7, 13, 19 and 23 respectively, as a function of the number of high-importance packets the receiver already has, i.e.  $r_1$ .

Different  $\Omega$  distributions yield different expected trajectories, and we use these predictions to design a good degree distribution  $\Omega$  for the desired range of packet drop rates. For the experiments that follow, we used the distribution ( $\Omega_7 = 0.6, \Omega_{12} = 0.15, \Omega_{17} = 0.08, \Omega_{22} = 0.05, \Omega_{27} = 0.04, \Omega_{37} = 0.08$ ).

### III. EXPERIMENTAL RESULTS AND DISCUSSION

We first present the intermediate performance of the proposed short-blocklength fountain code without UEP using the proposed degree distribution. Using 200 data packets and 20 fountain coded parity packets, Figure 2(a) shows the average fraction of decodable packets with (1) fountain codes (with intermediate performance but no UEP), (2) 20 extra random data packets, (3) no extra parity packets, and (4) MDS-based FEC. We can see that the proposed method can reduce the packet drop rate smoothly over a wide range of channel loss rates (assumed independent packet drops) compared to MDS-based FEC. There is also a huge improvement over the case where the receivers can receive 20 extra random uncoded data packets. Note that even when the average packet drop rate is below 10%, which is the rate MDS FEC is designed for, MDS FEC cannot guarantee 100% data recovery, e.g. a (220, 200) RS code cannot recover any lost packet if only 199 packets are received. This is a result of having finite blocklength.

We now demonstrate the performance of the proposed fountain code with unequal error protection. Again we assume 200 data packets and 20 parity packets. We assign importance level 10 to 80 packets, importance level 5 to another 40 packets and importance level 1 for the rest of the packets. Figure 2(b) shows the average fraction of decodable packets of different importance levels versus average channel packet drop rate. We see that with the built-in UEP, the important packets now have a much higher fraction of decodable packets. Again, the proposed method enables a smooth performance over a wide range of average packet drop rates.

Finally we show some encouraging results of the proposed codes on video sequences. For the baseline video encoding, we used H.264 (JM10.2) encoder to encode 60 frames of `foreman` and `tempete` (both CIF:  $352 \times 288$ ) sequences at 348 kbps and 836 kbps respectively (30 fps with I-P-P-P GOP structure and a GOP size of 15). The average PSNR without packet drop is 34.89 dB for the `foreman` sequence and 31.75 dB for the `tempete` sequence. The output file mode is chosen to be RTP packets with fixed size. The fountain code is applied over one second's duration of video ( $\sim 220$  packets). The generated fountain coded packets ( $\sim 22$ ) are assumed to be sent to a separate multicast group that receivers can

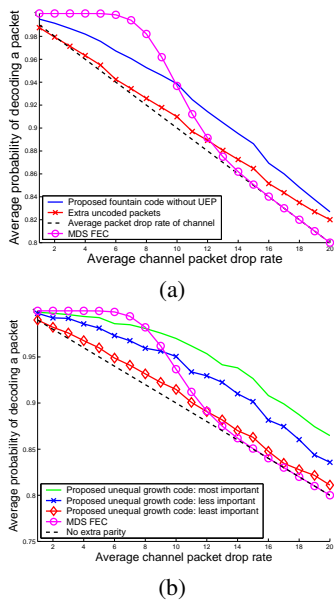


Fig. 2. (a) Average fraction of decodable packets with (1) fountain codes (with intermediate performance but no UEP), (2) extra random uncoded packet, (3) no parity packets, and (4) MDS FEC. (b) Average fraction of decodable packets of various weights compared to MDS FEC.

subscribe to if bandwidth permits.

For this experiment, we assume each user has 10% extra bandwidth to receive coded parity packets to recover dropped packets. We compare 3 systems: (1) video with random intra refresh but no parity packets coded at 110% of the original bitrate, (2) baseline video with MDS FEC and (3) baseline video with the proposed fountain codes with UEP. The 3 systems have equal PSNR when there are no packet drops. For System (3), we assigned weight 10 to packets containing I-frames, weight 5 to the first 10 packets containing P-frames (of each GOP) and weight 1 to the rest of the packets (we do not use the 3-slice partition for the current implementation). These weights are currently set based on heuristics. It is part of our ongoing work to optimize this to achieve better performance. Figure 3 plots the average PSNR (over 20 trials) versus packet drop rates for all 3 systems. The preliminary implementation of the proposed method has a smooth quality degradation over a wide range of packet drop rates, even when the packet drop percentage is far above the percentage of extra rate available. The proposed scheme also consistently outperforms random intra refresh.

One important benefit of the proposed method over MDS-based FEC is greatly reduced decoding complexity. A popular choice, the Reed-Solomon codes, require  $O(n^2)$  decoding complexity where  $n$  is the blocklength while the proposed codes have only  $O(n)$  encoding and decoding complexity. In addition, since each parity packet is generated independently of the others, this enables a very simple and robust system architecture. If the server serving the parity packets crashes, it can pick up at any point and start generating new parity without any of the previously generated parity packets going to waste. Further, when there are multiple servers, it eliminates the need for coordinating parity packet generation.

#### IV. CONCLUSIONS AND FUTURE WORK

We have presented the design of a novel short-blocklength fountain code with built-in unequal error protection (UEP) for video multicast. The goal is to target heterogeneous receivers with a wide range

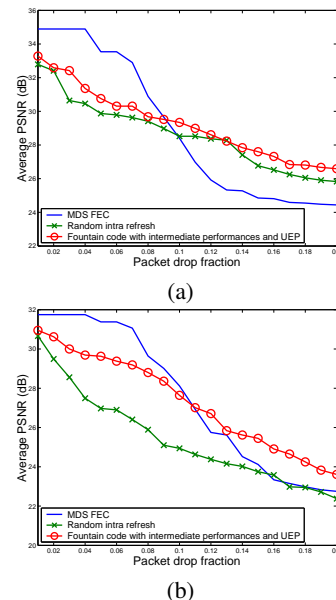


Fig. 3. PSNR vs. average packet drop probabilities for 3 systems: (1) baseline video with random intra refresh, (2) baseline video with MDS FEC and (3) baseline video with the proposed fountain codes with UEP. The 3 systems have equal PSNR when there are no packet drops. (a) *foreman* sequence (60 frames total, 348 kbps, 30 fps, GOP size 15) (b) *tempete* sequence (60 frames total, 836 kbps, 30 fps, GOP size 15).

of channel conditions and bandwidth constraints. This work is a first step towards a much more sophisticated code design that is tightly integrated with the video bitstream packetization strategy. Our results show that the proposed code design enables smooth quality degradation over a wide range of packet drop rates and consistently outperforms random intra refresh. It is our ongoing research to optimize the degree distribution and the weight assignment of video RTP packets. An inner code can also be used to further protect important packets (as in [9]), such as those containing I-frames, block modes and motion vectors. We also plan to adopt the 3-slice partition mode of H.264 packetization scheme to facilitate UEP of the fountain codes and compare with MDS-based UEP.

#### REFERENCES

- [1] P. A. Chou, A. E. Mohr, A. Wang, S. and Mehrotra, "Error control for receiver-driven layered multicast of audio and video," *IEEE Trans. on Multimedia*, 2001.
- [2] W. T. Tan and A. Zakhori, "Video multicast using layered FEC and scalable compression," *IEEE Trans. on Circuits and Systems for Video Technology*, 2001.
- [3] A. Kamra, J. Feldman, V. Misra and D. Rubenstein, "Growth codes: Maximizing sensor network data persistence," *Proc. of ACM SIGCOMM*, 2006.
- [4] M. Luby, "LT codes," *Proc. IEEE FOCS*, 2002.
- [5] Q. Xu, V. Stankovic and Z. Xiong, "Wyner-Ziv video compression and fountain codes for receiver-driven layered multicast," *Proceedings of Picture Coding Symposium*, 2004.
- [6] J-P. Wagner, J. Chakareski and P. Frossard, "Streaming of scalable video from multiple servers using rateless code," *Proceedings of International Conf. on Multimedia and Expo*, 2006.
- [7] N. Rahnavard, B.N. Vellambi and F. Fekri, "Rateless Codes with Unequal Error Protection Property," *IEEE Trans. on Inform. Theory*, to appear, 2007.
- [8] S. Sanghavi, "Intermediate performance of rateless codes," *Information Theory and Applications*, 2007.
- [9] A. Shokrollahi, "Raptor codes," *IEEE Trans. on Information Theory*, 2006.