Improved Intermediate Performance of Rateless Codes

Saejoon Kim and Seunghyuk Lee
Department of Computer Science and Engineering
Sogang University
Seoul, Korea
Email: {saejoon, shlee81}@sogang.ac.kr

***Abstract*—Rateless erasure resilient codes, *e.g.*, LT codes, can recover all message symbols perfectly with high probability from a slightly greater number of encoding symbols. However, it has been observed that only a few message symbols can be recovered from a less number of encoding symbols. In this paper, we investigate the performance of rateless codes when the number of received encoding symbols is less than that of the original message symbols. Through ordered generation of encoding symbols according to their degrees, we demonstrate that a significantly improved intermediate performance of rateless codes can be obtained.**

***Index Terms*—Rateless erasure resilient codes, intermediate performance, LT codes**

I. Introduction

Rateless erasure resilient codes such as LT codes [2] or Raptor codes [4] are universally capacity-achieving codes over the binary erasure channel. Given a set of message symbols, these codes produce a potentially infinite number of independently and identically generated encoding symbols on-the-fly and the set of message symbols can be recovered perfectly with high probability from any slightly greater number of encoding symbols. Moreover, these codes can be encoded and decoded in linear-time which makes them very appealing for various reliable data transmission applications.

On the other hand, while these rateless codes are capacity-achieving, these codes behave miserably when provided only with a number of encoding symbols that is less than that of the message symbols, *i.e.*, bad intermediate performance. For example, few message symbols will be recovered even if the number of encoding symbols received is only slightly less than the sufficient amount for full recovery. To this end, Sanghavi [3] has recently considered the asymptotic fraction of message symbols that can be recovered as a function of the number of received encoding symbols. While rateless codes presented in his work will perform very well for number of encoding symbols less than that of the message symbols, these codes no longer possess the capacity-achievability property and hence perform non-optimally when provided with a sufficient number of encoding symbols for full recovery. In a similar work, Kamra *et al.* [1] have presented *growth codes* for sensor network applications in which the degree of encoding symbols *grows* with respect to the number of message symbols recovered in order to facilitate good intermediate performance. Due to the large fraction of degree-one encoding symbols however, growth codes also do not achieve capacity and therefore suffer from the same problem as codes in Sanghavi's work.

In this work, we propose rateless codes that have both good intermediate performance and capacity-achievability property. Our codes are generated in a similar manner as growth codes, however from a capacity-achieving degree distribution in order to be able to recover all message symbols from a minimal number of received encoding symbols.

II. Code Description

We shall consider the class of LT codes as the rateless code and each symbol as a bit for simplicity in this paper. We will also consider message-passing decoding [2] as the decoding method.

Let $\mathcal{D} = (\Omega_1, \cdots, \Omega_k)$ represent a degree distribution where Ω_i is the probability that the value i is chosen and $\sum_{i=1}^{k} \Omega_i = 1$. Given a set of k message symbols, an LT code generates a stream of encoding symbols each of whose value is the modulo 2 sum of randomly and uniformly chosen i message symbols with probability Ω_i. Assume that any set of $k(1+\epsilon)$ received encoding symbols suffice to recover the k original message symbols with high probability for $\epsilon > 0$. A capacity-achieving degree distribution $\mathcal{D}$ means that for very large k, the value of ϵ is arbitrarily small.

An example of a capacity-achieving degree distribution for an LT code is the ideal soliton distribution (IDS) which has the following form

$$\rho(1) = \frac{1}{k}, \qquad \rho(i) = \frac{1}{i(i-1)}, \ 2 \le i \le k$$

where $\rho(i) \triangleq \Omega_i$. While IDS is capacity-achieving in theory, it is based on the assumption that the expected behavior of the message-passing decoder is the actual behavior and thus has nontrivial probability of decoding failure even with far more number of received encoding symbols than k. To circumvent this, a modified form of IDS called robust soliton distribution (RSD) is used in practice. RSD is also capacity-achieving and for $\mu(i) \triangleq \Omega_i$, it takes the form of

$$\mu(i) = \frac{\rho(i) + \tau(i)}{\beta}$$

where

$$\tau(i) = \begin{cases} \frac{R}{ik} & 1 \le i \le \frac{k}{R} - 1 \\ \frac{R \ln \frac{R}{\delta}}{k} & i = \frac{k}{R} \\ 0 & \frac{k}{R} + 1 \le i \le k \end{cases},$$

$R = c \cdot \ln \frac{k}{\delta} \cdot \sqrt{k}$ and $\beta = \sum_{i=1}^{k} \rho(i) + \tau(i)$. Here, c is a free parameter and δ is the allowed recovery failure probability that code designers can select. It can be verified through simulations that while LT codes defined by RSD are capacity-achieving, they behave miserably when provided only with encoding symbols whose size is less than $k(1+\epsilon)$.

To obtain good intermediate performance for LT codes, Sanghavi [3] has investigated the fraction of recoverable message symbols in terms of that of received encoding symbols. In [3], he has proved that, asymptotically, for rk received encoding symbols where $0 \le r \le \log 2$, $\Omega_1 = 1$ is the optimal degree distribution to recover the most number of message symbols possible, *i.e.*, no coding at all, whereas for $\log 2 \le r \le \frac{3}{4} \log 3$, the optimal degree distribution is $\Omega_2 = 1$. Result for $r \ge \frac{3}{4} \log 3$ is yet known.

In a similar line of work, however for sensor network applications, in order to essentially maximize the number of recovered message symbols provided with an arbitrary number of received encoding symbols, Growth codes [1] have been proposed. It was shown in [1] that Growth codes are optimal in terms of the number of recovered message symbols given any set of received encoding symbols, and in particular, to recover r message symbols where $r \le \frac{ik-1}{i+1}$, the optimal degree distribution has encoding symbols of degree no larger than i for $1 \le i \le k-1$. Letting $r_i = \frac{ik-1}{i+1}$, define the set of integers n_i's, $i = 1, 2, \cdots, k-1$, where

$$\begin{aligned} n_1 &= \sum_{j=0}^{r_1 - 1} \frac{k}{k-j} \\ n_i &= n_{i-1} + \sum_{j=r_{i-1}}^{r_i - 1} \frac{\binom{k}{i}}{\binom{j}{i-1}(k-j)}, \ 2 \le i \le k-1. \end{aligned}$$

Growth codes, like LT codes, generate encoding symbols by XOR'ing randomly and uniformly chosen message symbols in a way such that the first n_1 received encoding symbols are of degree one, the next $n_2 - n_1$ symbols of degree two, the next $n_3 - n_2$ symbols of degree three, and so forth. As with codes studies by Sanghavi [3], Growth codes perform very well with a smaller number of encoding symbols than the message symbols, however do not achieve capacity. Implicit in the construction of codes in [3] [1] is the knowledge of the number of received encoding symbols at the receiver side by the encoder in order to increase the degree of the next encoding symbols.

There are two problems with the approaches by [3] [1] for efficient data transmission over an erasure channel:

- These codes are not capacity achieving.
- The sender needs to know how many encoding symbols the receiver has received, *i.e.*, needs feedback, possibly up to k many times.

In an attempt to resolve these issues while requiring good intermediate performance, for our proposed code, we shall use a capacity-achieving degree distribution such as the RSD and generate encoding symbols in nondecreasing order of degrees up to some predetermined degree value. By not generating encoding symbols in nondecreasing order of degrees for all degree values, as will be verified in the next section, we will obtain codes that perform clearly better than otherwise and at the same time that require sufficiently smaller number of feedbacks than k times.

Now, supposing we are provided with a capacity-achieving degree distribution $\mathcal{D} = (\Omega_1, \cdots, \Omega_k)$, our proposed rateless code with feedback works as follows:

1. Estimate the value of $\epsilon(k, \delta)$ that will enable any set of $k(1+\epsilon)$ received encoding symbols to recover all message symbols with probabilty at least $1 - \delta$ for values of k and δ of interest.
2. Let the first $\Omega_1 k(1+\epsilon)$ received encoding symbols to be of degree one, the next $\Omega_2 k(1+\epsilon)$ received encoding symbols to be of degree two, and so forth until $\Omega_h k(1+\epsilon)$ encoding symbols of degree h are received for some $h (\ll k)$ to be determined later.
3. Remaining $\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)$ encoding symbols to be received are generated from the distribution $(\frac{\Omega_{h+1}}{\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)}, \frac{\Omega_{h+2}}{\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)}, \cdots, \frac{\Omega_k}{\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)})$ where $\frac{\Omega_j}{\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)}$ is the probability that the value j is chosen.
4. If more than $k(1+\epsilon)$ encoding symbols are to be received, these additional encoding symbols are generated from $\mathcal{D}$.

Thus after $\sum_{i=1}^{k} \Omega_i k(1+\epsilon) = k(1+\epsilon)$ encoding symbols are received using h feedbacks, all message symbols are recovered with probability at least $1 - \delta$. In particular, regardless of the value of h, all of our proposed codes will have the same asymptotic behavior as the original LT code for number of encoding symbols $\ge k(1+\epsilon)$. The rationale for not receiving all encoding symbols in nondecreasing order of degree is due to the fact that after a number, say h, of encoding symbols are received in nondecreasing order, calculation can show that the next encoding symbols to be received should have degree $> h+1$ to maximize symbol recovery success probability. There is a partial converse result in this direction from [3] which is stated in the next lemma.

Lemma 1 ([3]): If there exists $m \ge 1$ such that $z < \frac{m}{m+1}$, then it has to be that the optimal degree distribution to recover z fraction of message symbols has $\Omega_i = 0$ for all $i \ge m+1$.

According to the lemma, for $m \ge 99$ for example, the optimal degree distribution should not have degrees greater than 100. On the other hand, when all encoding symbols are generated in nondecreasing order of degree from RSD, simulation results show that for $k = 1000$ and $z < 0.99$ for example, the largest encoding symbol degree is less than 10 which is much less than 100. Similar results hold for smaller values of m. Thus while the lemma implies that encoding symbol degrees

should be nondecreasing for very small degree values, it does not imply that all encoding symbols should be generated in nondecreasing order of degree for optimal intermediate performance as encoding progresses. In perspective of minimizing the number of feedbacks necessary for our proposed code, the remaining $\sum_{i=h+1}^{k} \Omega_i k(1+\epsilon)$ encoding symbols of the $k(1+\epsilon)$ symbols are encoded randomly according to the distribution shown in step 3 in the above algorithm without trying to satisfy the result of the lemma with the aid of possibly a sufficiently more number of feedbacks. Furthermore, because Ω_i values for large i are very small, we found the performance differential to be negligible.

Table I shows a partial list of values of Ω_i and $\sum_{j=1}^{i}(1+\epsilon)\Omega_j$ of RSD for $k = 250, 1000$ and 4000. As will be shown through simulations in the next section, it turns out that h such that $\sum_{j=1}^{h}(1+\epsilon)\Omega_j \approx 1$ gives the best intermediate performance for all values of $h = 1, 2, \cdots, k$. Hence for our proposed rateless codes with feedback, only about $3, 4$ and 11 feedbacks are necessary for $k = 250, 1000$ and 4000, respectively.

TABLE I
DISTRIBUTION (PARTIAL) OF RSD FOR VARIOUS k'S

	i	Ω_i	$\sum_{j=1}^{i}(1+\epsilon)\Omega_j$
$k = 250$	1	0.040484	0.089065
$(\epsilon = 1.2)$	2	0.316527	0.785424
	3	0.11186	1.031515
	4	0.059105	1.161546
$k = 1000$	1	0.025998	0.044196
$(\epsilon = 0.7)$	2	0.360148	0.656447
	3	0.124266	0.8677
	4	0.064242	0.976911
	5	0.03981	1.044588
$k = 4000$	7	0.020965	0.942671
$(\epsilon = 0.32)$	8	0.01601	0.963804
	9	0.012674	0.980534
	10	0.010317	0.994153
	11	0.008587	1.005488
	12	0.007277	1.015093
	13	0.00626	1.023356

III. SIMULATION RESULTS

In this section, we demonstrate the empirical results of intermediate and asymptotic performances of various rateless codes discussed in the previous section. The first type is Luby's original LT code defined by RSD [2] denoted by LT. The second type is our proposed rateless codes defined by RSD using feedback. We denote by LT-h our proposed code whose encoding symbols are sorted up to degree h. For example, LT-k represent the LT code all of whose encoding symbols are sorted in nondecreasing order according to their degrees. Moreover, let h^* be the value of h where LT-h^* has the best performance among LT-h, $h = 1, 2, \cdots, k$. The last type of rateless code we consider is the Growth codes, denoted by GC, which possess very good intermediate performance.

Figures 1 and 2 show the message symbol recovery failure probability with respect to $\frac{n}{k}$ curves for $n \leq k$ and $n \geq k$, respectively, where $k = 250$ and n is the number of received encoding symbols. Note that the y-axes of the two figures are of different scale, *e.g.*, Fig. 2 is shown in log-scale. In these figures we let $\epsilon(250, 0.001) = 1.2$ for simulations of LT-h. Figure 1 indicates that LT performs poorly while GC performs very well for $n < k(1+\epsilon)$, but for GC, the last i and n_i for which $n_i - n_{i-1}$ is nonzero is $i = 125$ and $n_{125} = 363$, respectively, and thus Growth codes cannot generate more than 363 encoding symbols as can be witnessed by the curve shown in Fig. 2. We have tested generating $n(> 363)$ encoding symbols where the last $n - 363$ symbols are of degree 125, however the curve extended horizontally, *i.e.*, error floor, from the point corresponding to $\frac{n}{k} = \frac{363}{250}$ up to $\frac{n}{k} = 2$.

For our proposed rateless codes with feedback, all simulations we have tested in this work indicate that the overall performance of LT-h steadily improves until some point, namely h^*, and then deteriorates as the value of h is incremented. For this reason, in all figures in this paper for LT codes with feedback, we have included only the LT-h curves for values of h in the neighborhood of h^* and the LT-k curve for reference. For $k = 250$, LT-3 gave the best intermediate and asymptotic performance among all LT-h, $h = 1, 2, \cdots, k$.

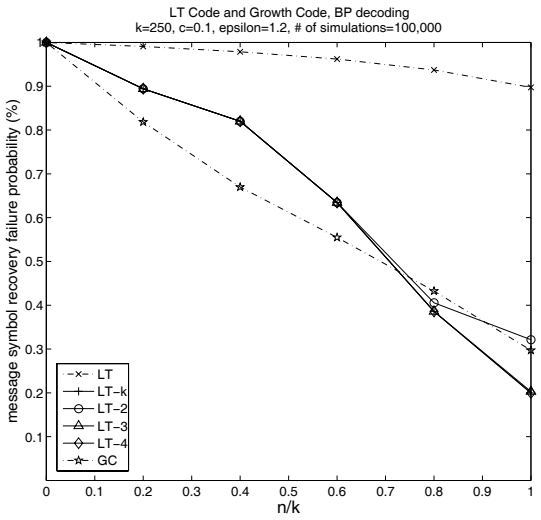

Fig. 1. Code Performance when $n \leq k = 250$

Figures 3 and 4 show the similar curves as Figs. 1 and 2, respectively, when $k = 1000$. For these figures, we let $\epsilon(1000, 0.001) = 0.7$ for simulations of our rateless codes with feedback. Similar to Figs. 1 and 2, LT shows the worst intermediate performance, GC performs the best for small values of $\frac{n}{k}$, and LT-4 performs the best for most of all other values of $\frac{n}{k}$. Note in particular that the performance improvement of LT-h^* over GC is more pronounced in these figures than that in Figs. 1 and 2. This is due to the fact that while LT codes defined by RSD are asymptotically capacity-achieving codes, Growth codes are not and thus the performance differential

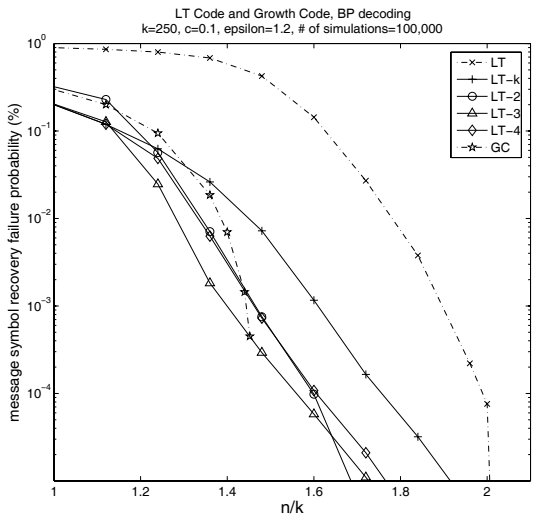

Fig. 2. Code Performance when $n \geq k = 250$

between the two types of codes will widen as k increases. This behavior will be observed again in $k = 4000$ case.

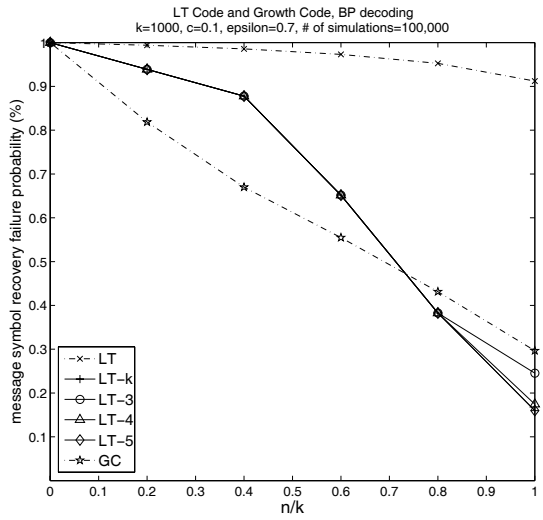

Fig. 3. Code Performance when $n \leq k = 1000$

Figures 5 and 6 show the similar curves as Figs. 1 and 2, respectively, when $k = 4000$. Here, we let $\epsilon(1000, 0.001) = 0.32$. Similar to the previous cases, LT-11 provides with the best graceful degradation of message symbol recovery while performing optimally as an increasing number of encoding symbols is received. Figure 7 shows a close-up view of Fig. 6 for $1 \leq \frac{n}{k} \leq 1.2$. The performance improvement of LT-h^* over GC is even more pronounced in these figures than that demonstrated in the previous cases by the same reason mentioned before.

IV. Conclusion

In this paper, the intermediate performance of rateless erasure resilient codes was considered. We have proposed a slight variant of the original LT codes through generating encoding symbols in nondecreasing order of their degrees through feedback from a capacity-achieving degree distribution. Simulation results indicate that our proposed codes perform always better than or similar to the original LT codes for all fraction of received encoding symbols while better than the Growth codes for a large fraction of received encoding symbols. Furthermore, results also indicate that a relatively small number of feedbacks are necessary to achieve the performance presented.

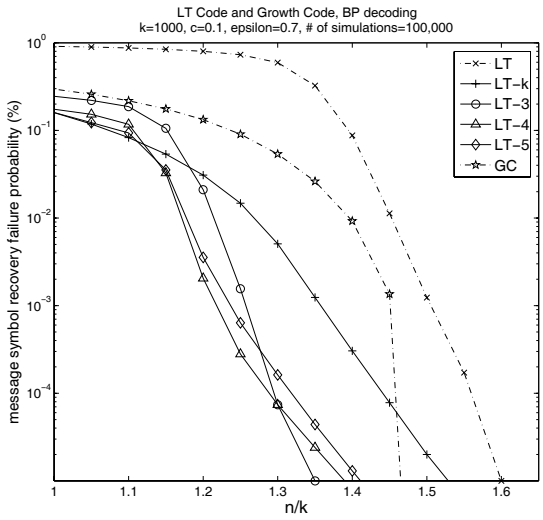

Fig. 4. Code Performance when $n \geq k = 1000$

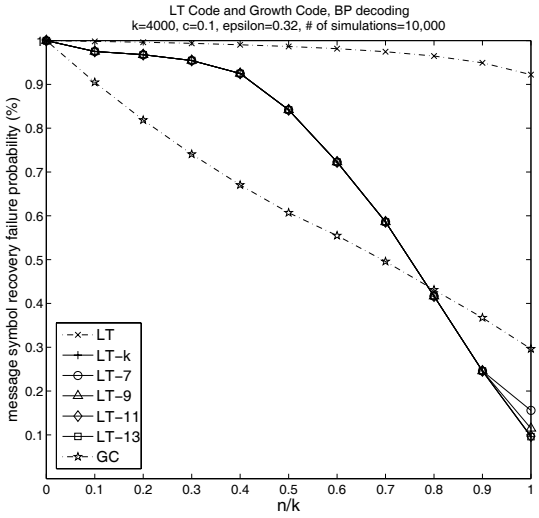

Fig. 5. Code Performance when $n \leq k = 4000$

References

[1] A. Kamra, J. Feldman, V. Misra and D. Rubenstein, "Growth codes: maximizing sensor network data persistence," *Proc. ACM SIGCOMM*, 2006.

[2] M. Luby, "LT-codes," *Proc. ACM Symposium on Foundations of Computer Science*, 2002.

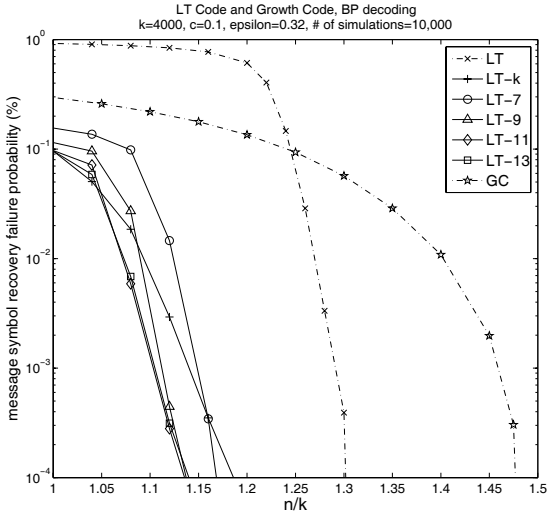

Fig. 6. Code Performance when $n \geq k = 4000$

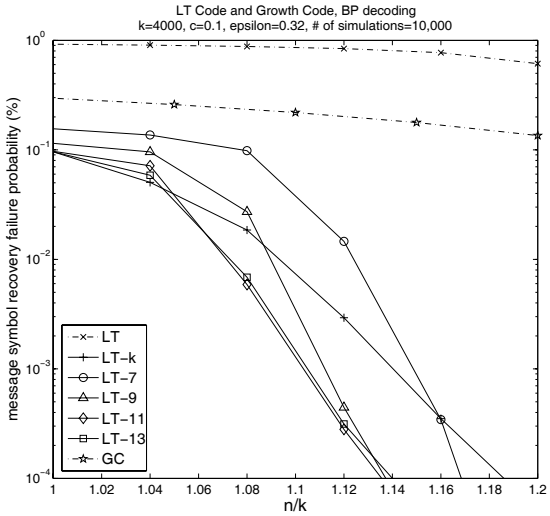

Fig. 7. Close-Up View of Fig. 6

[3] S. Sanghavi, "Intermediate performance of rateless codes," *Proc. Inform. Theory Workshop*, 2007.
[4] M.A. Shokrollahi, "Raptor Codes," *IEEE Trans. on Inform. Theory*, vol. 52 no. 6, pp. 2551-2567, June 2006.