Frame Fountain: Coding and Decoding MAC Frames

Qingmei Yao, Chong Tang, Shaoen Wu

School of Computing
University of Southern Mississippi
118 College Drive, Box 5106 , Hattiesburg, MS, 39406, USA
Emails: qingmei.yao, chong.tang@eagles.usm.edu, shaoen.wu@usm.edu

Abstract

To address the large latency and degraded network throughput due to the retransmission triggered by frame loss in high speed wireless networks, this paper proposes a solution called *Frame Fountain* that encodes and decodes data frames in the MAC layer. This solution intelligently encodes a number of redundant frames from original frames upon link loss rate so that a receiver can effectively recover lost original frames without significant retransmissions. Meanwhile, various probability distributions are investigated to find an optimal one as the degree distribution adopted by this coding solution. Extensive experiments show that, working with a degree distribution improved upon robust Soliton distribution, Frame Fountain can recover most of original frames with large probability.

Index Terms

FEC codes, Fountain codes, MAC, Frame Fountain

I. Introduction

Wireless networking technologies have been widely deployed in civil and military applications such as 3G/4G and IEEE 802.11 WLAN networks. However, wireless communication suffers from frame losses due to channel fading, shadowing, mobility and transmission collisions (interferences). Frame loss significantly undermines wireless network performance in that 1) latency is enlarged and 2) throughput is degraded. The large latency is incurred by the retransmission of lost frames in the MAC layer that is part of most MAC protocols for reliable link layer point-to-point transmission. For example, in IEEE 802.11 [**?**], if a source node which sends a data frame does not receive an acknowledgement (ACK) on time, it will retransmit this data frame until it receives the ACK from the receiver node or a timer (usually called Retransmission Timeout value) expires. Large latency afflicts multimedia applications. Meanwhile, the frame delivery ratio is significantly hurt by frame loss. Therefore, the throughput is impaired as well.

To improve wireless network performance experiencing frame loss incurred from channel instability, one important solution is to *code* information into a format different from its original presentation. Coding has been studied in different network layers. In the *physical layer*, channel coding algorithms, like Trellis [**?**] and Viterbi [**?**], provides redundancy, resilience, error detection and correction for modulated bits transmitted on wireless links. In the *networking layer*, in recent years, network coding solutions have been proposed for opportunistic transmission to improve network performance. For example, COPE [**?**] proposed for wireless mesh networks by Katti, *et al.* not only forwards the packets but also mixes packets form different sources into a single transmission and decomposes the packets at the receiver. In upper layers, a coding concept called Digital Fountain has been proposed by Byers, *et al.* [**?**] to generate a stream of packets, including some redundant packets, like in water fountain to address potential packet loss in multicast applications that do not allow retransmission. Since then, many Digital Fountain coding methods have been invented such as Luby transform codes (LT codes)[**?**] and Raptor codes [**?**].

This paper proposes a coding solution in *MAC* layer, named Frame Fountain, to reduce the retransmission because of the frame losses in high speed wireless networks, e.g. IEEE 802.11n. Frame Fountain encodes a number of redundant frames from a stream of original data frames upon certain probabilistic distributions and the frame loss ratio of a wireless link, and then sends all of these frames in a batch to the receiver with the expectation that even if some of these frames are lost, the complete set of raw frames can still be recovered with large probability. In particular, multiple probabilistic distributions were investigated to find an optimal one that can generate redundancy of various degrees in encoding to recover lost original frames with a great probability.

The remaining of this paper is organized as follows. Section II briefly introduces the background and previous work followed by our motivations. Section III discusses the coding concept of our proposed Frame Fountain and the coding methods using different probability distributions. Simulation results of comparing different coding methods are reported and analyzed in Section IV. Finally, the conclusion is given in Section V.

II. Related Work and Motivations

This section briefly reviews the channel coding methods in the physical layer and Digital Fountain coding proposals in the application layer. Then the motivations of our work to code frames in the MAC layer are presented.

A. Forward Error Correction Coding

Forward Error Correction (FEC), also called channel coding, was originated by Shannon in 1948 [**?**]. It is an important technique that significantly improves data transmission reliability in error-prone channels, particular true in wireless communication, by using an error control structure that adds redundant information through some algorithm to the original messages at the sender before transmission to make the message more resilient against noise and interference. If error occurs in the transmission on the channel, the receiver can use the redundancy to correct the errors in messages unless the number of errors is beyond the error correct capability of the FEC algorithm. Generally, there are two categories of FEC codes: convolutional codes and block codes.

Convolutional codes[**?**] are one of the most widely used channel codes in numerous applications to achieve reliable data transmission. Convolutional codes were first introduced by Elias [**?**] in 1957 as an replacement of block codes. This kind of codes works on arbitrary length of bits or symbols. The encoded bits depend not only on the current k raw bits but also on the previous input bits. Thus they can convert the entire data stream into one codeword. There are many convolutional coding algorithm such as low-rate convolutional coding [**?**], high-rate recursive convolutional coding [**?**] and cyclic convolutional codes [**?**]. Viterbi algorithm [**?**] which was first proposed by Viterbi in 1967 is the most often used algorithm to decode the bit streams coded by convolutional coding algorithms.

Block codes [**?**] work on fixed-size blocks of bits or symbols and can be used to either detect or correct errors. These coding methods use a block of k original information bits as input and generate a block of n coded bits in which $n - k$ redundant bits are added. These codes can be represented as (n, k) block codes. Many block codes have been proposed in the past such as Hamming codes[**?**], Bose-Chaudhuri-Hocqhuenghem (BCH) codes [**?**], Reed-Solomon codes [**?**] and Low Density Parity Check (LDPC)[**?**] .

B. Digital Fountain

Digital fountain was introduced in 1998 by Byers, *et al.* in the paper [**?**] to address end-to-end packet loss in multicast applications where no retransmission is allowed. The core of Digital Fountain is rateless error correcting coding algorithms whose code rates vary. With these coding algorithms, potentially a limitless sequence of encoding symbols can be generated from a given set of source symbols, say K symbols, and the original k source codes can be recovered from any of the encoded P symbols where P is no less than K. Although Reed-Solomon (RS) codes[**?**] and LDPC[**?**] codes were invented in 1960s, they can also be considered as Digital Fountain codes.

Luby Transform codes (LT codes) are the first practical Digital Fountain codes presented in 2002 by Luby [**?**]. They are rateless coding methods that encode and generate symbols on the fly. In encoding, first, an information message is divided into blocks of equal size and a degree d ($1 \leq d \leq n$, where n is the number of original blocks) is generated by the degree distribution. Then d blocks are randomly selected and are combined with bitwise XOR operation. The encoded block is transmitted in a packet. When enough encoded packets are generated or the sender get an acknowledgement(ACK) from the receiver the encoding process stops. The decoding process of LT codes depends on the identity of the received packets. If a degree-1 packet is received, it is itself an original block. Then this degree-1 packet is used to XOR with other received packets to remove itself from other encoded packets of degree greater than 1, and meanwhile decrement the degree of the encoded packet. If new degree-1 packets are found, the decoding process continues iteratively until all the original messages have been recovered.

It should be noted that the degree distribution, which is a probability distribution to define the number of blocks combined in one packet, is extremely significant in LT codes. To completely recover the orignal data and ensure the LT process success it has to meet the basic requirements: a few encoding packets must have high degree while many other packets must have low degree to keep the total number of operations small to be practical. To meet the requirements, Luby discussed two distributions: the ideal Soliton distribution and the robust Soliton distribution. Further discussion of these distributions is deferred to Section III where our proposed MAC frame encoding is presented.

C. Our Motivation

Although coding approaches have been well studied in the physical layer for bit streams, in the networking layer for opportunistic routing and in the application layer for reliable end-to-end transmission, there is no work attempting the coding for the MAC layer for hop-to-hop communication. In wireless networks, the unstable hop-by-hop links likely incur large frame loss rate. Retransmission mechanisms in most of MAC protocols for wireless networking is possible to yield large latency and significantly undermines network throughput. *We are therefore motivated to exploit encoding solutions to efficiently address these challenges due to the frame loss in wireless networks.*

III. Frame Fountain

A. *Frame Fountain: coding and decoding*

In this work, we propose an approach called Frame Fountain that intelligently encodes redundant frames from a stream of original MAC frames based on the current frame loss rate (r_l) of a wireless link. Then, both redundant and original frames are transmitted in a batch. It is expected that all original information can be recovered at a receiver even with a limited number of frames lost in the transmission. Namely, when a receiver receives the enough number of frames, it can probabilistically recover all the original frames in the batch immediately. In addition to reduce the retransmission, another strength of Frame Fountain is that for a batch of frames only one ACK needed to feedback to the source. Therefore, Frame Fountain can definitely improve the throughput efficiency and reduce the network latency. Below discussed in detail are the encoding and decoding processes and the degree distributions.

1) Encoding Process:: The encoding process of Frame Fountain is shown in Algorithm 1. In the encoding, there are two types of frames in a frame flow (or batch): original frame(s) and encoded redundant frame(s). The frame type is marked in the frame header. The number of redundant frames, k, is calculated upon a given frame lose rate r_l of a wireless link. Denote n as the number of original frames in each encoding cycle. Then $k = \lceil n/(1-r_l) \rceil - n$. Encoding stops at a sender either when the sender receives an ACK from the receiver indicating that original frames have been recovered or when the number of sending out frames is up to $n + k$.

To generate an encoded frame, the encoding algorithm first computes a degree number d with a degree distribution. Then it randomly (or sequentially with random starting point) selects d distinct original frames and combines these frames into one encoded redundant frame with bitwise XOR operation. In this process, two encoding methods are proposed: *random coding* and *continuous coding*. In the random coding, d distinct original frames are *randomly* selected and encoded into one redundant frame with bitwise XOR operation. In the continuous coding, a *streak* of d *sequential* original frames are encoded, but the starting point of the streak is *randomly* selected. In continuous coding, if the end of the entire sequence of original frames is reached, the fetching cyclically starts from the beginning of the entire sequence of original frames until d frames are taken. For example, if the entire sequence of original frames is a, b, c, d, e and the degree distribution generates a degree of 3. Random coding likely to generate a redundant frame of $c \oplus b \oplus e$, while continuous coding possibly generates a redundant frame of $e \oplus a \oplus b$.

Algorithm 1 The encoding process of Frame Fountain algorithm

```
1: calculate the number of redundancy frames $k$ by the given loss rate $r_l$ and the number of original frames needed to sent $n$
       $k = \lceil n/(1-r_l) \rceil - n$
2: send out the $n$ original frames
3: repeat
4:     choose a degree $d$ from the distribution $\rho(d)$
5:     choose $d$ distinct randomly or sequentially frames $(m(i_1), m(i_2), \cdots, m(i_d))$ form $n$ original frames
6:     send the frame $m(i_1) \oplus m(i_2) \oplus \cdots \oplus m(i_d)$
7: until the sender get an ACK from the receiver or the number of the redundancy is up to $k$
```

2) Decoding process:: Algorithm 2 explains the detail of the decoding process of Frame Fountain. The received frames are stored into two buffers: B_o for those original frames and B_e for those encoded redundant frames. If the received frame F_r is an encoded frame (this could be done by checking the mark in the frame header) with coding degree d, the decoding algorithm does XOR operation upon this frame against buffer B_o to get rid of those original frames encoded into the frame F_r but has been received and stored in buffer B_o. For example, if F_r is encoded as $A \oplus B \oplus C \oplus D$ with a degree of 4 and if C and D have been received in B_o, then the decoding does $F_r \oplus C \oplus D = A \oplus B$. After the operation, F_r is reduced to a frame F_r' with the degree reduced to d_r' by the number of original frames ruled out. If d_r' is 1, the frame F_r' should be an original frame. If it is a new original frame that is not stored in buffer B_o, it is first added into buffer B_o. Then, this new original frame is XORed against the left encoded frames in buffer B_e to iteratively conduct the decoding until no more decoding is possible to reduce the degrees of encoded frames in buffer B_e. Otherwise, if F_r' is still an encoded frame as in the example, it is stored in buffer B_e to wait till new original frame(s) to arrive to be decoded. One condition to stop the decoding process is when buffer B_o has n distinct original frames, namely all original frames have been received or recovered. Then, the receiver will stop receiving and decoding frames for this sequence of frames and send an ACK back to the sender. Another condition to stop decoding is when a receiver timer expires or the receiver receives the last frame of the receiving flow. In this case, the source has sent out all $n + k$ frames and stopped, but the receiver has not got enough frames to recover all original frames. Then, the receiver will send back an NACK with the information of the missing original frames. Then retransmission of lost frame(s) is necessary, but this can be done with a new sequence of frames in a new Frame Fountain cycle without

Algorithm 2 The decoding process of Frame Fountain algorithm

```
1: check the header of the received frame $F_r$
2: if $F_r$ is an original frame then
3:     put it into the buffer $B_o$
4:     check the number of frames in buffer $B_o$, if the number is $n$ send back an ACK and break
5: else
6:     compare the information in the frame header $F_r$
7:     do XOR operation with $F_r$ and frames which both in the buffer $B_o$ and also encoded in $F_r$, and get $F_r'$
8:     if the degree of $F_r'$ is 1 after the XOR operation then
9:         $F_r'$ becomes and original frame,
10:        check the header of other frames in buffer $B_e$ and reduce their complexity via do XOR operation with $F_r'$
11:        then put $F_r'$ in the buffer $B_o$ and check the number of frames of buffer $B_o$, if the number is n send back an ACK
           and break
12:    else
13:        put $F_r'$ into the buffer $B_e$
14:    end if
15: end if
16: repeat
17:    check header and degrees of frame in the $B_e$ buffer
18:    if find an frame with degree 1 then
19:        compare other frames' headers and do XOR operation if needed
20:        put this degree-1 frame into the $B_o$ buffer, check the number of frames of buffer $B_o$, if the number is n send back
           an ACK and break
21:    end if
22: until buffer $B_e$ has no frame with degree 1
23: check the number of frames of buffer $B_o$
24: if the number is $n$ then
25:    send back an ACK and tell the source to go to the next sequence's transmission
26: else
27:    send back an ACK to the source and piggyback the information of lost frames
28: end if
```

explicitly retransmitting lost frames one by one as in current MAC protocols. It should also be noted that the probability of this retransmission is significantly reduced because of the redundant frames.

The decoding process in this algorithm is suboptimal, because we do not do comparison between the encoded frames. For example, there are two encoded frames left in the buffer B_e: one is F_{e1} encoded by $a \oplus b \oplus c$ and the other one is F_{e2} encoded by $a \oplus c$. If we do XOR operation to the two frames we can get frame b ($F_{e1} \oplus F_{e2} = a \oplus b \oplus c \oplus a \oplus c = b$). However, this would increase the computation enormously and introduce more latency to the wireless network transmission. To improve the decoding algorithm of Frame Fountain is part of our future work.

B. Degree distributions

Just as in LT codes, the degree distribution is a critical element in Frame Fountain encoding algorithms. Suppose that each time the original and encoded frames are sent as a batch and each batch has n original frames and k encoded frames. We investigate several probability distributions as the degree distribution used in the Frame Fountain encoding process. They are presented as follows.

1) Uniform distribution:

$$p_i = 1/n \qquad \forall i = 1, \cdots, n.$$

2) Normal distribution:

$$\mu = \lfloor n/2 \rfloor, \quad \sigma = k/2$$

$$p_i = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \qquad \forall i = 1, \cdots, k$$

where $\quad x_i = \lfloor \text{ randn} * \sigma + \mu \rfloor$

3) sequential distribution:

$$p_i = \frac{1}{n - \lceil n/k \rceil} \qquad \forall i = \lfloor n/k \rfloor, \cdots, n$$

4) ideal Soliton distribution:

$$p_1 = 1/n$$

$$p_i = \frac{1}{i(i-1)}, \qquad fori = 2, 3, \cdots, n$$

*5) Robust Soliton distribution [**?**]:* First define, $R = c\ln(n/\delta)\sqrt{n}$, where c and δ are extra parameters; $c > 0$ is some suitable constant.

$$\tau_i = \begin{cases} \frac{R}{i \cdot n}, & \text{for } i = 1, 2, \cdots, (n/R) - 1, \\ \frac{R}{n}\ln(R/\delta), & \text{for } i = (n/R), \\ 0, & \text{otherwise.} \end{cases}$$

The robust Soliton distribution is denoted by a linear combination of the ideal Soliton distribution and τ_i,

$$p_i = \frac{p_i^* + \tau_i}{C},$$

where p_i^* corresponds to the ideal Soliton distribution and C is the normalization constant [**?**].

6) Improved Robust Soliton distribution: In the improved robust Soliton distribution, we also use $R = c\ln(n/\delta)\sqrt{n}$ as in the robust Soliton distribution where c and δ are extra parameters; $c > 0$ is some suitable constant.

$$\tau_i = \begin{cases} \frac{R}{(i-\omega)\cdot n} + \frac{R}{i \cdot n}, & \text{for } i = \omega, \cdots, 2\omega - 1 \\ \frac{R}{i \cdot n}, & \text{for } i = 2\omega, \cdots, n/R - 1, \\ \frac{n}{R}\ln(R/\delta), & \text{for } i = (n/R), \\ 0, & \text{otherwise.} \end{cases}$$

where $\omega = \lfloor n/k \rfloor$ and the ω must satisfy that $2\omega < n/R$, which means $R < k/2$. The ideal Soliton distribution has been changed to

$$p_i^* = \begin{cases} \frac{1}{i*(i-1)} + \frac{1}{n}, & \text{for } i = \omega + 1, \\ \frac{1}{i*(i-1)} + \frac{1}{(i-\omega)*(i-\omega-1)}, & \text{for } i = \omega + 2, \cdots, n, \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{Then,} \qquad p_i = \frac{p_i^* + \tau_i}{C}$$

where C is a normalization constant. The difference between the robust Soliton distribution and the improved one is on the sampling. In the improved distribution, only digrees larger than ω can be generated for encoding. The reason is that in Frame Fountain, enough degree-one symbols are already guaranteed by the original frames, the improved robust Soliton distribution only need to generate various large degrees (range: ω to n). In this case, enough frames can be encoded together as redundant frames to make sure there are enough diversity of encoded frames at the receiver.

IV. Experiment Analysis and Observations

Frame Fountain with different degree distributions are extensively evaluated. The following presents the experiment methodology and observations.

A. Experimental Methodology

The experiments are conducted on Matlab R2009b. The simulation in this experiment is only one hop frame transmission. The degrees of the encoded frames are notified to the receiver as an array and the indexes of the information of the encoded frames are assembled into a matrix for decoding at the receiver side. In the experiments, the encoding and decoding methods are implemented based on different degree distributions including Normal distribution, Uniform distribution, sequential distribution and the improved robust Soliton distribution. Each case repeats 10000 times. Both encoding methods are implemented and experimented: *random coding* and *continuous coding*

B. Observations

1) Coding Efficiency: To investigate the coding efficiency of two coding methods, *random coding* and *continuous coding*, upon various degree distributions, we compare their Cumulative Distribution Functions (CDF) of recovering original frames at a receiver. This experiment is configured with $n = 100$ and $r_l = 0.04$. In each run, the sender transmits 100 original and 5 encoded frames over the lossy link and the receiver gets only 100 frames in total, some of which are very likely encoded frames. Then the receiver conducts the decoding process to decode as many *original* frames as possible. Ideally, it is expected that a good encoding method is able to recover almost all original frames at its best. Namely, it is very desirable to see that an encoding method can recover all (or almost all) original frames (100 in our case) from received frames (100 in our case) with a large probability. The result of this case is plotted on Figure1, where the X-axis denotes the number of original frames

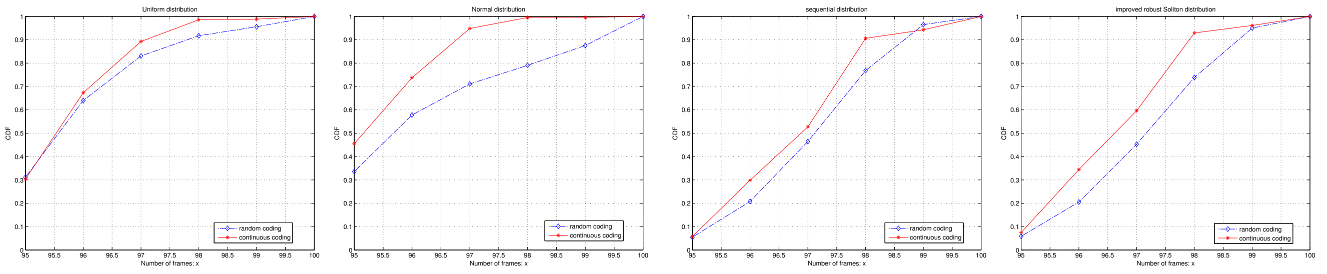


Fig. 1. CDFs of recovering original frames

the receiver can decode *at most* from received frames and the Y-axis shows the CDF of recovering the number of frames at the receiver's best. The solid line in the figure shows the coding efficiency of the *continuous coding* and the dash line for that of the *random coding*. From this figure, we obtain two observations. First, the random coding has more probability (with a sharper increase, larger gradient, on the figure) distributed on larger numbers on X-axis (towards the right) in *all* cases of various degree distribution. In another word, it is more likely to recover almost all original frames than continuous coding. Another observation is that, in the four degree distributions, the performance of uniform distribution is the worst. The normal distribution has the probability of 13% to decode 100 frames which is higher than other three distributions. However, its probability to decode more than 96 frames is 43% which is lower than 89.5% and 80% for sequential distribution and improved robust Soliton distribution respectively. Namely, its overall performance is worse. The sequential distribution has very similar performance to the improved robust Soliton distribution although the latter performs slightly better when the number of decoding frames is higher than 96.

Since the continuous coding performs much worse than the random coding, the remaining experiments are conducted upon the random coding only.

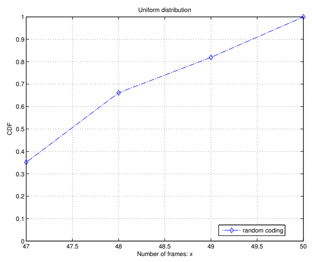
2) Comparison of Degree Distributions: We also attempt to identify the optimal degree distribution with more investigations. The first is to compute the *expectations* of probabilities of four degree distributions for recovering original frames at most. The expectation is calculated:

$$E[x] = \sum_{i=x_{min}}^{n} x_i p_i$$

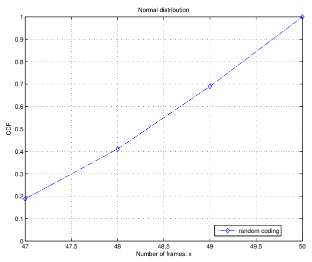
where x is the number of original frames that can be decoded by the receiver at its best. For the case of *random coding* in Figure1, the expectation are 96.34, 96.71, 97.53 and 97.59 for Uniform Distribution, Normal Distribution, Sequential Distribution and improved robust Soliton Distribution respectively. Since improved robust Soliton distribution has the largest expectation, it performs best in this case.

In addition, we conduct experiments as in Section IV-B1 under the setting of $n = 50$ and $r_l = 0.04$. The CDF results are shown on Figure 2 where the X-axis denotes the number of original frames the receive can decode *at most* from received frames and the Y-axis shows the CDF of recovering the number of frames at the receiver's best. We choose $n = 50$ to get close to the practice (currently in IEEE 802.11n, this number is about 64 in a typic channel coherent time). It is easy to see that the Uniform distribution is still the worst one since it only has the probability of 65% to decode more than 47 frames. Though the Normal distribution performs better when the decoded frames number is greater than 48, it performs worse for decoding small number of frames. On the other hand, the sequential distribution and improved robust Soliton distribution works much better than others. The expectation for sequential distribution is 48.63 and for the improved robust Soliton distribution is 48.68. Thus, the coding efficiency of improved robust Soliton distribution is slightly superior to the sequential distribution.

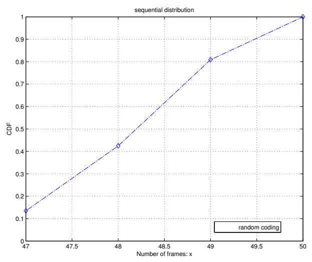
3) Impact of Loss Rate: To investigate the performance of random coding in various lossy environment, experiments are conducted to plot the CDF of recovering original frames in different loss rates for the sequential distribution and improved robust Soliton distribution. In these cases, there are 50 original frames with loss rate varying from 0.01 to 0.15.



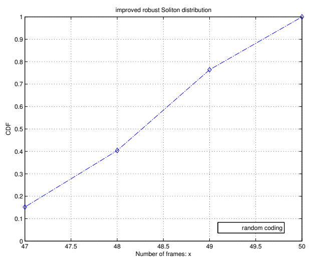
(a) uniform distribution



(b) normal distribution

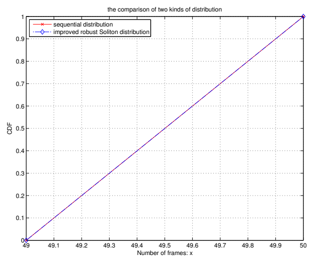


(c) sequential distribution

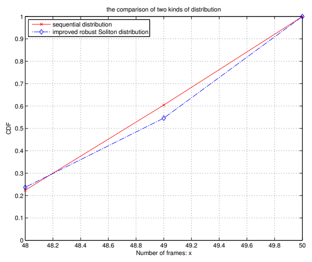


(d) improved robust Soliton distribution

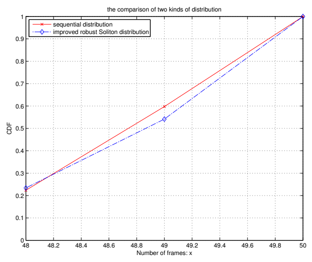
Fig. 2. Different degree distributions with random coding



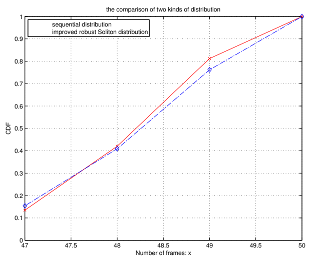
(a) comaprison for $n = 50$ & $r_l = 0.01$



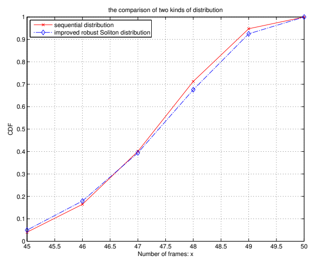
(b) comaprison for $n = 50$ & $r_l = 0.02$



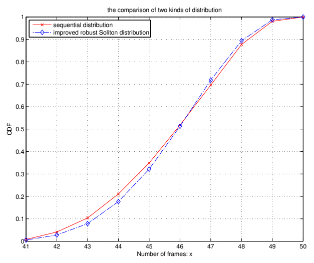
(c) comaprison for $n = 50$ & $r_l = 0.03$



(d) comaprison for $n = 50$ & $r_l = 0.05$



(e) comaprison for $n = 50$ & $r_l = 0.08$



(f) comaprison for $n = 50$ & $r_l = 0.15$

Fig. 3. sequential distribution and improved robust Soliton distribution with various loss rates

Figure 3 shows the results where X-axis denotes the number of original frames recovered at best and the Y-axis represents the CDF of recovering these original frames. When the loss rate of the wireless network is very small, for example $r_l = 0.01$ in case (a), both of these two distributions can achieve 100% efficiency (recover all original frames). The expectations $(E_{sequential}, E_{improved\_soliton})$ of recovering original frames in first six cases (a), (b), (c), (d), (e) and (f) are (50, 50), (49.18, 49.22), (49.17, 49.22), (48.63, 48.68),(47.73, 47.78), (46.21, 46.28). Overall, the improved robust Soliton distribution performs marginally better. Thus the Frame Fountain with improved robust Soliton distribution as degree distribution works efficiently to its purpose.

V. Conclusion and Future Works

In this work, we propose a coding method called Frame Fountain, the first encoding method working directly on the data frames in the MAC layer to reduce retransmission due to frame loss on wireless lossy links and improve network performance. Different distributions are investigated to find an optimal degree distribution for this coding solution. Extensive experiments demonstrate that Frame Fountain with the improved robust Soliton distribution can effectively recover original frames in lossy environments. Our future work is to implement and evaluate this solution on a test bed in practice.

VI. Acknowledgement

This work is supported by the National Science Foundation through grant OCI#1041292.